UNIVERSITAS MUHAMMADIYAH SIDOARJO
FAKULTAS PSIKOLOGI DAN ILMU PENDIDIKAN (FPIP)
PRODI PENDIDIKAN GURU ANAK USIA DINI (PG-PAUD) TERAKREDITASI B NOMOR : 2231 SK/BAN-PT/Akred/S/VII/2017
PRODI PENDIDIKAN GURU SEKOLAH DASAR TERAKREDITASI B NOMOR : 743/SK/BAN-PT/Akred/S/III/2018
PRODI PENDIDIKAN BAHASA INGGRIS TERAKREDITASI B NOMOR : 3057/SK/BAN-PT/Akred/S/XI/2018
PRODI PENDIDIKAN ILMU PENGETAHUAN ALAM (IPA) TERAKREDITASI B NOMOR : 432/SK/BAN-PT/Akred/S/III/2019
PRODI PENDIDIKAN TEKNOLOGI INFORMASI (TI) SK NOMOR : 0207/SK/BAN-PT/Akred/S/I/2017
PRODI PSIKOLOGI TERAKREDITASI B NOMOR : 0124/SK/BAN-PT/Akred/S/III/2016
TERAKREDITASI INSTITUSI (UNIVERSITAS) B
BAN-PT No. 225/SK/BAN-PT/Akred/PT/2019
Kampus I : Jl. Mojopahit 666 B Sidoarjo, Telp: 031 – 8945444, Fax: 031-8949333
Website: www.fpip.umsida.ac.id email:fpip@umsida.ac.id

SURAT TUGAS

Nomor: 189/II.3.AU/08.00/B/KEP/VII/2019

Yang bertanda tangan di bawah ini;

1. Nama : Dr. Akhtim Wahyuni, M.Ag
2. NIK : 202200
3. Pangkat/ Golongan : Lektor/ IIId
4. Jabatan : Dekan Fakultas Psikologi dan Ilmu Pendidikan
5. Unit Kerja : Fakultas Psikologi dan Ilmu Pendidikan

Menugaskan:

1. N a m a : **Fitria Nur Hasanah, M.Pd**
2. NIK : 216613
3. Pangkat Golongan : Asisten Ahli/ IIIb
4. Jabatan : Dosen Tetap
5. Unit Kerja : Fakultas Psikologi dan Ilmu Pendidikan

Untuk membuat buku ajar mata kuliah Basis Data di Fakultas Psikologi dan Ilmu Pendidikan Universitas Muhammadiyah Sidoarjo Semester Ganjil Tahun Akademik 2019/2020. Demikian surat tugas ini dibuat, untuk dapat dipergunakan sebagaimana mestinya.

Sidoarjo, 20 Agustus 2019
DEKAN FPIP,

Dr. Akhtim Wahyuni, M.Ag

BUKU AJAR
BASIS DATA

Fitria Nur Hasanah, M.Pd
Rahmania Sri Untari, M.Pd

BUKU AJAR MATA KULIAH
BASIS DATA

Oleh
Fitria Nur Hasanah, M.Pd
Rahmania Sri Untari, M.Pd

Diterbitkan oleh
UMSIDA PRESS
Jl. Mojopahit 666 B Sidoarjo

ISBN : 978-602-5914-89-8

Buku Ajar
BASIS DATA

Penulis :
Fitria Nur Hasanah, M.Pd
Rahmania Sri Untari, M.Pd

ISBN : 978-602-5914-89-8

Editor :
Septi Budi Sartika, M.Pd
M. Tanzil Multazam , S.H., M.Kn.

Copy Editor :
Fika Megawati, S.Pd., M.Pd.

Design Sampul dan Tata Letak :
Mochamad Nashrullah, S.Pd

Penerbit :
UMSIDA Press

Redaksi :
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit No 666B
Sidoarjo, Jawa TImur

Cetakan pertama, September 2019

KATA PENGANTAR

Alhamdulillah Puji Syukur Penulis sampaikan kehadirat Allah SWT, yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Buku Ajar Basis Data dengan baik. Buku ajar ini ditujukan kepada mahasiswa yang mengampuh mata kuliah Basis data. Dengan dibuatnya Buku Ajar ini penulis berharap agar dapat bermanfaat dan membantu dalam memahami materi Basis Data yang semakin berkembang.

Ucapkan terima kasih penulis ucapkan kepada :

1. Dr. Hidayatulloh, M.Si., Rektor UMSIDA yang memberikan kesempatan luas kepada tim penulis untuk berkarya dan menyumbangkan pikiran sehingga buku ajar ini terselesaikan.
2. Dr. Akhtim Wahyuni, M.PdI Dekan Fakultas Psikologi dan Ilmu Pendidikan yang memberikan arahan dan motivasi kepada penulis dalam menyelesaikan buku ajar Basis Data ini.
3. Rekan-rekan dosen di lingkungan FPIP UMSIDA yang telah berbagi pengalaman dalam penulisan buku ajar.

Penulis menyadari sekali bahwa Buku Ajar ini masih jauh dari kesempurnaan, maka dari itu penulis mengharapkan kritik dan saran pembaca demi kesempurnaan Buku Ajar ini kedepannya. Akhir kata penulis mengucapkan terima kasih, mudah-mudahan bermanfaat bagi para pembaca.

Sidoarjo, 2019
Penulis

DAFTAR ISI

BAB I
PENDAHULUAN

1. Deskripsi

Data merupakan hal yang penting bagi kehidupan manusia, pernyataan ini tidak dapat dipungkiri karena setiap hari manusia memrlukan dan menggunakan data dalam merancang segala sesuatu. Contoh sederhana mahasiswa ketika akan mengerjakan tugas tentu membutuhkan data untuk menyelesaikan tugasnya. Mahasiswa memerlukan data antara lain data tugas, data mata kuliah, data dosen, nilai ujian, dan lain-lain. Sementara itu dapatkan membayangkan kebutuhan mahasiswa atau organisasi lain terhadap data yang dibutuhkan? Saat ini baik instansi pendidikan maupun perusahaan sudah memiliki system informasi dan aplikasi. System informasi dan aplikasi yang terdapat pada instansi memerlukan data. Sehingga antara system informasi dan data memiliki hubungan yang erat dan tidak dapat dipisahkan.

Basis data adalah kumpulan informasi yang disimpan di dalam komputer secara sistematik sehingga dapat diperiksa menggunakan suatu program komputer untuk memperoleh suatu informasi dari database tersebut. Perangkat lunak yang digunakan untuk mengolah dan mengambil query basis data disebut sistem manajement basis data. Pemrosesan basis data sebagai perangkat andalan sangat diperlukan oleh berbagai institusi dan perusahaan. Dalam pengembangan sistem informasi diperlukan basis data sebagai media penyimpanan data. Kehadiran basis data dapat meningkatkan daya saing perusahaan tersebut.

Basis data dapat mempercepat upaya pelayanan kepada pelanggan, menghasilkan informasi dengan cepat dan tepat sehingga membantu pengambilan keputusan untuk segera memutuskan suatu masalah berdasaran informasi yang ada. Banyak

aplikasi yang dibuat dengan berlandaskan pada basis data antara lain semua transaksi perbankan, aplikasi pemesanan, penjadwalan penerbangan, proses registrasi dan pencatatan data mahasiswa pada perguruan tinggi, aplikasi pemrosesan penjualan, pembelian dan pecatatan data barang pada perusahaan dagang, pencatatan data pegawai beserta akivitasnya termasuk operasi penggajian pada suatu perusahaan, dan sebagainya. Beberapa informasi pada perusahaan retail seperti jumlah penjualan, mencari jumlah stok yang tersedia, barang apa yang paling laku dijual pada bulan ini, dan berapa laba bersih perusahaan dapat diketahui dengan mudah menggunakan basis data.

2. Kemampuan Akhir

Setelah mempelajari uraian materi dalam bab pembelajaran dan kegiatan belajar diharapkan mahasiswa dapat memiliki kompetensi sikap, pengetahuan dan ketrampilan yang berkaitan dengan materi yang ditunjukkan pada Gambar 1.1

Gambar 1.1 Peta Konsep Materi Basis Data

Bab II
KONSEP BASIS DATA

Setelah mempelajari bab ini, mahasiswa diharapkan mampu :

1. Menjelaskan definisi basis data
2. Menjelaskan perbedaan file tradisional dan basis data
3. Mengklasifikasikan komponen basis data
4. Menjelaskan sistem manajemen basis data
5. Menjelaskan tujuan dan manfaat basis data

1. Definisi Basis Data

Pengertian basis data dapat ditinjau dari dua sisi, secara kharfiah dan pengertian secara istilah. Menurut pengertian secara kharfiah, basis data terdiri dari dua kata yaitu basis dan data. Basis dapat diartikan sebagai suatu markas atau gudang, tempat bersarang atau tempat berkumpul. Data dapat diartikan merupakan representasi dari fakta dunia yang mewakili suatu obyek (manusia, barang, peristiwa, keadaan dsb) yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi atau kombinasinya.Adapun menurut pengertian secara istilah, terdapat beberapa definisi yaitu sebagai berikut :

a. Himpunan kelompok data (arsip) yang saling berhubungan yangdiorganisasi sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah
b. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundancy) yang tidak perlu, untuk memenuhi berbagai kebutuhan

c. Kumpulan file/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan tertentu.
d. Kumpulan data, yang dapat digambarkan sebagai aktifitas dari satu atau lebih organisasi yang berelasi.

Menurut Elmasri, penggunaan istilah basis data lebih dibatasi pada arti implisit yang khusus mempunyai beberapa pengertian, yaitu :

a. Basis data merupakan penyajian suatu aspek dari dunia nyata (*real word* atau *miniworld*). Misalnya basis data perbankan, perpustakaan, pertanahan, perpajakan
b. Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implicit. Sehingga apabila data terkumpul secara acak dan tanpa mempunyai arti, tidak dapat disebut basis data.
c. Basis data perlu diancanag, dibangun dan data dikumpulkan untuk suatu tujuan tertentu.
d. Basis data dapat digunakan oleh beberapa pemakai dan beberapa aplikasi yang sesuai dengan kepentingan pemakai.

2. Perbedaan File Tradisional dan Basis Data

File Tradisional adalah file dimana setiap user mengimplementasikan file yang dibutuhkan untuk aplikasi khusus sebagai bagian dari pemrograman aplikasinya. Basis data adalah sekumpulan file-file yang mempunyai kaitan antara satu file dengan file lain sehingga membentuk suatu bangunan data untuk menginformasikan suatu perusahaan atau instansi dalam batasan tertentu.

Perbedaan File Manajemen Tradisional & File Manajemen Basis Data

Perihal	File Tradisional	Manajemen Basis Data
Orientasi program	Sering terjadi kerangkapan data Kaku	Terkontrolnya kerangkapan data Luwes
Kelemahan	Timbulnya data rangkap & ketidak konsistenan Data tidak dapat digunakan bersama-sama Kesukaran dalam pengaksesan data Tidak fleksibel Data tidak standar	Storage yang digunakan besar Dibutuhkan tenaga spesialis Softwarenya mahal Kerusakan pada system database dapat mempengaruhi departemen lain yang terkait

3. Komponen Sistem Manajemen Basis Data

Basis data adalah merupakan suatu sistem yang dibangun oleh beberapa komponen diantaranya ada enam komponen pokok antara lain ialah:

a. Perangkat keras (hardware) dalam sistem komputer. Dalam sistem pengolahan basis data digital perangkat utama sebagai pengolah data dalah komputer.
b. Perangkat Lunak Aplikasi (software) lain yang mendukung dan bersifat opsional. Perangkat lunak digunakan untuk mendukung proses pengelolaan basis data. Misal: bahasa pemrograman C, basic pascal.

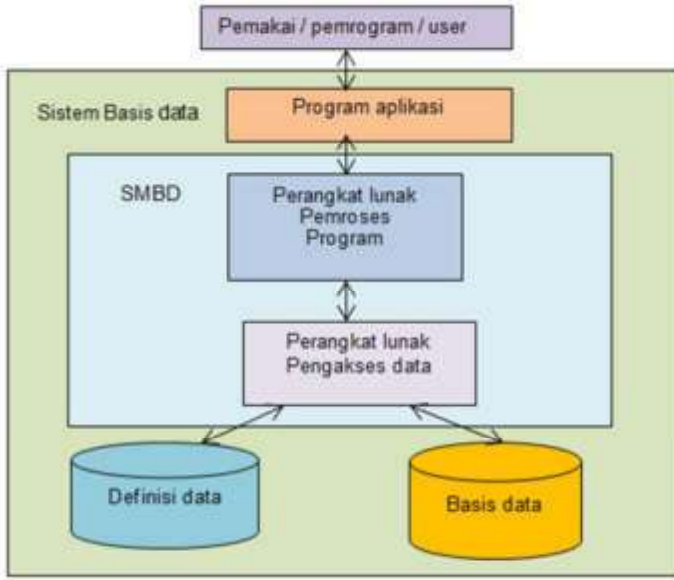
c. Sistem Operasi (operating system). Sistem operasi merupakan perangkat lunak yang digunakan untuk mengelola aplikasi basis data dan penggunaan sumberdaya komputer.
d. Basis data data lain yang mempunyai keterkaitan dan hubungan dengan basis data itu sendiri. Berisi atau memiliki objek-objek basis data seperti file, table, indeks. Mempunyai disfinisi struktur baik untuk basis data maupun objek-objek secara detail.
e. Sistem Pengelola Basis Data Database Management System atau database managemen system (DBMS). Merupakan program aplikasi untuk pengelolaan basis data, seperti Microsoft acces, oracle dan lian-lain
f. Pemakai (user), yaitu pengguna yang terlibat dalam pengelolaan basis dan penggunaan basis data

4. Sistem Manajemen Basis Data

Sistem manajemen basis data adalah merupakan sebuah tatanan (keterpaduan) yang terdiri atas sejumlah komponen-komponen fungsional (komputer) yang saling berhubungan secara bersama-sama, bertujuan untuk memenuhi suatu proses atau pekerjaaan tertentu. Sistem ini merupakan gabungan antara basis data dan kumpulan program atau perangkat lunak DBMS (d*atabase management system).*

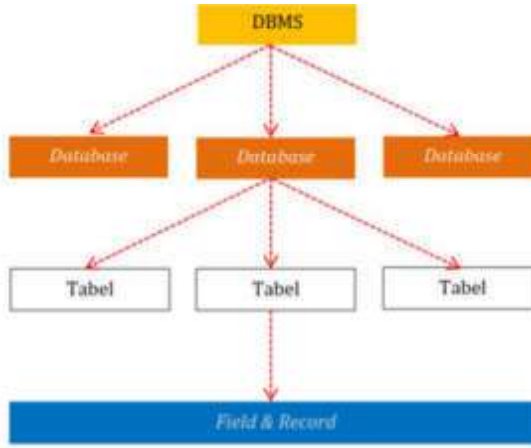
*DBMS a*dalah program aplikasi yang dibuat dan bekerja dalam satu system. *DBMS* didesain untuk membantu dalam hal pemeliharaan dan utilitas kumpulan data dalam jumlah besar. DBMS dapat menjadi alternatif penggunaan secara khusus untuk aplikasi, misalnya penyimpanan data dalam field dan menulis kode aplikasi yang spesifik untuk pengaturannya. Kumpulan file (table) yang saling berhubungan dalam di sebuah komputer dan sekumpulan program yang memungkinkan beberapa pemakai dan

atau program lain untuk mengakses dan memanipulasi file-file atau table-tabel tersebut.



Gambar 2.1. Konsep DBMS

Hubungan DBMS dengan basis data, tabel, field, dan record ditunjukkan pada Gambar 2.2.



Gambar 2.2 Hubungan DBMS dan basis data

5. Tujuan dan Manfaat Basis Data

Kesuksesan suatu organisasi bergantung pada kemampuannya menangkap data secara akurat dan tepat waktu. Hal tersebut berkaitan dengan operasi dan pengaturan data secara efektif, maupun penggunaan data untuk keperluan analisis untuk kebutuhan pendukung keputusan. Kemampuan untuk mengatur atau mengolah sejumlah data, dan kecepatan untuk mencari informasi yang relevan, adalah aset yang sangat penting bagi suatu organisasi.Beberapa tujuan penggunaan basis data adalah sebagai berikut :

1. Kecepatan dan Kemudahan (*Speed*) , melalui basis data diharapkan pengguna dapat melakukan penyimpanan, perubahan dan menampilkan kembali dengan cepat dan mudah.
2. Efisiensi Ruang Penyimpanan (*Space*). Penggunaan basis data mampu mengurangi pengulangan atau redundansi data. Hal ini dapat dilakukan dengan menerapkan sejumlah pengkodean atau

dengan membuat relasi-relasi (dalam bentuk file) antara kelompok data yang saling berhubungan.

3. Keakuratan (*Accuracy*), melalui basis data data keakuratan data lebih terjaga dengan menerapkan aturan dan batasan tertentu (*constraint*), tipe data, domain data dan keunikan data
4. Ketersediaan (*Availability*). Dengan basis data data yang sudah tidak dipakai dapat dipisahkan dari sistem database yang sedang aktif. Hal ini dapat dilakukan dengan cara penghapusan atau memindahkannya ke media backup untuk menghemat ruang penyimpanan. Selain itu dapat memanfaatkan teknologi jaringan komputer agar data yang berada di suatu lokasi atau cabang daat juga diakses oleh lokasi atau cabang lainnya.
5. Kelengkapan (*Completeness*). Agar data yang dikelola senantiasa lengkap baik relatif terhadap kebutuhan pemakai maupun terhadap waktu. Hal ini dapat dilakukan melaluipenambahan record-record data, perubahan struktur basis data, menambah field pada tabel atau menambah tabel baru.
6. Keamanan (*Security*). Walaupun tidak semua sistem basis data menerapkannya, keamanan dalam penggunaan basis data diperlakukan pada sistem yang besar dan serius. Dengan penerapan ini, setiap pengguna dibedakan hak aksesnya; yakni ditentukan obyek-obyek mana saja yang bisa diakses dan proses apa saja yang bisa dia dilakukan.
7. Kebersamaan (*Sharability*). Agar data yang dikelola oleh sistem mendukung lingkungan multiuser (banyak pemakai) dengan menjaga / menghindari munculnya problem baru seperti *inkonsistensi data* (karena terjadi perubahan data yang dilakukan oleh beberapa user dalam waktu yang bersamaan) atau kondisi *deadlock* (karena ada banyak pemakai yang saling menunggu untuk menggunakan data).

EVALUASI

1. Jelaskan beberapa pengertian atau definisi dari basis data!
2. Jelaskan mengapa seseorang memerlukan basis data?
3. Sebutkan beberapa contoh penerapan basis data dalam kehidupan sehari-hari!
4. Jelaskan perbedaan antara basis data dengan system manajemen basis data?

BAB III
ELEMEN LINGKUNGAN BASIS DATA

Setelah mempelajari bab ini, mahasiswa diharapkan mampu :

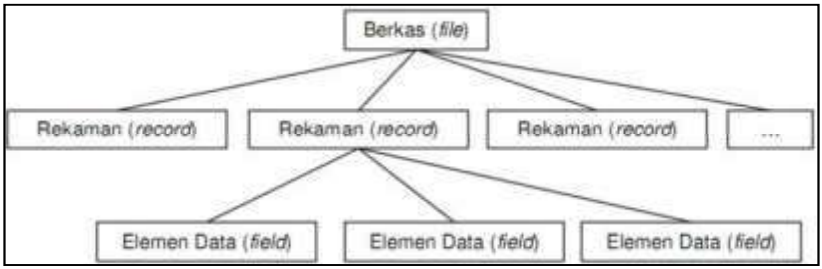
1. Menjelaskan arsitektur basis data
2. Mengklasifikasikan pengguna dalam basis data
3. Menjelaskan file tabel dan file record

1. Arsitektur Basis Data

Arsitektur basis data merupakan serangkaian pengetahuan tentang pemodelan data. Pengetahuan tentang File, table, field, record indeks, abstraksi data dan serangkaian konsep yang digunakan untuk membuat diskripsi struktur basis data. Melalui diskripsi Struktur basis data dapat ditentukan jenis data, hubungan dan konstrain (keterbatasan) data yang ditangani. Dalam basis data, data diorganisasikan kedalam bentuk elemen data (field), rekaman (record), dan berkas (file). Definisi dari ketiganya adalah sebagai berikut:

a. Elemen (kolom atau field) data adalah satuan data terkecil yang tidak dapat dipecah lagi menjadi unit lain yang bermakna. Misalnya data siswa terdiri dari NIS, Nama, Alamat, Telepon atau Jenis Kelamin.
b. Rekaman (record) merupakan gabungan sejumlah elemen data yang saling terkait. Istilah lain dari record adalah baris atau tupel.
c. Berkas(file) adalah himpunan seluruh record yang bertipe sama

Struktur hirarki sebuah database dapat digambarkan dalam diagram hirarki Gambar 3.1



Gambar 3.1. Struktur hirarki sistem basis data

2. Pengguna Basis Data

Pada tingkat pemakai, data base dikelompokkan menjadi beberapa tingkat pemakai yaitu antara lain sebagai berikut:

a. Database Administrator, ialah manusia yang mengorganisasi seluruh sistem basis data. Database adaministrator imemiliki tanggung jawab penuh dalam manajemen database meliputi: pengaturan hak akses, koordinasi dan monitoring serta bertanggung jawab terhadap kebutuhan hardware dan software. Dalam pekerjaannya biasanya dibantu oleh staf Admin.
b. *Database Designer*, adalah manusia yang bertugas merancang dan mengembangkan database. Database designer bertanggung jawab dalam identifikasi data yang tersimpan dalam database, menentukan struktur data yang tepat untuk disimpan dalam database. Database designer memerlukan koordinasi akan kebutuhan user database.
c. *Application Programmer,* ialah penggunaa yang berinteraksi dengan basis data melalui *Data Manipulation Language* (DML). DML meliputi program yang ditulis dalam bahasa pemrograman induk yang dipakai.

d. *End user*, adalah adalah pengguna yang memanfaatkan atau membutuhkan akses ke database melalui query, manambah, merubah menghapus maupun membuat *report database*. *End user*dapat dikategorikan:
   a) *Casual end users* atau pengguna tak tetap atau user mahir. Pengguna yang tidak selalu mengakses database, tapi kadang memerlukan informasi terbaru. Berinteraksi dengan sistem tanpa modul program, hanya menggunakan *query* (untuk akses dan manipulasi data) yang telah disediakan oleh DBMS.
   b) *Natïve* atau *parametric end users* atau user umum. Pengguna yang pekerjaan selalu konstan yaitu melakukan query dan update data. Misalnya:*bank teller*, pegawai reservasi. Pengguna ini berinteraksi dg sistem melalui pemanggilan suatu program aplikasi permanen (executable) yang telah dibuat sebelumnya oleh programmer.
   c) User Khusus (Specialized User). Pengguna yang menulis aplikasi basis data *non konvensional* untuk keperluan khusus yang bisa saja mengakses basis data dengan atau tanpa DBMS yang bersangkutan.
   d) *Sophisticated end users*. pengguna yang melengkapi kebutuhan database user, seperti engineer, scientist, business analyst.
   e) Stand-alone users. penggunauser yang mengelola personal database.
e. *System Analyst*, ialah pengguna yang merencanakan dan menentukan kebutuhan sistem.
f. *Application Programmers* (Software Engineering), ialah pengguna tanggungjawabnya berhubungan dengan kebutuhan koneksi database.

g. *Worker behind the scene, ialah pengguna* yang tidak tertarik pada database, tetapi lebih cenderung pada membangun data base atau kebutuhannya menggunakan alat bantu. Pengguna ini dibedakan menjadi

a) DBMS system designers dan implementer, ialah pengguna yang merancang dan mengimplementasikan modul-modul dan interface menggunakan paket-paket software DBMS. (seperti:. Modul: catalog, *procs query lang., procs interface, access & buffering data, controlling cuncurrency, handling data recovery & security; interfacing: interface for integrated system*).

b) *Tool developers*. Pengguna yang merancang dan mengimplementasikan tools untuk mendukung software DBMS. Seperti Tools untuk meningkatkan performance database, tool untuk monitoring operasional database.

c) *Operators dan maintenance personnel*. Para personel administrator yang bertanggung jawab akan jalannyaoperasional database termasuk maintenance (hardware/software) DBMS.

3. File Table dan File Record

Field adalah kumpulan dari karakter yang membentuk satu arti, maka jika terdapat field misalnya seperti NomerBarang atau NamaBarang, maka yang dipaparkan dalam field tersebut harus yang berkaitan dengan nomer barang dan nama barang. Atau definisi field yang lainnya yaitu tempat atau kolom yang terdapat dalam suatu tabel untuk mengisikan nama-nama (data) field yang akan di isikan.

Record adalah kumpulan field yang sangat lengkap, dan biasanya dihitung dalam satuan baris. Tabel adalah merupakan kumpulan dari beberapa record dan juga field. File adalah terdiri

dari record-record yang menggambarkan dari satu kesatuan data yang sejenis. Misalnya seperti file nama barang berisikan data tentang semua nama barang yang ada. Data adalah kumpulan fakta atau kejadian yang digunakan sebagai penyelesaian masalah dalam bentuk informasi. Pengertian basis data (database) adalah basis data yang terdiri dari dua kata, yaitu kata basis dan data. Basis dapat di artikan markas ataupun gudang, maupun tempat berkumpul.

Gambar 3.2 Field dan record

EVALUASI

1. Jelaskan secara singkat arsitektur atau struktur basis data!
2. Jelaskan secara singkat pengertian struktur fisik basis data!
3. Jelaskan urutan pengguna basis data berdasarkan prioritas tingkat pemakai!

BAB IV
MODEL DATA RELATIONAL

Setelah mempelajari bab ini, diharapkan mahasiswa mampu:
1. Memahami konsep pemodelan data
2. Menjelaskan model data relasional
3. Menjelaskan istilah dalam model data relasional
4. Menjelaskan kunci dalam model data relasional
5. Menjelaskan keuntungan model data relasional

1. Konsep Pemodelan Data

Pemodelan data merupakan sarana untuk melakukan abstraksi data. Merupakan sejumlah konsep untuk membuat diskripsi stuktur basis data. Kebanyakan model data memuat spesifikasi untuk operasi dasar (*basic operation)* dalam pengaksesan dan pembaharuan data. Pada perkembangan terakhir dikenal dengan istilah tabiat data (*data behavior*) pada pemrograman berorientasi object. Terdapat sejumlah cara dalam merepresentasikan model dalam perancangan basis data. Secara umum pemodelan data dapat dikelompokkan menjadi dua yaitu :

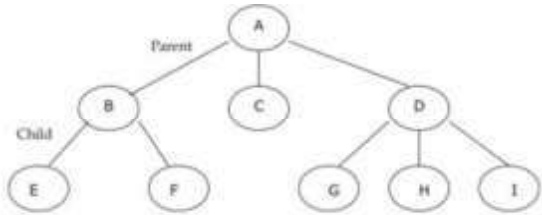
a. *Object based logical model*. Dalam pemodelan ini struktur atau hirarki basis data diilustrasikan berdasarkan object. Model ini meliputi: 1) Model keterhubungan entitas (Entity Relationalship Model atau ERD). 2) Model berorientasi object (Object-Oriented Model). 3) Model Data Semantik(Semantic Data Model). 2) Model data Fungsional (Function Data Model).
b. *Record-based logical model*. Dalam model ini struktur basis data diilustrasikan berdasarkan record. Model ini meliputi: 1) Model

relational (*Relational Model*). 2) Model Herarkis (Hierarchical Model) 3) Model Jaringan (*Network Model).*

Pada record based data model disamping digunakan untuk menguraikan struktur logika keseluruhan dari suatu database, juga digunakan untuk menguraikan implementasi dari system database (*higher level description of implementation*) Terdapat 3 data model pada record based data model :

1) Model Relasional,
   Dimana data serta hubungan antar data direpresentasikan oleh sejumlah table, dan masing -masing table terdiri dari beberapa kolom yang namanya unique. Model ini berdasarkan notasi teori himpunan (set theory), yaitu relation.
   Contoh :
   data base penjual barang terdiri dari 3 tabel :
   – Supllier
   – Path (Suku_cadang)
   – Delivery (pengiriman)

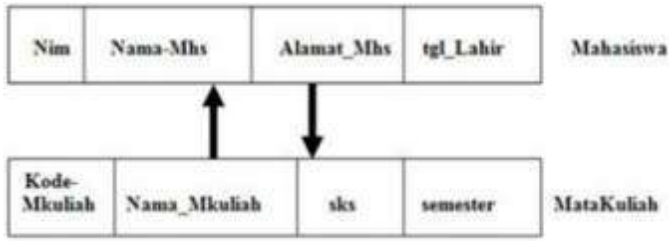
2) Model Hirarki
   Dimana data serta hubungan antar data direpresentasikan dengan record dan link (pointer), dimana record-record tersebut disusun dalam bentuk tree (pohon), dan masing-masing node pada tree tersebut merupakan record/grup data elemen

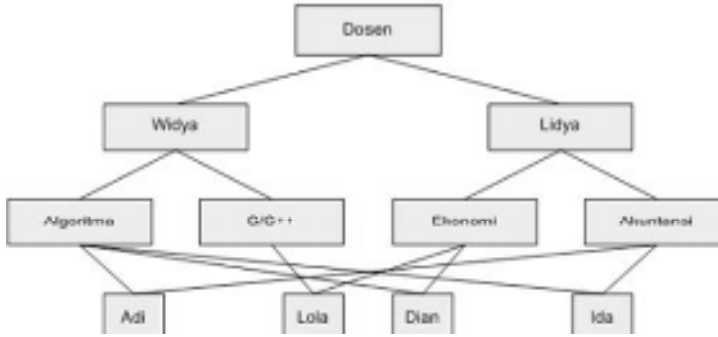


Gambar 4.1. Model Hirarki

3) Model Jaringan

Mirip dengan hirarkical model, dimana data dan hubungan antar data direpresentasikan dengan record dan links. Model data jaringan adalah pengembangan dari model data hirarkis. Dalam model data jaringan, child record diperkenannkan memiliki lebih dari satu parent record. Perbedaannya terletak pada susunan record dan linknya yaitu network model menyusun record-record dalam bentuk graph.





Gambar 4.2. Model data Jaringan

2. Model Data Relational

Model Data Relasional adalah suatu model basis data yang menggunakan tabel dua dimensi, yang terdiri atas baris dan kolom untuk menggambarkan sebuah berkas data. Model ini menunjukkan cara mengelola/mengorganisasikan data secara fisik dalam memory sekunder, yang akan berdampak pula pada bagaimana kita mengelompokkan data dan membentuk keseluruhan data yang terkait dalam sistem yang dibuat. Contoh table dan keterhubungannya:

MHS

NPM	Nama	Alamat
10296832	Nurhayati	Jakarta
10296126	Astuti	Jakarta
31296500	Budi	Depok
41296525	Prananingrum	Bogor
50096487	Pipit	Bekasi
21196353	Quraish	Bogor

MKUL

KDMK	MTKULIAH	SKS
KK021	P. Basis Data	2
KD132	SIM	3
KU122	Pancasila	2

NILAI

NPM	KDMK	MID	FINAL
10296832	KK021	60	75
10296126	KD132	70	90
31296500	KK021	55	40
41296525	KU122	90	80
21196353	KU122	75	75
50095487	KD132	80	0
10296832	KD132	40	30

Keuntungan Model Data Relasional

a. Bentuknya sederhana
b. Mudah melakukan berbagai operasi data (query, update/edit, delete).

3. Istilah dalam Model Relasional

Istilah-istilah yang digunakan dalam model data relasional, diantaranya : relasi, atribut, record, cardinalitas.

a. Relasi

   Merupakan sebuah tabel yang terdiri dari beberapa kolom dan beberapa baris.

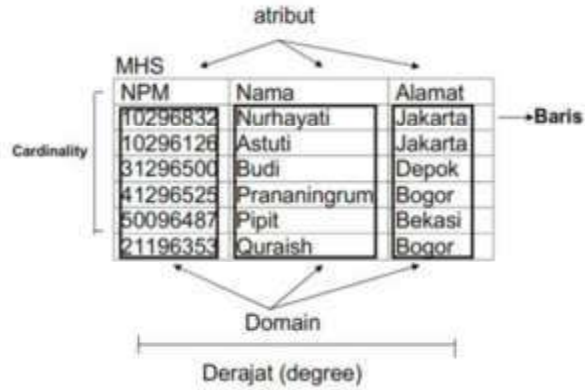
b. Entity atau Entitas

   Merupakan obyek yang mewakili sesuatu dalam dunia nyata dan dapat dibedakan antara satu dengan lainnya (unique)

c. Atribut

   Merupakan karakteristik dari entitas atau relationship, yang menyediakan penjelasan detail tentang entitas atau relationship. Dalam penerapannya (level fisik) atribut merupakan field atau kolom dari sebuah tabel

d. Record
   Merupakan baris pada sebuah relasi
e. Domain
   Merupakan kumpulan nilai yang valid untuk satu atau lebih atribut
f. Derajat (degree)
   Merupakan jumlah atribut dalam sebuah relasi (jumlah field)
g. Cardinality
   Merupakan jumlah tupel dalam sebuah relasi (jumlah record). Model data harus dapat merepresentasikan jumlah peristiwa dari obyek di dalam hubungan yang diberikan.



Gambar 4.3. Ilustrasi Penerapan Istilah

4. Jenis Kunci Relational

Key adalah merupakan suatu atribut yang menandakan kunci dari suatu entitas yang bersifat unik. *Key attribute* adalah satu atau beberapa atribut yang mempunyai nilai unik sehingga dapat digunakan untuk membedakan data pada suatu baris/record dengan baris lain pada suatu entitas. Key attribute dibedakan menjadi tiga yaitu: *Superkey, Candidat Key, Primary* key, dan *Foreign Key.*

a. *Superkey*

Satu atribut/kumpulan atribut yang secara unik mengidentifikasi sebuah tupel di dalam relasi (satu atau lebih field yang dapat dipilih untuk membedakan antara 1 record dengan record lainnya).

Contoh:

Untuk tabel MHS di atas, super key-nya:

- NPM
- NAMA (dengan syarat tidak ada nama yang sama)
- ALAMAT (dengan syarat tidak ada alamat yang sama)
- NPM + NAMA
- NPM + ALAMAT
- NAMA + ALAMAT
- NPM + NAMA + ALAMAT

b. *Candidat Key*

Atribut di dalam relasi yang biasanya mempunyai nilai unik (super key dengan jumlah field yang paling sedikit).

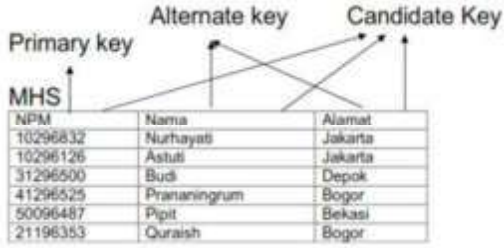
Dari contoh di atas, maka *candidate key*-nya adalah NPM, NAMA dan ALAMAT (karena hanya terdiri dari 1 field saja)

c. *Primary* key.

Candidate key yang dipilih untuk mengidentifikasikan tupel secara unik dalam relasi, maka *primary key* yang dipilih adalah NPM (unik, tidak ada NPM yang sama).

d. Foreign Key

Atribut dengan domain yang sama yang menjadi kunci utama pada sebuah relasi tetapi pada relasi lain atribut tersebut hanya sebagai atribut biasa.



5. Keuntungan Model Data Relational

Model Relasional merupakan model data yang paling banyak digunakan saat ini. Hal ini disebabkan oleh bentuknya yang sederhana dibandingkan dengan model jaringan/network atau model hirarki. Bentuk yang sederhana ini membuat pekerjaan seorang programmer menjadi lebih mudah, yaitu dalam melakukan berbagai operasi data (query, insert, update, delete, dan lainnya).

Kelebihan model data relasional, diantaranya

a. Bentuknya sederhana.
b. Data dapat diakses lebih cepat.
c. Struktur basis data mudah diubah.
d. Data lebih akurat.
e. Memudahkan user untuk membangun dan memodifikasi program aplikasi.
f. Memudahkan user dalam membentuk query yang kompleks untuk memanggil kembali (retrieve) data.

EVALUASI

1. Jelaskan perbedaan antara model basis data relasional, hirarki, dan jaringan!
2. Pada model basis data relasional terdapat beberapa *key attribute,* jelaskan perbedaan masing-masing *key attribute tersebut!*

BAB V
PERANCANGAN BASIS DATA DENGAN ERD

Setelah mempelajari materi ini, diharapkan mahasiswa mampu:
1. Menjelaskan pengertian Entity Relationship Diagram (ERD)
2. Memahami konsep dasar ERD
3. Menjelaskan komponen ERD
4. Membuat rancangan ERD

1 Definisi Entity Relationship Diagram (ERD)

Diagram relasi entitas atau *entity-relationship diagram* (ERD) adalah suatu diagram dalam bentuk gambar atau simbol yang mengidentifikasi tipe dari entitas di dalam suatu sistem yang diuraikan dalam data dengan atributnya, dan menjelaskan hubungan atau relasi diantara entitas tersebut. ERD merupakan model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak. ERD berupa model data konseptual, yang merepresentasikan data dalam suatu organisasi.ERD menekankan pada struktur dan relationship data

2 Memahami konsep dasar ERD

ERD digunakan oleh profesional sistem untuk berkomunikasi dengan pemakai eksekutif tingkat tinggi dalam perusahaan atau organisasi yang tidak tertarik pada pelaksanaan operasi sistem sehari-hari, namun lebih menekankan kepada beberapa hal yaitu :

a. Data apa saja yang diperlukan untuk bisnis mereka?
b. Bagaimana data tersebut berelasi dengan data lainnya?
c. Siapa saja yang diperbolehkan mengakses data tsb?

Untuk menggambarkan ER diagaram setidaknya ada empat langkah yang harus dilakukan oleh perancang basis data yaitu:

a. Menemukan atau mendefinisikan Entitas
b. Menemukan atau mendefinisikan atribute
c. Menemukan atau mendefinisikan Relasi
d. Menggambarkan ERD menggunakan notasi-notasi standar

3 Menjelaskan komponen ERD

Untuk dapat membuat entity relasional diagram, maka komponen yang harus terpenuhi adalah:

a. Obyek Data, Atribut dan Hubungan.

Obyek Data Adalah representasi dari hampir semua informasi gabungan yang harus dipahami oleh perangkat lunak. Objek data dapat berupa entitas eksternal (misalkan semua yang menghasilkan informasi), suatu benda (misal laporan atau tampilan), peristiwa (misalnya proses meminjam) atau event, peran (misalnya peminjam), unit organisasi atau suatu struktur. Sebagai contoh : orang atau mobil dapat dipandang sebagai objek data bila salah satu dari mereka dapat didefinisikan dalam bentuk atribut. Deskripsi objek data menghubungkan objek data dengan semua atributnya. Obyek data dihubungkan satu dengan yang lainnya, misalkan seorang dapat memiliki mobil, dimana hubungan "memiliki" mengkonotasikan suatu hubungan khusus antara seorang dengan mobil.

Atribut Menentukan property suatu obyek data dan mengambil salah satu dari tiga karakteristik yang berbeda. Atribut dapat digunakan untuk:

1) Menamai sebuah contoh dari obyek data
2) Menggambarkan contoh

3) Membuat referensi ke contoh yang lain pada tabel yang lain.

Hubungan Obyek data disambungkan satu dengan lainnya dengan berbagai macam cara. Andaikan ada dua objek data, buku dan toko buku, obyek tersebut dapat diwakilkan dengan menggunakan dua notasi sederhana, dibangun suatu hubungan anatar buku dengan toko buku karena kedua obyek data tersebut berhubungan. Hubungan tersebut dapat berupa :

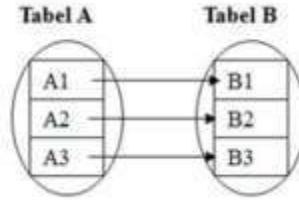
1) Toko buku memesan buku
2) Toko buku menampilkan buku
3) Toko buku menjual buku

Hubungan memesan, menampilkan, menjual mendefinisikan hubungan yang relevan antara buku dan toko buku. Penting untuk dicatat bahwa hubungan obyek mempunyai dua arah, dimana mereka dpaat dibaca dari dua arah, misalnya :toko buku memesan buku atau buku dipesan oleh toko buku.

b. Kardinalitas dan Modalitas Kardinalitas

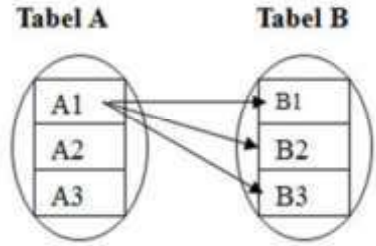
Model data harus dapat merepresentasikan jumlah peristiwa dari obyek di dalam hubungan yang diberikan.

1) Satu ke satu (1:1) Misalnya: seorang suami hanya dapat memiliki satu istri, dan seorang istri hanya mempunyai satu suami. Contoh ilustrasi dapat ditunjukkan pada Gambar 5.1



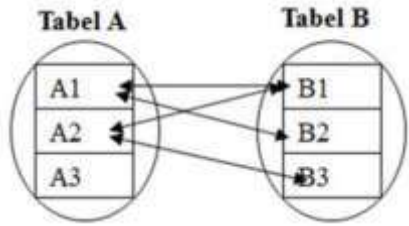
Gambar 5.1 Relasi Satu ke Satu

2) Satu ke banyak (1:N) Misalnya: seorang ibu kandung dapat memiliki banyak anak, tetapi seorang anak hanya dapat memiliki satu ibu kandung. Contoh ilustrasi dapat ditunjukkan pada Gambar 5.2



Gambar 5.1 Relasi Satu ke Banyak

3) Banyak ke banyak (M:N) Misalnya: seorang paman dapat memiliki banyak keponakan, sementara itu seorang keponakan dapat memiliki banyak paman. Contoh ilustrasi dapat ditunjukkan pada Gambar 5.3



Gambar 5.3 Relasi Banyak ke Banyak

Modalitas dari suatu hubungan adalah nol bila tidak ada kebutuhan eksplisit untuk hubungan yang terjadi atau hubungan itu bersifat opsional. Modalitas bernilai satu jika suatu kejadian dari hubungan merupakan perintah.

4 Membuat rancangan ERD

Untuk menggambarkan ERD setidaknya ada tiga langkah yang harus dilakukan oleh perancang basis data yaitu:

a. Menemukan atau mendefinisikan Entitas
b. Menemukan atau mendefinisikan atribute
c. Menemukan atau mendefinisikan Relasi
d. Menentukan kardinalitas

a. Entitas (Entity)

Entitas merupakan obyek yang mewakili sesuatu dalam dunia nyata dan dapat dibedakan antara satu dengan lainnya (unique). Setiap entitas memiliki beberapa atribut yang mendeskripsikan karakteristik dari objek tersebut. Atribut Dapat berupa:

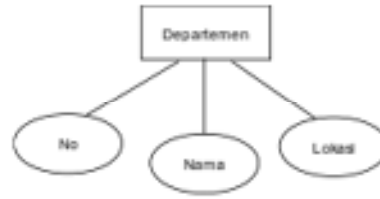
- Fisik (mobil, rumah, manusia, pegawai dsb)
- Abstrak/konsep (department, pekerjaan, mata kuliah dsb)
- Kejadian (pembelian, penjualan, peminjaman, dll)

Entitas dapat dibedakan menjadi dua macam yaitu **Entitas kuat** dan **entitas lemah**. **Entitas lemah** adalah yang keberadaannya tergantung pada entitas lain. Gambar dibawah ini menjelaskan notasi umum entitas kuat dengan nama entitas Anggota dan entitas lemah dengan nama entitas tanggungan. Entitas tanggungan disebut sebagai **entitas lemah** karena jika data seorang pegawai dihapus maka data tanggungannya juga akan terhapus. Keberadaan data tanggungan tergantung pada data di pegawai.

Gambar 5.4 Contoh Notasi

Contoh :

Entitas	Atribut
Petugas	NIP, Nama, Alamat, Agama, Jenis kelamin
Departemen	No, Nama, Lokasi



Gambar 5.5 Contoh Penggunaan Simbol Entitas dan Atribut

Urutan langkah untuk menemukan atau mendefinisikan entitas dalam suatu sistem adalah:

a. Buat ilustrasi/gambaran cerita tentang sistem yang akan dicari entitasnya
b. Tandai setiap objek yang diwakili oleh kata benda yang ada di dalam ilustrasi tersebut
c. Untuk setiap objek tersebut yakinkan bahwa ia memiliki karakteristik yang nanti disebut sebagai atribut
d. Tentukan objek yang merupakan entitas (Jika memang ia memiliki karakteristik jadikan ia sebagai entitas)
e. Menggambarkan entitas beserta atributnya menggunakan notasi simbol yang telah ditentukan.

Contoh cara menentukan entitas:

Langkah 1. *Deskripsi tentang gambaran sistem (misalnya gambaran tentang system informasi perpustakaan):*

Departemen A mempunyai perpustakaan, perpustakaan ini dijaga oleh dua orang pegawai. Anggota perpustakaan ini adalah seluruh pegawai departemen yang sudah terdaftar menjadi anggota perpustakaan. Koleksi buku yang dimiliki ada sekitar 200 eksemplar, dengan berbagai jenis buku, pengarang dan penerbit. Untuk melakukan peminjaman buku, anggota melakukan proses peminjaman ke petugas dengan memberikan kartu anggota perpustakaan, lama peminjaman adalah dua minggu, proses pengembalian dilakukan dengan memberikan buku dan kartu anggota untuk pengecekan. Jika anggota terlambat mengembalikan maka akan dikenakan denda Rp.1000, 00 per hari.

Langkah 2. *Tandai setiap objek yang diwakili oleh kata benda yang ada di dalam ilustrasi tersebut.*

Departemen A mempunyai perpustakaan, perpustakaan ini dijaga oleh dua orang **pegawai. Anggota** perpustakaan ini adalah seluruh pegawai departemen yang sudah terdaftar menjadi anggota perpustakaan. Koleksi buku yang dimiliki ada sekitar 200 eksemplar, dengan berbagai **jenis buku, pengarang** dan **penerbit**. Untuk melakukan peminjaman buku, anggota melakukan proses **peminjaman** ke petugas dengan memberikan kartu anggota perpustakaan, lama peminjaman adalah dua minggu, proses **pengembalian** dilakukan dengan memberikan buku dan kartu anggota untuk pengecekan. Jika anggota terlambat mengembalikan maka akan dikenakan **denda** Rp.1000, 00 per hari.

Langkah 3. *Untuk setiap objek tersebut yakinkan bahwa ia memiliki karakteristik yang nanti disebut sebagai atribut*

Pegawai : id, nama, Username, Password
Anggota : No_anggota, Nama, alamat, Nomor_telp, jenis_kelamin
Buku : Kode, kategori,Judul,Jumlah_halaman, ISBN, Pengarang, penerbit
Jenis buku : Kode, Jenis
Pengarang : Kode, Nama
Penerbit : Kode, Nama
Peminjaman : Kode_peminjaman, Id_Petugas, No_anggota, Kode_Buku, Tgl_pinjam, Tgl_kembali
Pengembalian : Kode_pengembalian, No_anggota, tgl_kembali, denda

b. Atribut

Atribut adalah sifat-sifat atau karakteristik pada suatu entitas. Nama atribut ini identik dengan nama kolom atau field pada suatu tabeldalam basis data.Candidat Key adalah merupakan superkey yang jumlah atributnya paling sedikit.
Misalnya *candidat key* untuk entitas petugas antara lain:

1) Id_Pegawai
2) Nama (jika dapat dijamin kalau tidak ada nama yang sama antara satu baris dengan baris yang lain)

Primary key adalah suatu *candidat key* yang dipilih menjadi kunci utama karena sering dijadikan acuan untuk mencari informasi, ringkas, menjadi keunikan suatu baris. MisalnyakodeBuku antara buku yang satu dengan buku yang lain pasti berbeda, dalam hal ini kodeBuku dapat digunakan sebagai

suatu key. Gambar 5.6 menjelaskan simbol atau notasi primary key.

Gambar 5.6 Primary key

c. Relasi (Relationship)

Relasi menyatakan hubungan antara dua atau beberapa entitas.Setiap relasi mempunyai batasan (constraint) terhadap kemungkinan kombinasi entitas yang berpartisipasi. Batasan tersebut ditentukan dari situasi yang diwakili relasi tersebut. Ragam atau jenis relasi dibedakan menjadi beberapa macam antara lain adalah.

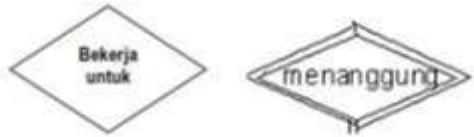
a) Relasi Binary. Relasi ini merupakan relasi antara 2 entitas. Relasi ini dibedakan menjadi :
   - Relasi One-to-one (notasi 1:1)
   - Relasi One-to-many (notasi 1:N) atau many-to-one (notasi N:1)
   - Relasi Many-to-many (notasi M:N)

b) Relasi Ternary. Relasi ini merupakan relasi antara 3 entitas atau lebih.

   Dalam Relasi One-to-one (1:1) setiap atribute dari satu entitas berpasangan dengan satu attribute dari entitas yang direlasikan.Dalam relasi One-to-many (1:N) atau many-to- one (N:1) satu atribute berelasi dengan beberapa atribute dari entitas yang direlasikan.Dalam Many-to-many (M:N) satu atribute berelasi dengan beberapa atribute dari entitas yang direlasikan, begitu pula sebaliknya.

Notasi Relasi

Sebagaimana entias dalam relasi juga dapat dibedakan menjadi relasi kuat dan relasi lemah. gambar dibawah ini menjelaskan notasi umum untuk relasi kuat dan relasi lemah.



Gambar 5.5 Relasi kuat dan relasi lemah

Menemukan Relasi

Beberapa langkah yang dapat dilakukan untuk menemukan atau mengidentifikasi relasi yaitu antara lain sebagai berikut:

1) Dari gambaran cerita sistem, tandai setiap hubungan yang diwakili oleh kata kerja yang ada di dalam ilustrasi tersebut beserta entitas yang berhubungan
2) Identifikasikan rasio kardinalitas dari setiap hubungan
3) Identifikasikan batasan partisipasi dari setiap hubungan yang ada berikut kemungkinan atribut yang muncul dari setiap hubungan

Gambarkan hubungan tersebut dalam bentuk notasi diagram dan gabungkan dengan notasi Entitas dan atribut yang dibuat sebelumnya. Sebagai contoh tentukan relasi untuk sistem informasi perpustakaan dengan melihat deskripsi sistem di atas.
Langkah-langkah penyelesaian adalah :

Langkah 1:

Dari gambaran cerita sistem, tandai dan tentukan setiap hubungan yang diwakili oleh kata kerja yang ada di dalam

ilustrasi dan entitas yang berhubungan. Identifikasi hubungan antara entitas

Tabel Entitas dan Relasi

Entitas 1	Hubungan	Entitras 2
Anggota	Melakukan	Peminjaman
Pegawai	Mengelola	Peminjaman
Pengarang	Menulis	Buku
Penerbit	Menerbitkan	Buku
Anggota	Melakukan	Pengembalian
Pegawai	Mengelola	Pengembalian

Dari tabel di atas maka berikut relasinya :

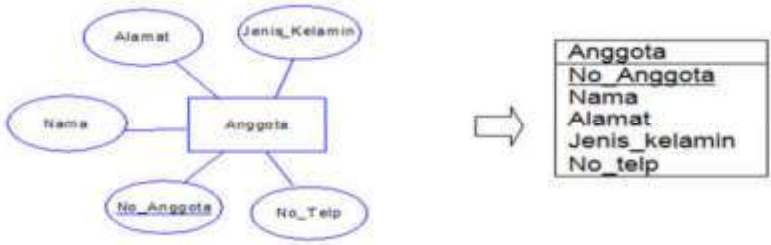
Gambar 5.6 Notasi hubungan entitas dan relasi

Langkah 2.

Mapping ER diagram ke tabel. Didalam data base yang menjadi pusat perhatian dan intisari dari sistem database itu adalah table dan relasinya. Tabel ini sama artinya dengan entitas pada model data pada level konseptual. Setiap orang bisa membuat table tetapi membuat table yang baik tidak semua orang dapat melakukannya. Kebutuhan akan membuat tabel yang baik ini

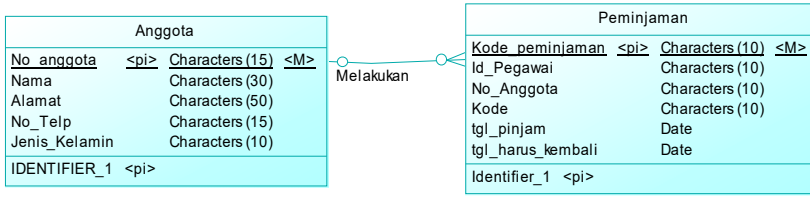
maka muncul teori beberapa teori atau metode yaitu mapping ERD to table.

Contoh:
Dari relasi yang telah dibuat pada sistem informasi perpustakaan, pada setiap entitasnya pilih satu atribut kunci sebagai primary key (atribut harus uniqe).



Gambar 5.7. Maping Notasi ke Diagram ER

Dengan cara yang sama dapat dilakukan mapping ERD to table pada semua entitas. Setelah semua entitas selesai dibuat, tentukan relasi antar entitas sesuai dengan tabel hubungan antar entitas di atas. Misalnya relasi antara entitas anggota dan Peminjamanya adalah melakukan



Gambar 5.8 Entity Relationship Diagram

EVALUASI

1. Jelaskan secara singkat tentang ERD yang kalian ketahui!
2. Tugas identifikasi entitas dan atribut
   Lakukan observasi ke perusahaan yang ada di sekitar anda, terkait kebutuhan sistem basis data, susun narasi gambaran sistem dari perusahaan tersebut serta identifikasi terkait :
   a. Entitas
   b. Atribut
   c. Relasi antar entitas
   d. Gambaran ERD

Bab VI
KONSEP NORMALISASI

Setelah mempelajari materi ini, diharapkan mahasiswa mampu:
1. Memahami konsep normalisasi
2. Menjelaskan aturan normalisasi
3. Menerapkan proses normalisasi
4. Merancang bentuk normal pertama (1NF)
5. Merancang bentuk normal tahap kedua (2NF)
6. Merancang bentuk normal tahap ketiga (3NF)

1. Konsep Normalisasi

Normalisasi diartikan sebagai suatu teknik yang menstrukturkan atau mendekomposisi atau memecah data menggunakan cara–cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan–penyimpangan (*anomalies*) yang terjadi akibat adanya kerangkapan data dalam relasi dan inefisiensi pengolahan. Anomali merupakan proses pada basis data yang memberikan efek samping yang tidak diharapkan (misalnya terjadinya redudansi). Bila ada anomali maka relasi mungkin perlu dipecah menjadi beberapa tabel lagi agar diperoleh database yang optimal.

Proses normalisasi dilakukan jika terjadi dependensi (ketergantungan). Dependensi menjelaskan nilai sebuah atribut yang menentukan nilai atribut lainnya. Terdapat beberapa jenis dependensi, diantaranya dependensi fungsional dan dependensi transitif.

a. Dependensi Fungsional, suatu atribut Y mempunyai dependensi fungsional terhadap atribut X jika dan hanya jika setiap nilai nilai X berhubungan dengan sebuah nilai Y ($X \rightarrow Y$).

b. Dependensi Transitif, atribut Z mempunyai dependensi transitif terhadap X jika atribut Y memiliki dependensi fungsional terhadap X dan jika atribut Y memliki dependesi fungsional terhadap Y.

Proses normalisasi akan menghasilkan relasi yang optimal, yaitu :

a. Memiliki struktur *record* yang mudah untuk dimengerti.
b. Memiliki struktur *record* yang sederhana dalam pemeliharaan.
c. Memiliki struktur *record* yang mudah untuk ditampilkan kembali untuk memenuhi kebutuhan pemakai.
d. Minimalisasi kerangkapan data guna meningkatkan kinerja system

Beberapa konsep yang harus dipahami sebelum mengimplementasikan teknik normalisasi data antara lain ialah: 1) ketergantungan fungsional. 2) Domain dan tipe data. 3) Konsep key atribut (Field/atribute kunci).

2. Aturan Normalisasi

Sebuah basis data dapat dikatakan baik, jika setiap tabel yang menjadi unsur pembentuk basis data tersebut juga telah berada dalam keadaan baik atau normal. Selanjutnya, sebuah tabel dapat dikategorikan baik (*efisien*) atau normal, jika telah memenuhi 3 (tiga) kriteria berikut :

a. Jika ada *dekomposisi* (penguraian) tabel, maka dekomposisinya harus dijamin aman (*Lossless–Join Decomposition*).
b. Terpeliharanya ketergantungan fungsional pada saat perubahan data (*DependencyPreservation*).
c. Tidak melanggar *Boyce–Code Normal Form* (BCNF)

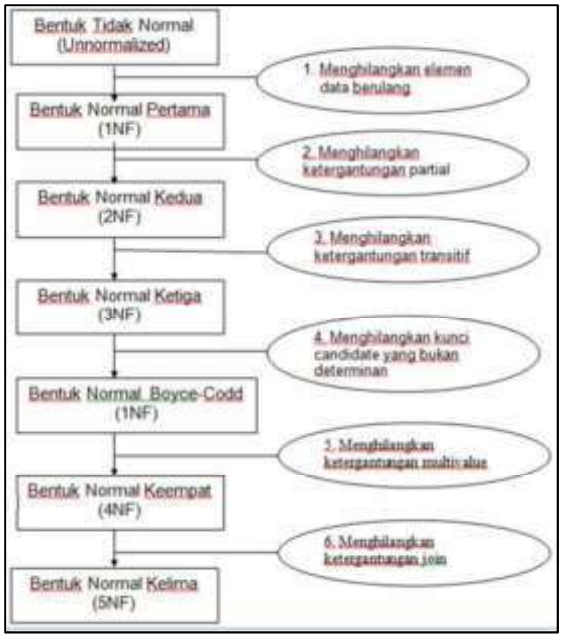
3. Proses Normalisasi

Hasil dari proses normalisasi adalah tabel–tabel data dalam bentuk normal (*normal form*), yaitu tabel–tabel data yang terhindar dari dua hal yaitu: Pengulangan informasi dan Potensi *inkonsistensi* data pada operasi pengubahan. Terdapat enam bentuk normal (normal form) dalam teknik normalisasi data, keenam bentuk tersebut adalah :

a. Bentuk Normal Tahap pertama (1^{st} NF)
b. Bentuk Normal Tahap Kedua (2^{nd} NF)
c. Bentuk Normal Tahap Ketiga (3^{rd} NF)
d. Bentuk Normal Boyce - Code (BCNF)
e. Bentuk Normal Tahap Keempat (4^{rd} NF)
f. Bentuk Normal Tahap Kelima (5^{rd} NF)

Dalam proses normalisasi, data diuraikan dalam bentuk tabel, selanjutnya dianalisis berdasarkan persyaratan tertentu ke beberapa tingkat. Apabila tabel yang diuji belum memenuhi persyaratan tertentu, maka tabel tersebut perlu dipecah menjadi beberapa tabel yang lebih sederhana sampai memenuhi bentuk yang optimal.

Langkah yang dilakukan untuk melakukan normalisasi ditunjukkan pada Gambar 6.1



Gambar 6.1 Langkah Proses Normalisasi Data

4. Merancang bentuk normal pertama (1NF)

Bentuk normal ke satu (1NF) dilakukan penghapusan beberap grup elemen yang berulang agar tidak terjadi redudansi atau agar menjadi satu nilai tunggal yang berinteraksi di antara setiap baris pada tabel. Pada 1NF Setiap data disajikan dalam bentuk flat file (tabular/ tabel), seluruh atribut kunci terdefinisikan, tidak ada pengulangan group pada table, dan semua atribut bergantung pada Primary Key.

Bentuk normal ke satu 1 NF ini mempunyai beberapa ciri antara lain yaitu:

a. Setiap data dibentuk dalam flat file (file data/ rata)

b. Data dibentuk dalam satu record demi satu record dan nilai dari field field berupa "atomic value", tidak dapat dibagi-bagi lagi.
c. Tidak ada set atribute yang berulang ulang atau atribute bernilai ganda (multivalue).
d. Tidak ada set atribut *composite* atau kombinasinya dalam domain data yang sama.
e. Tiap field hanya satu pengertian, bukan merupakan kumpulan kata yang mempunyai arti mendua, hanya satu arti saja dan juga bukanlah pecahan kata sehingga artinya lain.

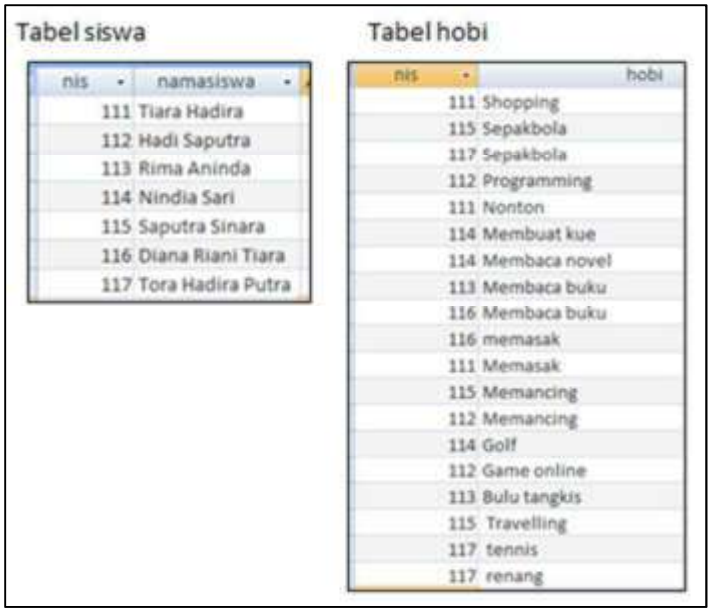
Contoh table yang belum memenuhi bentuk 1NF:

nis	namasiswa	hobi
111	Tiara Hadira	Memasak, Nonton, shopping
112	Hadi Saputra	Programming, game online, Memancing
113	Rima Aninda	Membaca buku, bulu tangkis
114	Nindia Sari	Membuat kue, membaca novel, golf
115	Saputra Sinara	Memancing, travelling, sepakbola
116	Diana Riani Tiara	Membaca buku, memasak, bowling
117	Tora Hadira Putra	Sepakbola, tennis, renang

data pada tabel yang belum memenuhi 1NF, bentuk lain dapat dijabarkan seperti berikut:

nis	namasiswa	hobi1	hobi2	hobi3
111	Tiara Hadira	Memasak	Nonton	shopping
112	Hadi Saputra	Programming komputer	game online	Memancing
113	Rima Aninda	Membaca buku	bulu tangkis	
114	Nindia Sari	Membuat kue	membaca novel	golf
115	Saputra Sinara	Memancing	travelling	sepakbola
116	Diana Riani Tiara	Membaca buku	memasak	bowling
117	Tora Hadira Putra	Sepakbola	tennis	renang

Untuk dapat memenuhi aturan bentuk 1NF, dilakukan dekomposisi menjadi 2 entitas, yakni tabel siswa dan tabel hobi seperti berikut :



Bentuk Normal Pertama (1NF) :

a. Setiap data disajikan dalam bentuk flat file (tabular/ tabel)
b. Seluruh atribut kunci terdefinisikan
c. Tidak ada pengulangan group pada tabel
d. Semua atribut bergantung pada Primary Key

5. Merancang bentuk normal tahap kedua (2NF)

Pada bentuk normal ke dua (2NF) syarat yang harus dipenuhi adalah:

a. Bentuk data telah memenuhi kriteria bentuk normal kesatu.
b. Atribute bukan kunci haruslah bergantung secara fungsi pada kunci utama atau primary key.
c. Sudah ditentukan kunci kunci field, dimana kunci field haruslah unik dan dapat mewakili atribute lain yang menjadi anggotanya

Sebagai contoh ditentukan sebuah tabel siswa sebagai berikut :

NIS	Nama_siswa	Alamat	Kode_ Mapel	Nama_Mapel	Nama_Guru	Nilai

Tabel di atas telah memenuhi 1NF, namun belum memenuhi 2NF. Tabel di atas perlu didekomposisi menjadi beberapa tabel untuk memenuhi syarat 2NF. sebagai berikut :

Tabel Nilai : (NIS, Kode_mapel, Nilai)

Tabel Siswa : (NIS, Nama_siswa, Alamat)

Tabel Mapel : (Kode_mapel, Nama_mapel, Nama_Guru)

6. Merancang bentuk normal tahap ketiga (3NF)

Untuk menjadi bentuk normal ketiga (3 NF) suatu tabel harus mempunyai ciri-ciri sebagai berikut:

a. Memenuhi bentuk 2 NF (normal kedua)
b. Atribut bukan kunci tidak memiliki dependensi transitif terhadap kunci utama atau primary key.
c. Setiap atribute bukan kunci haruslah bergantung hanya pada primary key dan pada primary key secara menyeluruh

Berikut ini adalah contoh relasi yang telah memenuhi bentuk 2 NF, tetapi belum memenuhi bentuk 3 NF :

NIS	Nama_siswa	Alamat_jln	Alamat_kota	Alamat_prov	Kodepos

Pada relasi di atas, masih terdapat atribut non primary key (yakni Alamat_kota dan Alamat_Prov) yang memiliki ketergantungan terhadap atribut non primary key yang lain, yaitu Kode_pos.

Kodepos : {Alamat_kota, Alamat_prov}

Untuk memenuhi syarat 3NF, maka relasi tersebut harus didekomposisi sebagai berikut :

Siswa : (NIS, Nama_siswa, Alamat_jn, Kodepos)

Kodepos : (Kodepos, Alamat_kota, Alamat_prov)

7. Bentuk Normal Boyce - Code (BCNF)

BCNF merupakan kasus khusus 3NF, table atau relasi table berada dalam bentuk BCNF jika:

a. Setiap penentu (determinan) pada tabel adalah sebuah kunci kandidat (candidate key)
b. Jika tabel hanya mengandung satu kunci kandidat maka bentuk 3NF sama dengan BCNF.

8. Bentuk Normal Tahap Keempat (4rd NF)

Tabel pada basis data berada dalam bentuk 4NF jika dan hanya jika telah berada dalam bentuk 3NF dan setiap sebuah ketergantungan multi nilai yang rumit X→→Y, dimana X merupakan superkey yang berarti X merupakan candidate key.

9. Bentuk Normal Tahap Kelima (5rd NF)

Table pada basis data berada dalam bentuk 5NF jika, telah berada dalam bentuk 4NF dan setiap ketergantungan join berhubungan dengan candidate key secara tidak langsung.

EVALUASI

Perhatikan table di bawah ini :

NO_PROY	NAMA_PROY	NRP	NAMA_PEG	KEAHLIAN	UPAH_HARI	HARI_KERJA
12	Train A	105	Sambudi	Mandor	50000	25
		107	Tantowi	Tkg. Las	40000	24
		108	Subardi	Tkg. Las	40000	25
		110	Ferry	Tkg. Listrik	40000	20
		115	Triyadi	Buruh kasar	25000	21
		116	Wagino	Buruh kasar	25000	24
16	Train C	102	Sucono	Mandor	50000	25
		103	Reynaldi	Tkg. Las	40000	22
		109	Rachmat	Tkg. Las	40000	25
		112	Syamsul	Tkg. Listrik	40000	22
		114	Ngadiyo	Buruh kasar	25000	25
		120	Tugino	Buruh kasar	25000	25
		122	Wawan	Buruh kasar	25000	25

Buatlah bentuk 1NF, 2NF, dan 3 NF (jika memungkinkan) dari table tersebut dan buat relasinya.

Bab VII
PERINTAH SQL: Data Definition Language (DDL)

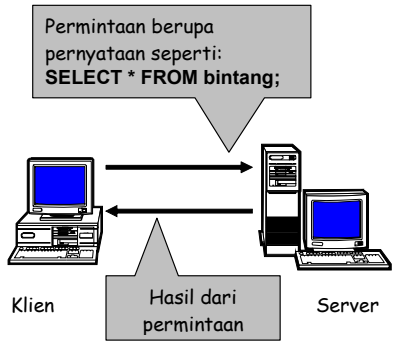
Structure Query Language (SQL) telah menjadi bagian dari arsitektur aplikasi. SQL DDL memungkinkan objek basis data, seperti skema, domain, table, view, dan index untuk dibuatkan dan dihapuskan.

Setelah mempelajari materi ini, diharapkan mahasiswa mampu:

1. Menjelaskan definisi SQL
2. Menjelaskan type data SQL
3. Menerapkan pembuatan basis data
4. Menerapkan pembuatan table dengan query
5. Menerapkan relasi antar table dengan query
6. Memodifikasi table dengan query

1 Menjelaskan definisi SQL

SQL *(Structured Query Language)* adalah sebuah bahasa yang digunakan untuk mengakses data dalam software DBMS. Bahasa ini merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data mendukung bahasa ini untuk melakukan pengelolaan datanya. Untuk melakukan pengaksesan data, digunakan MySQL. MySQL merupakan basis data server yang bersifat open source, multiplatform, dan berbasis relasional. Ilustrasi penggunaan SQL ditunjukkan pada Gambar 7.1



Gambar 7.1 Ilustrasi proses akses data di SQL

Instruksi – instruksi atau pernyataan SQL dapat dikelompokkan menjadi DDL (data definition language) dan DML (data manipulation language).

a. DDL (*Data Definition Language*)

DDL adalah sebuah metode Query SQL yang berguna untuk mendefinisikan struktur data pada sebuah basis data, Query yang dimiliki DDL adalah :

CREATE	:	Membuat Database dan Tabel
Drop	:	Menghapus Tabel dan Database
Alter	:	Melakukan perubahan struktur tabel yang telah dibuat, baik menambah Field (Add), mengganti nama Field (Change) ataupun menamakannya kembali (Rename)

2 Menjelaskan type data SQL

Tipe data adalah suatu bentuk pemodelan data yang

dideklarasikan pada saat melakukan pembuatan tabel. Tipe data ini akan mempengaruhi setiap data yang akan dimasukkan ke dalam sebuah tabel. Data yang akan dimasukkan harus sesuai dengan tipe data yang dideklarasikan. Berbagai type data dapat dilihat pada Tabel berikut.

Tabel 7.1. Type Data untuk Bilangan (Number)

Type Data	Keterangan
TINYINT	Ukuran 1 byte. Bilangan bulat terkecil, dengan jangkauan untuk bilangan bertanda: -128 sampai dengan 127 dan untuk yang tidak bertanda : 0 s/d 255.
SMALLINT	Ukuran 2 Byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -32768 s/d 32767 dan untuk yang tidak bertanda : 0 s/d 65535
MEDIUMINT	Ukuran 3 byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -8388608 s/ d 8388607 dan untuk yang tidak bertanda : 0 s/d 16777215
INT	Ukuran 4 byte. Bilangan bulat dengan jangkauan untuk bilangan bertanda : -2147483648 s/d 2147483647 dan untuk yang tidak bertanda : 0 s/d 4294967295
BIGINT	Ukuran 8 byte. Bilangan bulat terbesar dengan jangkauan untuk bilangan bertanda : -9223372036854775808 s/d 9223372036854775807 dan untuk yang tidak bertanda : 0 s/d 18446744073709551615
FLOAT	Ukuran 4 byte. Bilangan pecahan
DOUBLE	Ukuran 8 byte. Bilangan pecahan
REAL	Ukuran 8 byte. Sinonim dari DOUBLE

DECIMAL (M,D)	Ukuran M byte. Bilangan pecahan, misalnya DECIMAL(5,2 dapat digunakan untuk menyimpan bilangan -99,99 s/d 99,99

Table 7.2. Type Data untuk Waktu

Type Data	Keterangan
DATETIME	Ukuran 8 byte. Kombinasi tanggal dan jam, dengan jangkauan dari '1000-01-01 00:00:00' s/d '9999-12-31 23:59:59'
DATE	Ukuran 3 Byte. Tanggal dengan jangkauan dari '1000-01-01' s/d '9999-12-31'
TIMESTAMP	Ukuran 4 byte. Kombinasi tanggal dan jam, dengan jangkauan dari '1970-01-01 00:00:00' s/d '2037'
TIME	Ukuran 3 byte. Waktu dengan jangkauan dari '839:59:59' s/d '838:59:59'
YEAR	Ukuran 1 byte. Data tahun antara 1901 s/d 2155

Table 7.3. Type Data untuk Karakter dan Lain-lain

Type Data	Keterangan
CHAR	Mampu menangani data hingga 255 karakter.
VARCHAR	Mampu menangani data hingga 255 karakter. Tipe data VARCHAR tidak mengharuskan untuk memasukkan data yang telah ditentukan oleh kita.
TINYBLOB, TINYTEXT	Ukuran 255 byte. Mampu menangani data sampai 2^8-1 data.
ENUM('nilai1','nilai2',…,'nilaiN')	Ukuran 1 atau 2 byte. Tergantung jumlah nilai enumerasinya (maksimum 65535 nilai)

3 Menciptakan Basis Data

Database adalah sebuah media utama yang harus dibuat dalam membangun sebuah basis data agar nantinya dapat kita letakkan beberapa tabel dengan field-fieldnya. MySQL adalah database berbasis relasional, struktur data diatur melalui pembuatan table-table yang saling berkaitan (mempunyai relasi).

Tiga elemen yang merupakan model fundamental dari relasi adalah:

c. Struktur Data (Table)

Terdiri dari baris (*row* atau *record*) dan setiap baris terdiri dari kolom-kolom (*column* atau *field*) yang terdefinisi melalui tipe data pada kolom tersebut.

d. Integritas Data

Isi data sesuai kondisi sebenarnya, misalkan field "tinggi_badan" tidak boleh negative, "jenis_kelamin" hanya mempunya nilai 'L' dan 'P'. Kesesuaian data dengan nilai sebenarnya ini disebut juga "batasan nilai untuk integritas data" atau "integrity constraints".

e. Manipulasi Data

Data yang tersimpan dapat dimanipulasi dengan bahasa query seperti SQL.

Perintah yang digunakan untuk menciptakan database pada MySQL dengan Syntax berikut :

create database nama_database;

Contoh :

```
1 create database Perpus;
```

```
create database Perpus;
/* Affected rows: 1  Baris ditemukan: 0  Peringatan: 0  Durasi untuk 1 query: 0,015 sec. */
```

Pada contoh diatas, query OK menyatakan bahwa pembuatan database dengan nama perpus berhasil dibuat.

4 Membuat Desain Tabel dengan Query

Suatu file database (*.mdb, *.accdb) terdiri dari satu atau lebih table, index dan komponen lain. Sedangkan dalam satu tabel bisa terdiri dari satu atau lebih record data masing-masing berisi informasi yang sejenis. Dalam tabel terdapat baris dan kolom. Baris diistilahkan dengan recordset dan kolom dengan field.

f. Membuat Tabel

Tabel merupakan komponen utama pembentuk database (basis data). Tabel terletak pada sebuah database, sehingga pembuatan tabel dilakukan setelah sebuah database telah dibuat. Dalam tabel terdapat bari dan kolom. Seperti yang telah dijelaskan pada pembehasan sebelumnya baris diistilahkan dengan recordset dan kolom dengan field. Ilustrasi ditunjukkan pada Gambar 7.2

Gambar 7.2 Ilustrasi baris dan kolom

Perintah yang digunakan untuk menciptakan tabel dengan syntax query berikut :

```
CREATE TABLE nama_tabel
(
      field1   tipe-data(length),
      field2   tipe-data(length),
      …
      fieldn   tipe-data(length)
)
```

Keterangan :

Komponen	: Keterangan
Tabel	: Nama dari tabel yang akan dibuat.
Field1, Field2	: Nama dari masing-masing field yang akan digunakan pada tabel yang baru dibuat. Anda harus membuat minimal satu field.
Tipe_data	: Tipe data dari field yang digunakan pada tabel baru.
length	: Ukuran dari field dalam karakter. Digunakan hanya untuk tipe data Text.

Contoh :

Membuat table dengan nama Tabel petugas

Field : Id_petugas, Nama, Username, dan password.

Maka Query yang dibuat untuk membentuk Tabel petugas adalah sebagai berikut:

```
create table petugas (
Id_petugas varchar(20),
Nama varchar(50),
Username varchar(15),
Password varchar (15)
);
```

Beberapa elemen umum yang harus ditentukan dalam membuat sebuah tabel:

a. Nama dari tabel harus Unique untuk setiap file database, tidak diperkenankan dalam satu folder terdapat lebih dari satu nama file database yang sama.
b. Nama dari field (kolom) harus bersifat Unique untuk setiap tabel (tidak boleh sama).
c. Tipe data dan ukuran masing-masing field (kolom) harus disesuaikan dengan kondisi data yang akan disimpan.
d. Pemakaian Constraint yang diikutkan dalam pembentukan suatu tabel, terdiri dari Null, Not Null, Primary Key, Unique dan Foreign Key atau gabungan beberapa Constraint yang ada.

Primary key atau ***unique key*** adalah suatu nilai di basis data yang digunakan untuk mengidentifikasi keunikan baris-baris di dalam table. Dalam membuat sebuah database, kita akan menemukan sebuah record yang data nya tidak boleh sama dengan record yang lain. Agar data tidak terjadi redudansi data maka harus membuat sebuah kolom yang di deklarasikan sebagai kunci primer (*primary key*).

Syntax untuk menciptakan *primary key* ada beberapa cara, yaitu :

```
CREATE TABLE nama_tabel
(
     field1  tipe-data(length)PRIMARY KEY,
     field2  tipe-data(length),
     …
     fieldn  tipe-data(length)
)
```

Contoh :

Buat table Anggota dengan menggunakan query dari Table 4

Tabel 7.4. Tabel Anggota

Field	Type	Length	Keterangan
NIS	Varchar	15	Primary key
Nama	Varchar	50	
Prodi	Varchar	15	
Jk	Varchar	10	
alamat	Varchar	50	

Maka query yang digunakan untuk membentuk table anggota adalah :

```
create table anggota (
NIS varchar(15) Primary key ,
Nama varchar(50),
prodi varchar(15),
Jk varchar(10),
alamat varchar(50)
);
```

Jika table berhasi dibuat maka akan muncul pesan seperti script berikut:

```
create table anggota (  NIS varchar(15) Primary key ,  Nama varchar(50),  prodi varchar(15),  Jk varchar(10),  alamat varchar(50)  );
/* Affected rows: 0  Baris ditemukan: 0  Peringatan: 0  Durasi untuk 1 query: 0,114 sec. */
```

Altrernatif lain penulisan Constraint dengan ***Primary Key*** NIS dimana constraintnya diletakkan pada bagian terakhir:

```
Create Table anggota (
NIS Varchar (15),
Nama Varchar (50),
Prodi Varchar (15),
Jk Varchar (10),
Alamat Varchar (50),
Constraint PK_Anggota Primary Key (NIS) );
```

jika ingin menambahkan ***Primary Key*** pada tabel yang sudah terbentuk tetapi belum mempunyai Primary Key, format penulisannya seperti berikut:

```
Alter Table anggota
 Add Constraint PK_Anggota Primary Key (NIS)
```

g. Menghapus Tabel

Menghapus data adalah menghilangkan satu atau beberapa record data dari suatu tabel. Perintah query yang digunakan untuk menghapus adalah Delete, hanya dapat digunakan untuk menghapus record (baris) dan idak dapat digunakan untuk menghapus field (kolom). Untuk menentukan record yang akan dihapus dapat dilakukan perintah "Where". Jika tidak menggunakan perintah ini maka seluruh record yang ada pada tabel yang bersangkutan akan terhapus semua.

Untuk menghapus Tabel yang telah dibuat dapat menggunakan query berikut :

DROP TABLE nama_tabel;

Contoh :

Perintah untuk menghapus table petugas : **Drop table petugas;**

h. Relasi Antar Tabel dengan Query

Relasi tabel adalah hubungan sebuah tabel dengan tabel lainnya. Sehingga tabel tidak lagi berdiri sendiri, melainkan dapat dihubungkan antara satu dengan yang lainnya dan menjadi satu kesatuan. Terdapat dua buah kolom yang diperlukan untuk menghubungkan sebuah tabel dengan tabel lainnya.

1) Kolom yang pertama, yaitu kolom **primary key** (kunci utama) pada tabel yang satu.

2) Kolom yang kedua adalah **foreign key** (kunci asing) pada tabel lainnya.

Foreign Key adalah kolom atau field pada suatu tabel yang berfungsi sebagai kunci tamu dari tabel lain. **Foreign Key** sangat berguna bila kita bekerja dengan banyak tabel yang saling berelasi satu sama lain. Contoh dari Foreign key adalah sebagai berikut.

Setelah membuat sebuah hubungan kita wajib menentukan aturan yang dikenakan padanya. Aturan dasar yang telah baku adalah :

1) **Field yang dihubungkan** dari tabel utama **haruslah** bersesuaian berupa sebuah **Primary Key** dan **Foreign Key**.
2) Kedua field yang saling terhubung harus memiliki **jenis data yang sama**.

Sebuah tabel hanya boleh memiliki satu buah primary key (kunci utama). Namun, sebuah tabel boleh memiliki lebih dari satu buah foreign key (kunci asing). Oleh karena itu, pilihlah satu buah kolom pada tabel yang akan dijadikan primary key yang dapat mewakili kolom lainnya dan nilainya pun unik, misalnya kolom NIM, kode_buku, dan lainnya.

Sintaks relasi antar tabel dengan query:

```
CREATE TABLE nama_tabel (
  field-1 type(length) PRIMARY KEY,
  field-2 type(length),➔ kolom yang akan direlasikan
  field-3 type(length),
  …… ….(….)
  foreign key (field-2) references table_yg_direlasikan
  (primary_key_pd_table_yg_direlasikan)  );
```

Contoh :

Database Perpustakaan terdiri dari 3 tabel : table kategori, table penerbit, dan table buku

Tabel Kategori

Field	Type	Length	Keterangan
Kode_kategori	Varchar	20	Primary key
Nama_kategori	Varchar	20	

Tabel Penerbit

Field	Type	Length	Keterangan
kode_penerbit	Varchar	20	Primary key
Nama_penerbit	Varchar	25	
Alamat_penerbit	Varchar	25	

Tabel Buku

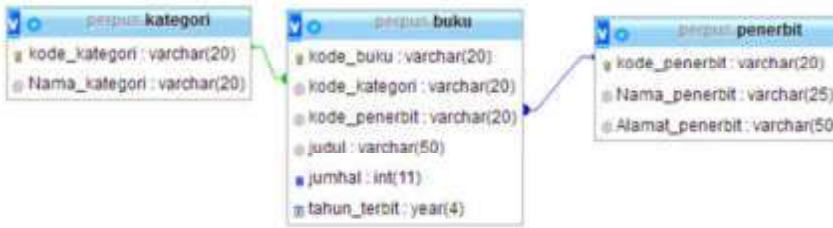
Field	Type	Length	Keterangan
kode_buku	Varchar	20	Primary key
kode_kategori	**varchar**	20	Foreign key
kode_penerbit	**varchar**	20	Foreign key
judul	**varchar**	50	
jumhal	**Int**		
tahun_terbit	**year**		

Pembuatan **table** dan **relasi tabel** dengan query, langkah awal yang harus dilakukan adalah dengan menentukan kolom yang uniq (*primary key*), langkah selanjutnya menghubungkan atau merelasikan primary key dan foreign key pada table yang akan direlasikan.

Mengacu pada table table kategori, table penerbit, dan table buku, contoh query yang digunakan untuk merelasikan tiga table tersebut adalah :

```
create table kategori (
kode_kategori varchar (20) primary key,
Nama_kategori varchar(20)
);

create table penerbit(
kode_penerbit varchar(20) primary key,
Nama_penerbit varchar(25),
Alamat_penerbit varchar(50)
);

create table buku(
kode_buku varchar(20) primary key,
kode_kategori varchar (20),
kode_penerbit varchar(20),
judul varchar(50),
jumhal int,
tahun_terbit year,
foreign key (kode_kategori) references kategori (kode_kategori),
foreign key (kode_penerbit) references penerbit (kode_penerbit)
);
```

Hasil relasi table yang telah dibuat:



i. Memodifikasi Tabel

Perubahan tabel yang telah dibuat akan selalu dilakukan mengingat perkembangan database, termasuk diantaranya menambahkan beberapa field pada tabel, mengganti nama field maupun table.

a) Mengganti Nama Table

Query SQL untuk merubah nama tabel dengan menggunakan **RENAME**, sintax seperti berikut :

RENAME TABLE tabel_lama **TO** tabel_baru;

Contoh : merubah nama table petugas menjadi admin

```
rename table petugas to admin;
```

b) Menambah field pada table

Menambah kolom dapat diartikan sebagai langkah untuk menyisipkan field baru pada sebuah tabel. Untuk melakukan penambahan Field maka **ALTER** spesifikasi yang digunakan adalah **ADD**. Sintax yang digunakan adalah:

ALTER TABLE nama_tabel **ADD** nama_field Type_data(length);

Contoh : menambahkan kolom Jenis kelamin pada table admin

```
ALTER TABLE admin ADD Jk varchar(10);
```

c) Menghapus field pada table

Pada pembuatan database pasti terdapat kesalahan seperti pada field tabel yang berlebihan dan lain-lain. Untuk melakukan Penghapusan Field maka ALTER spesifikasi yang digunakan adalah **DROP**. Sintax yang digunakan adalah :

ALTER TABLE nama_tabel **DROP** nama_field;

Contoh : menghapus kolom Jk pada table admin

```
ALTER TABLE admin DROP Jk;
```

Beberapa ketentuan yang harus diperhatikan dalam melakukan penghapusan tabel:

- Sebelum tabel dihapus, Anda harus menutup terlebih dahulu tabel tersebut.

- Suatu tabel yang telah dihapus tidak dapat dikembalikan seperti semula.
- Data record yang ada pada tabel yang dihapus juga akan terhapus.
- Tidak ada pesan/persetujuan terlebih dahulu selama proses penghapusan tabel dilaksanakan.

EVALUASI

1. Susun DDL untuk membuat table **mahasiswa** dengan spesifikasi sebagai berikut

Field	Type	Keterangan
Nim	Char(20)	Primary Key, panjang harus tepat 20 karakter
Nama	Varchar(30)	Tidak boleh kosong
JK	Char(1)	Jenis Kelamin, hanya mempunyai nilai L untuk laki-laki dan P untuk perempuan
GolDarah	Char(2)	Golongan Darah: A, B, AB, O
Ipk	Numeric	Interval 0 sampai 4

2. Buat table peminjaman dan tabel pengembalian untuk database perpustakaan dan relasikan antar table.

Bab VIII
PERINTAH SQL: Data Manipulation Language (DML)

Pada bab ini membahas mengenai sintaks dalam SQL Data manipulation Language dan beberapa contoh detail.

Setelah mempelajari materi ini, diharapkan mahasiswa mampu:

1. Menjelaskan pengertian DML
2. Menerapkan perintah insert dalam perancangan basis data
3. Menerapkan perintah select dalam perancangan basis data
4. Menerapkan perintah update dalam perancangan basis data
5. Menerapkan perintah delete dalam perancangan basis data

1. Pengertian DML

DML adalah sebuah metode Query yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari Query DML ini untuk melakukan pemanipulasian database yang telah dibuat. Query yang dimiliki DML adalah :

INSERT	:	Memasukkan data pada Tabel Database
UPDATE	:	Pengubahan terhadap data yang ada pada Tabel Database
DELETE	:	Penghapusan data pada tabel Database

2. Perintah insert dalam perancangan basis data

Memasukkan data atau entry data, dalam semua program yang menggunakan query SQL sebagai standar permintaannya, digunakan perintah INSERT. Syarat untuk memasukkan data adalah

telah terciptanya tabel pada sebuah database. Sintax yang digunakan adalah :

INSERT INTO nama_tabel **VALUES** (‘isi_field1’, ‘isi_field2’, ‘isi_field3’,….., ‘isi_fieldN’);

Contoh :

```
INSERT INTO admin  VALUES ('p01', 'Edward', 'admin','admin');
```

Atau bisa dengan menyertakan nama field pada table

```
INSERT INTO admin (Id_petugas,Nama, Username,Password )  VALUES ('p02', 'Jenifer', 'admin','admin');
```

Hasilnya :

Id_petugas	Nama	Username	Password
p01	Edward	admin	admin
p02	Jenifer	admin	admin

3. Perintah select dalam perancangan basis data

Menampilkan data adalah hal yang sangat penting karena kita harus melihat dan menyeleksi suatu data dalam table maupun antar table. Untuk Melihat data atau *Selection,* Query yang digunakan adalah **SELECT** yang diikuti beberapa pernyataan khusus berkenaan dengan tabel yang diseleksi. Pemakaian perintah **Select** digunakan hanya untuk melakukan proses pengambilan data, bukan digunakan untuk mengubah data di dalam database.

a) **Menampilkan Data dari Sebuah Tabel**

Untuk menampilkan dari sebuah tabel dapat menggunakan Sintax berikut:

SELECT * FROM nama_tabel;

Query diatas mengartikan bahwa data dari seluruh Field yang terdapat dalam tabel akan ditampilkan.

Contoh :

```
39 select * from admin;
```

admin (4×2)

Id_petugas	Nama	Username	Password
p01	Edward	admin	12345
p02	Amin	admin	admin

Atau

SELECT (Field1, field2, ……, FieldN) **FROM** nama_tabel;

Query diatas mengartikan bahwa data yang akan ditampilkan didalam tabel hanya filed – filed tertentu.

Contoh :

```
37 select Id_petugas, Nama from admin ;
```

admin (2×2)

Id_petugas	Nama
p01	Edward
p02	Amin

b) **Menampilkan Data dengan Perintah Kondisi (Where)**

WHERE yang artinya dimana, untuk menampilkan data menggunakan perintah where (dimana) dapat menggunakan perintah berikut :

SELECT * FROM nama_tabel **WHERE** kondisi

Contoh :

Jika ingin menampilkan data dari admin dengan kriteria nama **Amin** maka sintak yang digunakan adalah:

```
39 select * from admin where Nama='amin';
```

admin (4×1)

Id_petugas	Nama	Username	Password
p02	Amin	admin	admin

c) **Menampilkan Data dengan Perintah Like**

Perintah Like kadang dibutuhkan dalam pembuatan database yaitu dalam menampilkan data tertentu yang hanya berkaitan dengan kata-kata yang diinginkan.

Query yang digunakan adalah :

SELECT * FROM nama_tabel
WHERE Kondisi **LIKE '%**nama_kaitan**%'**;

Contoh :
Jika ingin menampilkan judul buku dari table buku yang mengandung kata joomla.

```
42 SELECT * FROM buku
43    WHERE buku_judul LIKE '%joomla%';
```

buku (9×2)

buku_kode	kategori_kode	penerbit_kode	buku_judul	buku_jumlah	bu
B002	K2	P1	Step by Step Joomla	45	Ch
B004	K1	P2	Joomla Powerful Extensions	20	Jo

d) **Menampilkan Data dengan Pengurutan Sorting (Order By)**

Fungsi ini digunakan untuk melakukan pengurutan data, sehingga data dari sebuah atau beberapa tabel dapat tampil berurutan sesuai keinginan.

Pengurutan data terbagi menjadi dua :

- *ASC* (pengurutan dengan Ascending)
- *DESC* (pengurutan dengan Descending)

Sintax yang digunakan adalah :
SELECT * FROM nama_tabel **ORDER BY** kolom Type

Contoh :
Jika ingin menampilkan nama admin dari urutan abjad

```
45 select * from admin order by Nama desc;
```

admin (4×2)

Id_petugas	Nama	Username	Password
p01	Edward	admin	12345
p02	Amin	admin	admin

e) **Menampilkan Data dengan Pengelompokkan Data (Group By)**

Group By adalah fungsi untuk mengelompokkan data dalam sebuah kolom yang ditunjuk. Fungsi ini akan menghasilkan kelompok data dengan menghilangkan data yang sama dalam satu tabel. Maka apabila dalam satu kolom terdapat beberapa data yang sama maka data yang akan ditampilkan hanya salah satu. Sintax yang digunakan seperti berikut :

SELECT * FROM nama_tabel **GROUP BY** nama_kolom;

Contoh :

```
46 SELECT * FROM admin GROUP BY Nama;
```

admin (4×2)

Id_petugas	Nama	Username	Password
p02	Amin	admin	admin
p01	Edward	admin	12345

f) **Menampilkan Data dari Beberapa Tabel**

Perintah **SELECT** mempunyai banyak sekali variasi. Mungkin bisa disebut perintah yang mempunyai variasi paling banyak di antara perintah-perintah lainnya.

Seperti yang kita ketahui bahwa MySql adalah sebuah **RDBMS (Relational Database Management System)** yang artinya bahwa tabel-tabel tersebut **bisa ber-relasi/tidak ber-relasi dengan tabel lainnya**. Perintah **SELECT** adalah untuk menampilkan data yang berada di dalam tabel. Pertanyaannya adalah bagaimana jika ingin melihat isi tabel yang ber-relasi dengan tabel lainnya. Terdapat banyak cara untuk menggabungkan beberapa table yang memiliki relasi pada database. Pada workshop ini akan dibahas salah satu cara, yaitu menggunakan **inner join.**

Dengan inner join, tabel akan digabungkan dua arah, sehingga tidak ada data yang NULL di satu sisi. Sebagai contoh, kita akan menggabungkan/ mengambil data dari tabel buku dan table kategori dimana kita akan menampilkan daftar buku dengan kategori tertentu.

Misalkan isi tabel buku dan kategori adalah sebagai berikut :

perpus.kategori: 3 total baris (approximately)

kode_kategori	Nama_kategori
K01	Komik
K02	Fiksi
K03	Mapel

perpus.buku: 2 total baris (approximately)

kode_buku	kode_kategori	kode_penerbit	judul	jumhal	tahun_terbit
B01	K03	P02	Cisco dan CCNA Jaringan Komputer	557	2014
B02	K03	P01	Database System	349	2015

Jika ingin mengambil data buku dengan kategori tertentu maka menggunakan **Inner Join dengan WHERE.** Penggabungan dengan klausa WHERE memiliki bentuk umum sebagai berikut:

SELECT tabel1.*, tabel2.* FROM tabel1, tabel2
WHERE tabel1.PK=tabel2.FK;

Berikut ini perintah SQL untuk menggabungkan tabel buku dan kriteria:

```
1 select B.kode_buku, K.Nama_kategori, B.judul
2 from buku B, kategori K
3 where B.kode_kategori=K.kode_kategori;
```

Hasil #1 (3×2)

kode_buku	Nama_kategori	judul
B01	Mapel	Cisco dan CCNA Jaringan Komputer
B02	Mapel	Database System

4. Perintah update dalam perancangan basis data

Memperbarui isi data atau update data adalah sebuah proses meremajakan data lama menjadi data yang lebih baru. Namun tidak semua data dalam database yang perlu diremajakan, melainkan sebagian data yang dianggap perlu untuk diremajakan. Query SQL yang digunakan adalah UPDATE.

UPDATE nama_tabel
SET field_1 = 'data_baru', field_2 ='data_baru',
…… , field_N ='data_baru'
where field_uniq='data';

Contoh :
Jika ingin mengubah data petugas dengan id_petugas "P01" maka sintak yang digunakan adalah:

```
UPDATE admin SET Nama ='Edward', Username ='admin', Password = '12345'
where Id_petugas='P01';
```

Id_petugas	Nama	Username	Password
p01	Edward	admin	12345
p02	Jenifer	admin	admin

5. Perintah delete dalam perancangan basis data

Untuk menghapus data, MySQL memiliki query bernama DELETE. Berikut Sintax untuk menghapus semua data yang terdapat pada table.

DELETE FROM nama_tabel;

Sedangkan berikut sintax untuk menghapus data yang diinginkan dari sebuah table.

DELETE FROM nama_tabel **WHERE** kondisi;

Contoh :

Jika akan menghapus data admin yang mempunyai Id_petugas = P02

```
DELETE FROM admin WHERE Id_petugas='P02';
```

Untuk menghapus seluruh record :

```
Delete from nama_table
```

EVALUASI

Perhatikan Tabel propinsi dan Tabel kota berikut :

Tabel Propinsi

kode_prop	nama_prop	jumlah_penduduk
jabar	jawa barat	0
jak	dki jakarta	0
jateng	jawa tengah	0
jatim	jawa timur	0
yog	di yogyakarta	0

Tabel Kota

kode_kota	kode_prop	nama_kota
ban	jatim	banyuwangi
ban	yog	bantul
bks	jabar	bekasi
blr	jateng	blora
byl	jateng	boyolali
kdr	jatim	kediri
kul	yog	kulon progo
mlg	jatim	malang
sby	jatim	surabaya
sle	yog	sleman

Kerjakan soal berikut berdasarkan Tabel Propinsi dan Tabel Kota:

1. Gunakan perintah select untuk menampilkan **nama_propinsi** dari kota **blora** dan **malang**
2. Tambahkan total **jumlah_penduduk** pada **Tabel Propinsi**

Bab IX
ADVANCE SQL

Setelah mempelajari materi ini, diharapkan mahasiswa mampu:
1. Menerapkan fungsi Agregat dalam perancangan basis data
2. Menerapkan Sub Query dalam perancangan basis data
3. Menerapkan View dalam perancangan basis data
4. Menerapkan Trigger dalam perancangan basis data

1. Fungsi Agregat dalam perancangan basis data

Fungsi agregat merupakan fungsi yang menerima koleksi nilai dan mengembalikan nilai tunggal sebagai hasilnya. Menurut Indrajani (2014) Standar ISO mendefinisikan lima fungsi agregat yang ditunjukkan pada Tabel 9.1.

Tabel 9.1 Fungsi Agregat

Fungsi	Deskripsi
COUNT	Mengembalikan jumlah (banyaknya atau kemunculannya) nilai di suatu kolom
SUM	Mengembalikan jumlah (total atau *sum*) nilai di suatu kolom
AVG	Mengembalikan rata-rata (*average*) nilai di suatu kolom
MIN	Mengembalikan nilai terkecil (*minimal*) di suatu kolom
MAX	Mengembalikan nilai terbesar (*maximal*) di suatu kolom

Keyword DISTINCT dapat dimanfaatkan untuk mengeliminasi duplikasi kemunculan data yang sama.

Sintaks *keyword* **DISTINCT** diperlihatkan sebagai berikut:

```
SELECT DISTINCT A1, A2, ..., An
FROM r1, r2, r3, ..., rm
WHERE P
```

Contoh penggunaan fungsi aggregate, perhatikan Tabel 9.2 .

Tabel 9.2 Penggunaan Fungsi Agregat

kode_mk	nama_mk	sks	semester
PTI447	Praktikum Basis Data	1	3
TIK342	Praktikum Basis Data	1	3
PTI333	Basis Data Terdistribusi	3	5
TIK123	Jaringan Komputer	2	5
TIK333	Sistem Operasi	3	5
PTI123	Grafika Komputer	3	5
PTI777	Sistem Informasi	2	3

a) Contoh Penggunaan COUNT(*)

Fungsi Count yang mempunyai bentuk COUNT(field) digunakan untuk menghitung banyaknya data dalam suatu table dari hasil query. Dengan fungsi ini dapat dihitung jumlah data yang diperoleh atas dasar field tertentu atau seluruh field pada query. Penggunaan COUNT untuk mendapatkan jumlah data, maka sintak yang digunakan adalah:

```
1 SELECT COUNT(*) AS jumlah_record
2 FROM matakuliah;
```

Maka hasil yang ditampilkan adalah “7” karena jumlah total mata kuliah yang terdapat pada table adalah sejumlah 7 mata kuliah.

b) Contoh Penggunaan SUM

Fungsi sum mempunyai bentuk SUM(x) digunakan untuk menghasilkan nilai penjumlahan dari suatu data bilangan dimana x adalah nilai numerik atau nama field yang memiliki nilai numeric. Penggunaan SUM untuk mendapatkan jumlah total sks dari table matakuliah, maka sintak yang digunakan adalah:

```
1 SELECT SUM(sks) AS total_sks FROM matakuliah;
```

c) Contoh Penggunaan MIN, MAX, AVG

Penggunaan MIN untuk mendapatkan sks terendah dari table matakuliah, maka sintak yang digunakan adalah:

```
1 SELECT MIN(sks) AS min_sks FROM matakuliah;
```

Penggunaan MAX untuk mendapatkan sks tertinggi dari table matakuliah, maka sintak yang digunakan adalah:

```
1 SELECT MAX(sks) AS min_sks FROM matakuliah;
```

Fungsi AVG yang mempunyai bentuk AVG(x) digunakan untuk menghasilkan nilai rata-rata dari suatu data bilangan. Dimana x adalah nilai numerik atau nama field yang memiliki nilai numerik atau rumus yang menghasilkan nilai numerik. Penggunaan AVG untuk mendapatkan rata-rata sks dari table matakuliah, maka sintak yang digunakan adalah:

```
1 SELECT AVG(sks) AS rata_rata FROM matakuliah;
```

d) Pengelompokan Data

Pengelompokan data berdasarkan semester dari table matakuliah, sintak yang digunakan adalah:

```
SELECT semester, COUNT(semester) AS jumlah
FROM matakuliah
GROUP BY(semester)
```

e) Menyaring Hasil Fungsi Agregat

Penyeleksian data sks dengan jumlah sks lebih besar dari 3 dan dikelompokkan berdasarkan semester dari table matakuliah, sintak yang digunakan adalah:

```
SELECT semester, COUNT(semester) AS jumlah
FROM matakuliah
GROUP BY(semester)
HAVING jumlah>3
```

Aturan-aturan yang berlaku :

a. SUM dan AVG digunakan untuk field numeric, sedangkan COUNT, MIN, dan MAX dapat digunakan untuk field numeric dan non numeric
b. COUNT (*) berfungsi untuk menghitung seluruh baris dalam table walaupun terdapat NULL atau duplikasi.
c. Menggunakan DISTINCT sebelum nama kolom berfungsi untuk menghilangkan duplikasi
d. DISTICT tidak berpengaruh terhadap operasi MIN atau MAX, tetapi berpengaruh pada SUM atau AVG.
e. Fungsi aggregate dapat digunakan dalam SELECT dan HAVING

2. **Sub Query**

Subquery (disebut juga subselect atau nested select/query atau inner- select) adalah query SELECT yang ada di dalam perintah SQL lain— misalnya SELECT, INSERT, UPDATE, atau DELETE. Keberadaan subquery secara nyata mampu

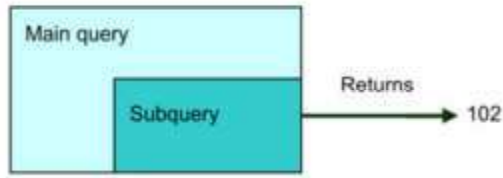
menyederhanakan persoalan- persoalan rumit berkaitan query data.

Sebagai contoh, terdapat pernyataan sebagai berikut: "**Dapatkan data mahasiswa yang alamatnya sama dengan mahasiswa dengan nim 104".** Secara normal, diperlukan dua tahapan untuk menyelesaikan permasalahan tersebut. Langkah awal yang harus dilakukan adalah mendapatkan alamat dari mahasiswa yang memiliki nim 104, selanjutnya dengan menggunakan operator bias diseleksi data mahasiswa yang alamatnya sama dengan mahasiswa yang mempunyai nim 104. Penyelesaian dengan sub query, dapat dilakukan dengan satu tahapan.

Bentuk umum dari sub query adalah :

```
SELECT A1, A2, ..., An
FROM x1, x2, x3, ..., xm
WHERE P
   (SELECT A1, A2, ..., An
   FROM x1, x2, x3, ..., xm
   WHERE P);
```

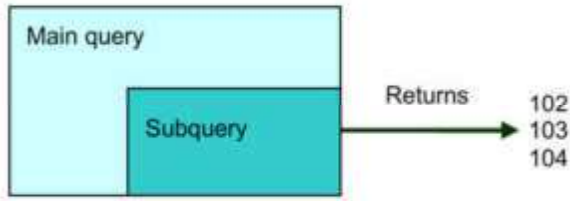
Terdapat 3 jenis sub query diantaranya scalar sub query, multiple-row sub query, dan multiple-column sub query. **Subquery baris tunggal (scalar)** hanya mengembalikan hasil satu baris data. Operator yang digunakan adalah =, >, >=, <, <=, atau <>. Ilustrasi scalar sub query ditunjukkan pada Gambar 9.1



Gambar 9.1 Ilustrasi Scalar Sub Query

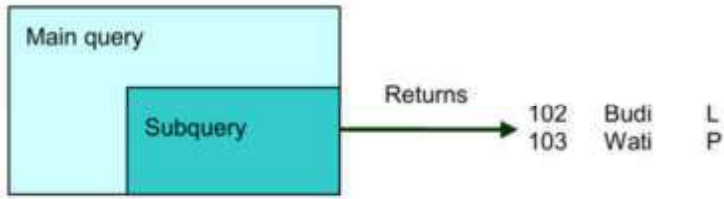
Subquery baris ganda (multiple-row) mengembalikan lebih dari satu baris data. Subquery baris ganda dapat menggunakan

operator komparasi IN, ANY/SOME, atau ALL. Operator IN memiliki arti sama dengan anggota di dalam list. Operator ANY/SOME digunakan untuk membandingkan suatu nilai dengan setiap nilai yang dikembalikan oleh sub query. Operator ALL digunakan untuk membandingkan nilai dengan semua nilai yang dikembalikan oleh query. Ilustrasi multiple-row sub query ditunjukkan pada Gambar 9.2



Gambar 9.2 Ilustrasi multiple-row Sub Query

Subquery kolom ganda (multiple-column) mengembalikan lebih dari satu baris dan satu kolom data. Ilustrasi multiple-column sub query ditunjukkan pada Gambar 9.3



Gambar 9.3 Ilustrasi multiple-column Sub Query

Sebagai contoh, perhatikan table mahasiswa, table ambil_mk, table mata kuliah, table dosen, dan table jurusan sebagai berikut:

Tabel mahasiswa

nim	nama	jenis_kelamin	alamat
101	Arif	L	Jl. Kenangan
102	Budi	L	Jl. Jombang
103	Wati	P	Jl. Surabaya
104	Ika	P	Jl. Jombang
105	Tono	L	Jl. Jakarta
106	Iwan	L	Jl. Bandung
107	Sari	P	Jl. Malang

Tabel ambil_mk

nim	kode_mk
101	PTI447
103	TIK333
104	PTI333
104	PTI777
111	PTI123
123	PTI999

Tabel matakuliah

kode_mk	nama_mk	sks	semester	kode_dos
PTI447	Praktikum Basis Data	1	3	11
TIK342	Praktikum Basis Data	1	3	11
PTI333	Basis Data Terdistribusi	3	5	10
TIK123	Jaringan Komputer	2	5	33
TIK333	Sistem Operasi	3	5	10
PTI123	Grafika Multimedia	3	5	12
PTI777	Sistem Informasi	2	3	99

Tabel dosen

kode_dos	nama_dos	alamat_dos
10	Suharto	Jl. Jombang
11	Martono	Jl. Kalpataru
12	Rahmawati	Jl. Jakarta
13	Bambang	Jl. Bandung
14	Nurul	Jl. Raya Tidar

Tabel jurusan

kode_jur	nama_jur	kode_dos
TE	Teknik Elektro	10
TM	Teknik Mesin	13
TS	Teknik Sipil	23

Contoh penerapan Scalar Sub Query

Untuk mendapatkan data mahasiswa yang alamatnya sama dengan mahasiswa dengan nim 104 bisa digunakan scalar sub query sebagai berikut:

```
SELECT * FROM mahasiswa
WHERE alamat = (SELECT alamat
                FROM mahasiswa
                WHERe nim=104)
```

Jika di eksekusi maka hasil yang ditampilkan adalah :

```
+-----+------+---------------+-------------+
| nim | nama | jenis_kelamin | alamat      |
+-----+------+---------------+-------------+
| 102 | Budi | L             | Jl. Jombang |
| 104 | Ika  | P             | Jl. Jombang |
+-----+------+---------------+-------------+
2 rows in set (0.00 sec)
```

Contoh penerapan Multiple-Row Sub Query

Penggunaan **Operator IN**, sebagai contoh jika ingin mendapatkan data mahasiswa yang mengambil mata kuliah.

```
SELECT nim, nama FROM mahasiswa
WHERE nim IN (SELECT nim FROM ambil_mk)
```

Jika di eksekusi maka hasil yang ditampilkan adalah :

```
+-----+------+
| nim | nama |
+-----+------+
| 101 | Arif |
| 103 | Wati |
| 104 | Ika  |
+-----+------+
3 rows in set (0.00 sec)
```

Penggunaan **Operator ANY/SOME**, sebagai contoh jika ingin mendapatkan data matakuliah yang memiliki sks lebih kecil dari sembarang sks matakuliah di semester 5.

```
SELECT * FROM matakuliah
WHERE sks < ANY
    (SELECT sks FROM matakuliah WHERE semester=5)
```

Jika di eksekusi maka hasil yang ditampilkan adalah :

```
+---------+----------------------+-----+----------+----------+
| kode_mk | nama_mk              | sks | semester | kode_dos |
+---------+----------------------+-----+----------+----------+
| PTI447  | Praktikum Basis Data |   1 |        3 |       11 |
| PTI777  | Sistem Informasi     |   2 |        3 |       99 |
| TIK123  | Jaringan Komputer    |   2 |        5 |       33 |
| TIK342  | Praktikum Basis Data |   1 |        3 |       11 |
+---------+----------------------+-----+----------+----------+
4 rows in set (0.00 sec)
```

Penggunaan operator ALL, sebagai contoh jika ingin mendapatkan data matakuliah yang memiliki sks yang lebih kecil dari semua sks mata kuliah di semester 5

```
SELECT * FROM matakuliah
WHERE sks < ALL
    (SELECT sks FROM matakuliah WHERE semester=5)
```

Jika di eksekusi maka hasil yang ditampilkan adalah :

```
+---------+----------------------+-----+----------+----------+
| kode_mk | nama_mk              | sks | semester | kode_dos |
+---------+----------------------+-----+----------+----------+
| PTI447  | Praktikum Basis Data |   1 |        3 |       11 |
| TIK342  | Praktikum Basis Data |   1 |        3 |       11 |
+---------+----------------------+-----+----------+----------+
2 rows in set (0.16 sec)
```

Contoh penerapan Multiple-column Sub Query

Penggunaan multiple-column sebagai contoh jika ingin menampilkan data mata kuliah yang semester dan sksnya sesuai dengan semester dan sks mata kuliah dengan kode “PTI333”.

```
SELECT * FROM matakuliah
WHERE (semester, sks) IN
              (SELECT semester, sks FROM matakuliah
              WHERE kode_mk='PTI333')
```

Jika di eksekusi maka hasil yang ditampilkan adalah :

kode_mk	nama_mk	sks	semester	kode_dos
PTI123	Grafika Multimedia	3	5	12
PTI333	Basis Data Terdistribusi	3	5	10
TIK333	Sistem Operasi	3	5	10

3 rows in set (0.00 sec)

Operasi pada **sub query** dapat juga dikombinasikan dengan **fungsi agregat,** sebagai contoh jika ingin mendapatkan data matakuliah yang memiliki sks sama dengan sks terbesar.

```
SELECT * FROM matakuliah
WHERE sks = (SELECT MAX (sks) FROM matakuliah)
```

Jika di eksekusi maka hasil yang ditampilkan adalah :

kode_mk	nama_mk	sks	semester	kode_dos
PTI123	Grafika Multimedia	3	5	12
PTI333	Basis Data Terdistribusi	3	5	10
TIK333	Sistem Operasi	3	5	10

3 rows in set (0.06 sec)

3. **View**

View merupakan *virtual table* yang dibangun dari satu atau beberapa table. View tidak membuat penyimpanan secara fisik seperti table, namun hanya menyimpan referensi/pointer ke record pada table-table yang berkaitan. Yang tersimpan dalam view hanyalah perintah seleksi data, namun bentuk fisik view dapat diperlakukan sebagaimana sebuah table.

Keuntungan view diantaranya :

a. Kompleksitas pengambilan data dapat disembunyikan dalam view
b. User hanya perlu tahu nama dari view
c. Data yang direferensi oleh view dapat juga di select dengan perintah lainnya dan hasilnya kemudian di filter
d. Isi data di view dapat diupdate (data hasil update akan diteruskan pada isi data tabel yang direference oleh view)

Struktur umum sebuah view adalah :

```
CREATE VIEW namaView (kolom1, kolom2, …) AS
SELECT kolom_a, kolom_b, …
FROM namaTable
WHERE predikat
```

Melalui view dapat dilakukan INSERT, UPDATE dan DELETE dengan beberapa batasan. Salah satu contoh penggunaan view untuk menampilkan kode dan nama provinsi, perhatikan table berikut :

Tabel propinsi

kode_prop	nama_prop	jumlah_penduduk
jabar	jawa barat	234987
jak	dki jakarta	342561
jateng	jawa tengah	374653
jatim	jawa timur	236457
yog	di yogyakarta	276357

maka contoh sintak perintah view sebagai berikut :

```
1  CREATE VIEW vw_propinsi AS
2  SELECT kode_prop, nama_prop
3  FROM propinsi
```

Untuk menampilan atau mengakses data view dapat menggunakan perintah **select** sebagai berikut:

```
SELECT * FROM vw_propinsi
```

Perintah view dapat digunakan untuk menyeleksi data dari beberapa table, seperti contoh sebagai berikut.

Perhatikan **table propinsi** sebelumnya dan **table kota** berikut:

Tabel kota

kode_kota	kode_prop	nama_kota
ban	jatim	banyuwangi
ban	yog	bantul
bks	jabar	bekasi
blr	jateng	blora
byl	jateng	boyolali
kdr	jatim	kediri
kul	yog	kulon progo
mlg	jatim	malang
sby	jatim	surabaya
sle	yog	sleman

Perintah yang digunakan untuk menampilkan kota tertentu termasuk dalam propinsi mana, maka digunakan sintak :

```
1  CREATE VIEW vw_kota AS
2  SELECT k.kode_kota, k.nama_kota,
3         p.kode_prop, p.nama_prop
4  FROM kota k, propinsi p
5  WHERE k.kode_prop=p.kode_prop
```

Contoh perintah view yang digunakan untuk **mengupdate** data jumlah penduduk pada propinsi jawa timur :

```
1  UPDATE vw_propinsi SET jumlah_penduduk =’35000’
2  WHERE kode_prop = ’jatim’
```

Field nama merupakan referensi dari field nama yang ada pada tabel propinsi. Perubahan isi field nama di view vw_propinsi akan mempengaruhi isi field jumlah penduduk di tabel propinsi. Perintah view yang digunakan untuk **menghapus** data pada view hamper sama dengan update data pada view.

4. **Trigger**

Trigger merupakan procedural yang dieksekusi secara otomatis sebagi respon atas suatu kejadian yang berhubungan dengan table. Trigger dieksekusi jika terjadi database event seperti, INSERT, UPDATE, dan DELETE. Trigger INSERT akan aktif pada saat adanya pemasukan record baru, Trigger UPDATE akan aktif pada saat ada perubahan pada sebuah record, dan Trigger DELETE akan aktif pada saat terdapat penghapusan record.

SIntaks umum dari trigger adalah:

```
CREATE
    [DEFINER = { user | CURRENT_USER }]
    TRIGGER trigger_name trigger_time trigger_event
    ON tbl_name FOR EACH ROW trigger_body
```

Keterangan :

a. **trigger_name** : nama trigger.
b. **trigger_time** : kapan kita mengeksekusi trigger, apakah sebelum atau sesudah perubahan pada row data table. Jadi pilihannya adalah **AFTER** atau **BEFORE**.
c. **trigger_event** : merupakan event atau peristiwa yang menyebabkan trigger dilakukan. Pilihan event tersebut adalah **INSERT**, **UPDATE**, **DELETE.**
d. **tbl_name** : nama table.
e. **trigger_body** : *statement-statement* perintah SQL yang akan dilakukan. Jika perintahnya lebih dari satu maka gunakan dalam blok statement **BEGIN ... END**.

f. Jika **DEFINER** dispesifikasikan maka kita memutuskan trigger tersebut dijalankan hanya oleh user tertentu (dalam format penulisan **user@host**). Jika tidak dispesifikasikan, maka user yang melakukan perubahan (CURRENT_USER) adalah pilihan *default*.

Contoh Penggunaan: Trigger After Insert

Pada saat buku dipinjam petugas harus memasukkan data ke dalam tabel pinjam. Pada situasi seperti ini akan menggunakan trigger dengan metode after insert pada tabel peminjaman. Ketika tabel Peminjaman di isi dengan data peminjam maka akan terjadi proses pengurangan stok buku pada tabel buku.

Maka perintah yang digunakan dalam bahasa query adalah:

```
CREATE TRIGGER tr_pinjam AFTER INSERT
   ON Peminjaman FOR EACH ROW
BEGIN
update buku set jml_buku=jml_buku - 1
where kode_buku =new.kode_buku ;
END
```

Pada contoh diatas, pada saat terjadi transaksi peminjaman buku, maka secara otomatis jumlah buku pada table buku berkurang 1. Untuk penerapan Event UPDATE dan DELETE pada trigger menggunakan mekanisme yang sama.

EVALUASI

Untuk menjawab soal, silahkan perhatikan table berikut:

TABEL 1 : Tabel Penjualan Barang

Kode_buku	Jenis_Buku	Nama_buku	Harga	Stok
NL123	Novel	Tere Liye: Rindu	53500	10
NL234	Novel	Rembulan Terang	62200	15
TS456	Teks	Statistika	80500	9
KK678	Komik	Doraemon Vol 2	25500	2
KK321	Komik	Sincan Vol 1	24250	5
TS980	Teks	Basis Data	150000	2
KS222	Kamus	Kamus Bahasa Inggris	35000	18

Berdasarkan Tabel 1

1. Dapatkan harga buku dengan harga termurah dan harga termahal dan kelompokkan berdasarkan jenis bukunya!
2. Dengan menggunakan sub query, dapatkan judul buku dan harga buku dengan stok di atas 10!
3. Buat view **buku mahal** untuk menampilkan Judul buku dan jenis buku dengan harga diatas 50000 dengan nama!
4. Dengan menerapkan Trigger, buat perintah yang digunakan untuk mengurangi stok buku ketika terjadi proses pembelian buku!

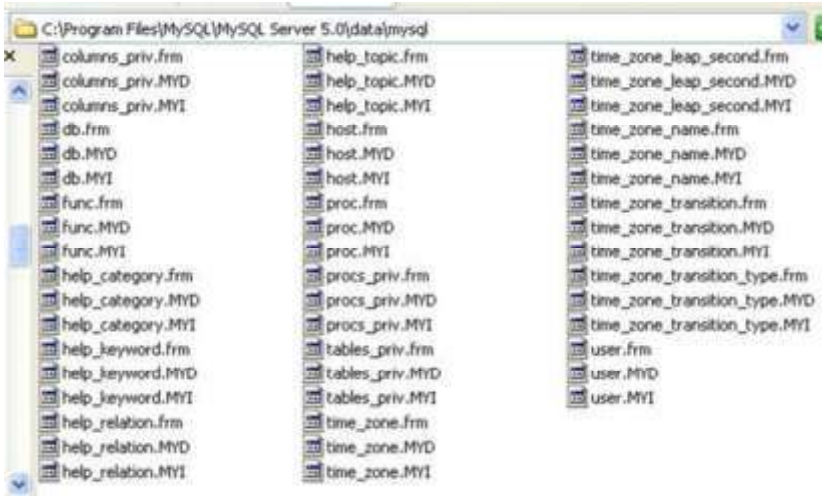
Bab X
BACKUP DAN RECOVERY BASIS DATA

Setelah mempelajari materi ini, diharapkan mahasiswa mampu:
1. Menerapkan Backup Basis Data
2. Menerapkan Recovery Basis Data

Basis data adalah suatu sistem yang harus dapat diandalkan kinerjanya. Tetapi dalam proses pemanfaatan system basis data tidak selamanya dapat berjalan dengan lancar. Adakalanya suatu basis data mengalami gangguan. Gangguan- gangguan itu dapat menyebabkan kerusakan data pada sistem tersebut. Kerusakan data pada sistem basis data dapat dicegah dengan berbagai macam teknik. Untuk pencegahan kerusakan data tersebut dapat dilakukan dengan menggunakan metode backup dan restore. Metode backup dan restore merupakan metode yang sudah lama digunakan untuk menanggulangi kerusakan data.

Metode backup dan restore basis data ini juga digunakan untuk membuat salinan data yang ada pada server secara berkala untuk proses maintenance. Server pada MySQL memiliki berbagai jenis table, diantaranya MyISAM, ISAM, InnoDB, dan DBD. Jenis table tersebut dapat memiliki beberapa file yang membentuk kesatuan table. Sebagai contoh, tipe tabel MyISAM memiliki tiga buah file, yaitu *file.frm*, *file.myd* dan *file.myi*. File-file tersebut merupakan data file bagi satu buah tabel bertipe MyISAM dapat ditunjukkan pada Gambar 10.1. Sementara pada Tabel InnoDB memiliki file data berupa file tipe.frm dapat ditunjukkan pada Gambar 10.2.

Terdapat banyak cara yang dapat dilakukan untuk melakukan proses backup dan recovery basis data. Salah satu cara paling sederhana dapat dilakukan dengan menyalin data file dari tabel - tabel yang ada pada server MySQL, sebagai contoh, jika ingin membackup file pada tabel berjenis MyISAM, maka dapat meng-copy file-file yang berekstensi *.frm,.myd* dan *.myi*. Contoh file tersebut dapat ditunjukkan pada Gambar 10.1.



Gambar 10.1 File data table MyISAM

Gambar 10.2. File data table InnoDB

1. **Membackup Basis Data**

Proses backup basis data di My SQL terdapat beberapa cara diantaranya dengan menggunakan query, melalui administrator, dan dengan mengakses browser pada localhost.

a. **Proses backup dengan sintaks SQL**

Sintaks yang digunakan untuk melakukan backup dengan query adalah:

```
SELECT INTO OUTFILE
BACKUP TABLE
LOAD DATA INFILE
```

Sebelum proses backup dilakukan, dipastikan tidak ada proses penulisan atau perubahan data dalam table. Sintak yang digunakan untuk melakukan penguncian table dengan perintah:

```
LOCK TABLES Nama_Table WRITE;
```

Sebagai contoh ketika akan melakukan backup pada table peminjaman. Maka tahap awal penguncian table sintaks yang dibuat adalah :

```
LOCK TABLES peminjaman WRITE;
```

Setelah proses penguncian table selesai maka langkah selanjutnya adalah melakukan pengosongan memory (FLUSH). Ini dilakukan untuk memastikan tidak ada proses yang berlangsung terhadap data pada table. Sintaks yang digunakan sebagao proses pengosongan memory adalah:

```
FLUSH Tables;
```

Setelah proses penguncian dan pengosongan table, maka proses backup basis data dapat dilakukan. Proses backup table dapat dilakukan secara keseluruan atau sebagian.

Sintaks untuk melakukan backup table adalah :

```
SELECT * INTO OUTFILE 'backup_peminjaman'
   FROM peminjaman;
```

Jika proses backup table berhasil maka aka nada pesan:

```
Query OK, 1 row affected <0,05 sec>
```

Proses pengecekan juga dapat dilakukan dengan melihat apakah file dengan nama **backup_peminjaman** telah terdapat pada lokasi penyimpanan file.

b. Membackup melalui localhost PHPMyAdmin

Proses backup basis data yang kedua dapat dilakukan melalui editor bantu dengan localhost PHPMyAdmin. Aplikasi PHPMyAdmin merupakan free software. Selain untuk melakukan backup dan restore basis data aplikasi ini juga dapat digunakan untuk memanipulasi dan membuat basis data. Tampilan dari PHPMyAdmin ditunjukkan pada Gambar 10.3.

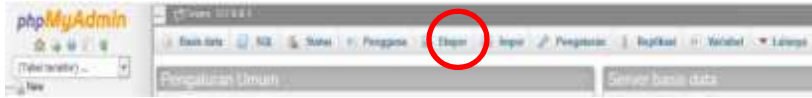
Gambar 10.3 Tampilan Aplikasi PHPMyAdmin

Aplikasi PHPMyAdmin dapat dijalankan melalui browser dengan mengetikkan pada URL ://localhost/PHPMyAdmin. Sebelum mengakses PHPMyAdmin maka Server local (missal : Xampp) harus diaktifkan terlebih dahulu. Server local dapat ditunjukkan pada Gambar 10. 4

Gambar 10.4 Tampilan server local

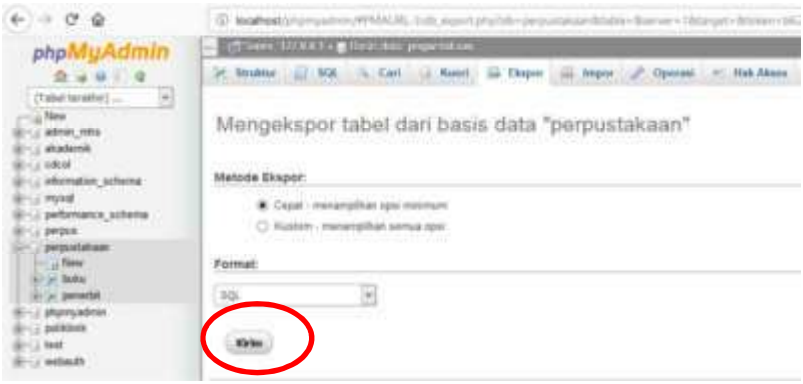
Langkah backup basis data melalui PHPMyAdmin adalah sebagai berikut :

1) Jalankan PHPMyAdmin melalui browser sehingga muncul tampilan seperti Gambar 10.3.
2) Jika ingin melakukan backup terhadap 1 basis data pilih basis data yang akan di backup dan klik menu export pada Gambar 10.5

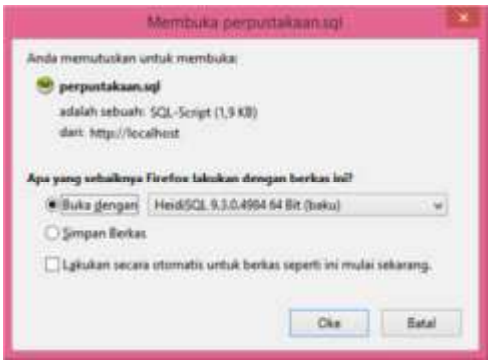


Gambar 10.5. Menu pada PHPMyAdmin

Setelah tampilan pada menu export keluar, pilih format backup (SQL, Latex, Excel, Word, atau CSV) lalu klik **Kirim** (ditunjukkan pada Gambar 10.6). Setelah proses selesai maka database perpustakaan akan terdownload seperti pada Gambar 10.7.



Gambar 10.6. Tampilan menu Export

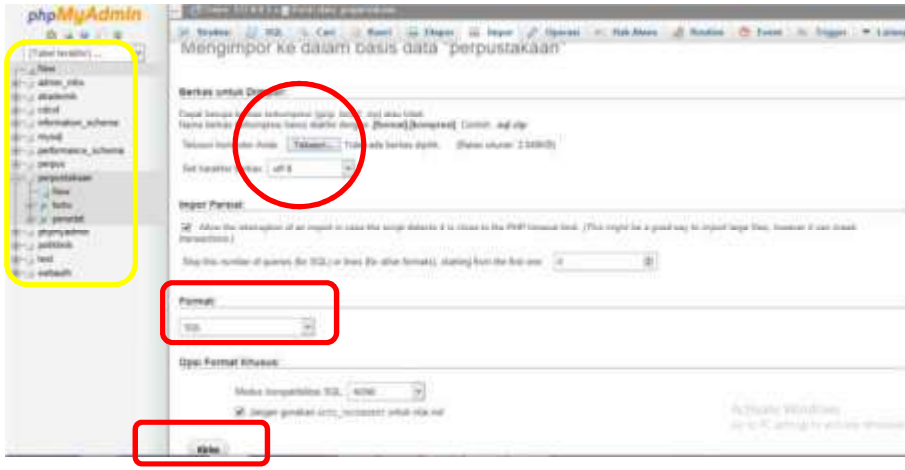


Gambar 10.7 jendela dialog file download

2. Merestore Basis Data

Seperti proses backup basis data, proses restore pada basis data juga terdapat beberapa cara seperti yang telah disebutkan sebelumnya. Pada buku ajar ini akan dijelaskan restore basis data dengan menggunakan aplikasi PHPMyAdmin.

Langkah awal yang dilakukan untuk restore basis data adalah memilih basis data yang akan direstore. Lalu klik menu **IMPORT** yang ada pada Gambar 10. 5. Tampilan menu import ditunjukkan pada Gambar 10.8.



Gambar 10.8 Tampilan menu import

Setelah menu import keluar maka pilih button **Telusuri** untuk mengambil file backup basis data yang telah tersimpan pada computer. Pastikan format yang dipilih sesuai. Jika semua proses selesai pilih button **kirim**. Untuk memastikan apakah basis data yang telah direstore telah masuk, maka periksa pilihan list basis data

pana menu sebelah kiri. Jika basis data telah terpasang, maka proses restore telah berhasil.

EVALUASI

Jawablah pertanyaan di bawah ini :

1. Jelaskan mengapa backup perlu diterapkan pada basis data!
2. Menurut pendapat kalian, kapankah waktu terbaik untuk melakukan backup terhadap suatu basis data?
3. Sebutkan proses backup dan restore yang anda ketahui dalam database MySQL

DAFTAR PUSTAKA

Date, C.J. 2000, *An Introduction to Database System*, Addison Wesley Publishing Company, Vol. 7, New York.

Elmasri dan Navathe. 2007. *Fundamentals of Database Systems, Fifth Edition. Boston: Pearson Education, Inc. Addison Wesley*

Elmasri, Ramez; Navathe, Shamkant B., 2001, F*undamentals of Database Systems*, The Benjamin/ Cummings Publishing Company, Inc., California.

Fatansyah. 2012. Basis Data. Bandung:Informatika

Indrajani. 2015. *Database Design.* Jakarta: Elex Media Komputindo

Kadir, Abdul. 2008. *Belajar Database menggunakan MySQL*. Yogyakarta : Andi Offset

Mardiani, Eri, Rahmansyah, Nur, dkk. 2016. Aplikasi Penggajian Menggunakan Visual Basic, MySQL, dan Data Report. Jakarta: Elex Media Komputindo

Nugroho, Bunafit. 2015. Aplikasi Pemrograman Web Dinamsi dengan PHP dan MySQL. Yogyakarta: Gava Media.

Puspitasari, Dwi. 2016. Normalisasi Tabel pada Basis Data Relasional. Prosiding Sentia. Vol 8. A.341-344.

Shalahudin, M. 2010. Modul Pembelajaran Struktur Data. Bandung: Modula.

BIODATA PENULIS

Fitria Nur Hasanah, M.Pd. dilahirkan di Lamongan, 23 September 1987. Pada tahun 2008, penulis mendapatkan gelar Diploma Manajemen Informatika di Universitas Brawijaya, lulus Sarjana Pendidikan Teknik Informatika di Universitas Negeri Malang tahun 2011, kemudian melanjutkan gelar magister Pendidikan Kejuruan dengan konsentrasi Informatika di Universitas Negeri Malang lulus tahun 2015. Tahun 2011 penulis mengawali karirnya sebagai Guru di SMK Nasional Malang bidang Rekayasa Perangkat Lunak dan Teknik Komputer Jaringan. Selanjutnya tahun 2016 menjadi Dosen tetap di Prodi Pendidikan Teknologi Informasi Universitas Muhammdiyah Sidoarjo . Beberapa penelitian yang pernah dilakukan oleh penulis adalah tentang model pembelajaran dan pengembangan media pembelajaran.

Rahmania Sri Untari, M.Pd dilahirkan di Malang, 19 April 1989. Pendidikan S1 diselesaikan di Program Studi Pendidikan Teknik Informatika Universitas Negeri Malang pada tahun 2011. Pendidikan S2 di Program Pascasarjana Jurusan Pendidikan Kejuruan diselesaikan pada tahun 2015. Pada tahun 2016, penulis melanjutkan Pendidikan S3 Jurusan Pendidikan Kejuruan di Universitas Negeri Malang dan sekarang masih berada pada masa studi. Pada tahun 2011 penulis mengawali karirnya di SMA Negeri 6 Surabaya sebagai guru Teknik Informatika. Selanjutnya, pada tahun 2015 penulis melanjutkan karirnya untuk menjadi Dosen tetap di Prodi Pendidikan Teknologi Informasi Universitas Muhammadiyah Sidoarjo. Beberapa penelitian yang pernah dilakukan penulis adalah tentang model pembelajaran, kurikulum, pengembangan media pembelajaran.

UNIVERSITAS MUHAMMADIYAH SIDOARJO
FAKULTAS PSIKOLOGI DAN ILMU PENDIDIKAN (FPIP)
PRODI PENDIDIKAN GURU ANAK USIA DINI (PG-PAUD) TERAKREDITASI B NOMOR : 2231 SK/BAN-PT/Akred/S/VII/2017
PRODI PENDIDIKAN GURU SEKOLAH DASAR TERAKREDITASI B NOMOR : 743/SK/BAN-PT/Akred/S/III/2018
PRODI PENDIDIKAN BAHASA INGGRIS TERAKREDITASI B NOMOR : 3057/SK/BAN-PT/Akred/S/XI/2018
PRODI PENDIDIKAN ILMU PENGETAHUAN ALAM (IPA) TERAKREDITASI B NOMOR : 432/SK/BAN-PT/Akred/S/III/2019
PRODI PENDIDIKAN TEKNOLOGI INFORMASI (TI) SK NOMOR : 0207/SK/BAN-PT/Akred/S/I/2017
PRODI PSIKOLOGI TERAKREDITASI B NOMOR : 0124/SK/BAN-PT/Akred/S/III/2016
TERAKREDITASI INSTITUSI (UNIVERSITAS) B
BAN-PT No. 225/SK/BAN-PT/Akred/PT/2019
Kampus I : Jl. Mojopahit 666 B Sidoarjo, Telp: 031 – 8945444, Fax: 031-8949333
Website: www.fpip.umsida.ac.id email:fpip@umsida.ac.id

SURAT TUGAS

Nomor: 215/II.3.AU/08.00/B/KEP/VII/2020

Yang bertanda tangan di bawah ini;

Nama	: Dr. Akhtim Wahyuni, M.Ag
NIK	: 202200
Pangkat/ Golongan	: Lektor/ IIId
Jabatan	: Dekan Fakultas Psikologi dan Ilmu Pendidikan
Unit Kerja	: Fakultas Psikologi dan Ilmu Pendidikan

Menugaskan:

N a m a	: **Fitria Nur Hasanah, M.Pd**
NIK	: 216613
Pangkat Golongan	: Asisten Ahli/ IIIb
Jabatan	: Dosen Tetap
Unit Kerja	: Fakultas Psikologi dan Ilmu Pendidikan

Untuk membuat buku ajar mata kuliah Rekayasa Perangkat Lunak di Fakultas Psikologi dan Ilmu Pendidikan Universitas Muhammadiyah Sidoarjo Semester Ganjil Tahun Akademik 2020/2021. Demikian surat tugas ini dibuat, untuk dapat dipergunakan sebagaimana mestinya.

Sidoarjo, 15 Juli 2020
DEKAN FPIP,

Dr. Akhtim Wahyuni, M.Ag

ISBN 978-623-6833-89-6 (PDF)

UMSIDA Press
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit 666 B Sidoarjo
Sidoarjo, Jawa Timur

Fitria Nur Hasanah, M.Pd.
Rahmania Sri Untari, M.Pd.

REKAYASA PERANGKAT LUNAK

BUKU AJAR
REKAYASA PERANGKAT LUNAK

Oleh
Fitria Nur Hasanah, M.Pd
Rahmania Sri Untari, M.Pd

Diterbitkan oleh
UMSIDA PRESS

UNIVERSITAS MUHAMMADIYAH SIDOARJO
2020

BUKU AJAR
REKAYASA PERANGKAT LUNAK

Penulis:
Fitria Nur Hasanah, M.Pd
Rahmania Sri Untari, M.Pd

ISBN :
978-623-6833-89-6

Editor:
Mohammad Suryawinata, S.Pd., M.Kom

Design Sampul dan Tata Letak:
Mochammad Nashrullah, S.Pd.
Amy Yoga Prajati, S.Kom

Penerbit:
UMSIDA Press
Anggota IKAPI No. 218/Anggota Luar Biasa/JTI/2019
Anggota APPTI No. 002 018 1 09 2017

Redaksi
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit No 666B
Sidoarjo, Jawa Timur

Cetakan Pertama, September 2020

KATA PENGANTAR

Alhamdulillah Puji Syukur Penulis sampaikan kehadirat Allah SWT, yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Buku Ajar Rekayasa Perangkat Lunak dengan baik. Buku ajar ini ditujukan kepada mahasiswa yang mengampuh mata kuliah Rekayasa Perangkat Lunak. Dengan dibuatnya Buku Ajar ini penulis berharap agar dapat bermanfaat dan membantu dalam memahami materi Rekayasa Perangkat Lunak yang semakin berkembang.

Ucapkan terima kasih penulis ucapkan kepada :

1. Dr. Hidayatulloh, M.Si., Rektor UMSIDA yang memberikan kesempatan luas kepada tim penulis untuk berkarya dan menyumbangkan pikiran sehingga buku ajar ini terselesaikan.
2. Dr. Akhtim Wahyuni, M.Ag Dekan Fakultas Psikologi dan Ilmu Pendidikan yang memberikan arahan dan motivasi kepada penulis dalam menyelesaikan buku ajar ini.
3. Rekan-rekan dosen Pendidikan Teknologi Informasi UMSIDA yang telah berbagi pengalaman dalam penulisan buku ajar, serta membantu dalam penyelesaikan buku ajar Rekayasa Perangkat Lunak.

Penulis menyadari sekali bahwa Buku Ajar ini masih jauh dari kesempurnaan, maka dari itu penulis mengharapkan kritik dan saran pembaca demi kesempurnaan Buku Ajar ini kedepannya. Akhir kata penulis mengucapkan terima kasih, mudah-mudahan bermanfaat bagi para pembaca.

Sidoarjo, September 2020
Penulis

DAFTAR ISI

BATANG TUBUH DAN
SUB-CAPAIAN PEMBELAJARAN MATA KULIAH

BAB	Sub-Capaian Pembelajaran Mata Kuliah
BAB I PENDAHULUAN	Mahasiswa mampu memahami deskripsi mata kuliah Rekayasa Perangkat Lunak
BAB II KONSEP DASAR RPL	1. Mahasiswa mampu menjelaskan definisi rekayasa perangkat lunak 2. Mahasiswa mampu menjelaskan tujuan rekayasa perangkat lunak 3. Mahasiswa mampu menjelaskan ruang lingkup rekayasa perangkat lunak 4. Mahasiswa mampu menjelaskan Jenis perangkat lunak 5. Mahasiswa mampu memahami tanggung Jawab profesional dan etika
BAB III PERENCANAAN PERANGKAT LUNAK	1. Mahasiswa mampu menyusun tujuan perencanaan proyek 2. Mahasiswa mampu menerapkan teknik pengumpulan data 3. Mahasiswa mampu menguraikan ruang lingkup perangkat lunak 4. Mahasiswa mampu menyusun estimasi proyek perangkat lunak
BAB IV SOFTWARE DEVELOPMENT LIVE CYCLE (SDLC)	1. Mahasiswa mampu menjelaskan definisi SDLC 2. Mahasiswa mampu menerapkan model SDLC

BAB V BASIS DATA	1. Mahasiswa mampu menjelaskan definisi basis data 2. Mahasiswa mampu menerapkan ERD 3. Mahasiswa mampu menerapkan studi kasus ERD
BAB VI DATA FLOW DIAGRAM (DFD)	1. Mahasiswa mampu menjelaskan definisi DFD 2. Mahasiswa mampu menjelaskan komponen DFD 3. Mahasiswa mampu menerapkan tingkatan level DFD 4. Mahasiswa mampu menerapkan studi kasus DFD
BAB VII PEMODELAN DAN UML	1. Mahasiswa mampu menjelaskan kompleksitas pengembangan perangkat lunak 2. Mahasiswa mampu menjelaskan pemodelan perangkat lunak 3. Mahasiswa mampu menerapkan UML 4. Mahasiswa mampu menerapkan Use Case Diagram 5. Mahasiswa mampu menerapkan Activity Diagram
BAB VIII MANAJEMEN PROYEK PERANGKAT LUNAK	1. Mahasiswa mampu menerapkan perencanaan proyek 2. Mahasiswa mampu menerapkan pengujian perangkat lunak
BAB IX PEMELIHARAAN PERANGKAT LUNAK	1. Mahasiswa mampu menjelaskan tenik pemeliharaan perangkat lunak 2. Mahasiswa mampu menerapkan Siklus Hidup Pemeliharaan Sistem (SMLC)

BAB I
PENDAHULUAN

1. Deskripsi

Perangkat lunak (*software*) adalah program computer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi terkait analisis kebutuhan, model desain, dan user manual Shalahuddin (2016). *IEEE Computer Society* mendefinisikan Rekayasa Perangkat Lunak (RPL) sebagai penerapan suatu pendekatan yang sistematis, disiplin dan terkuantifikasi atas pengembangan, penggunaan dan pemeliharaan perangkat lunak, serta studi atas pendekatan-pendekatan ini, yaitu penerapan pendekatan engineering atas perangkat lunak. RPL sendiri adalah suatu disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal yaitu analisa kebutuhan pengguna, menentukan spesifikasi dari kebutuhan pengguna, disain, pengkodean, pengujian sampai pemeliharaan sistem setelah digunakan. Dari pengertian ini jelaslah bahwa RPL tidak hanya berhubungan dengan cara pembuatan program komputer. Pernyataan ”semua aspek produksi” pada pengertian di atas, mempunyai arti semua hal yang berhubungan dengan proses produksi seperti manajemen proyek, penentuan personil, anggaran biaya, metode, jadwal, kualitas sampai dengan pelatihan pengguna merupakan bagian dari RPL.

RPL merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunakan mesin. Perangkat lunak banyak dibuat dan pada akhirnya sering tidak digunakan karena tidak memenuhi kebutuhan pelanggan atau bahkan karena masalah non teknis seperti keenggan pemakai perangkat (*user*) untuk mengubah cara kerja dari manual ke

otomatis, atau ketidakmampuan *user* menggunakan komputer. Oleh karena itu, rekayasa perangkat lunak dibutuhkan agar perangkat lunak yang dibuat tidak menjadi perangkat lunak yang tidak terpakai.

RPL lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat kepada pelanggan (*customer*). Adapun ilmu computer lebih fokus pada teori dan konsep dasar perangkat computer. Rekayasa perangkat lunak lebih fokus pada bagaimana membuat perangkat lunak yang memenuhi kriteria berikut:

a. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (*maintainability*)
b. Dapat diandalkan dengan proses bisnis yang dijalankan perubahan yang terjadi (*dependability robust*)
c. Efisien dari segi sumber daya dan penggunaan
d. Kemampuan untuk dipakai sesuai dengan kebutuhan (*usability*)
   Dari kriteria di atas maka perangkat lunak yang baik adalah perangkat lunak yang dapat memenuhi kebutuhan pelanggan (*customer*) atau *user* (pemakai perangkat lunak) atau berorientasi pada pelanggan atau pemakai perangkat lunak, bukan berorientasi pada pembuat atau pengembang perangkat lunak.

2. Kemampuan Akhir

Mahasiswa diharapkan akan dapat memiliki kompetensi sikap, pengetahuan dan ketrampilan yang berkaitan dengan materi yang ditunjukkan pada Gambar 1.1 setelah mempelajari materi pada buku ajar Rekayasa Perangkat Lunak.

Gambar 1.1 Peta Konsep Materi Rekayasa Perangkat Lunak

Bab II
Konsep Rekayasa Perangkat Lunak (RPL)

Setelah mempelajari bab konsep rekayasa perangkat lunak, maka diharapkan :

6. Mahasiswa mampu menjelaskan definisi rekayasa perangkat lunak
7. Mahasiswa mampu menjelaskan tujuan rekayasa perangkat lunak
8. Mahasiswa mampu menjelaskan ruang lingkup rekayasa perangkat lunak
9. Mahasiswa mampu menjelaskan Jenis perangkat lunak
10. Mahasiswa mampu memahami tanggung Jawab profesional dan etika

Bayangkan anda mempunyai sebidang tanah yang akan dibangun rumah. Lalu pertanyaan apa yang muncul pertama kali?

"Bagaimana proses pembangunan rumah anda ?"

a. Jika anda memulai membangun dengan cepat ? (hanya dibantu oleh anak anda yang berumur 14 tahun)
b. Jika anda pergi ke sembarang pengembang
c. Jika Anda mempekerjakan seorang arsitek untuk mendesain dari awal

"Apakah yang akan dihasilkan ?"

Lalu, Bagaimana dengan membangun perangkat lunak ?

Software development biasanya akan melakukan hal yang sama etika mendapatkan persoalan sederhana yang membutuhkan solusi komputasi: *Berfikir sejenak, menghadap komputer dan kemudian mulai mengetikkan baris demi baris code. Tidak ada kertas-kertas yang memuat perancangan aristektur dan algoritma secara rinci, karena semua rancangan itu ada di dalam kepala*.

Oleh karena itu kita memerlukan **Rekayasa Perangkat Lunak**

1. Definisi Rekayasa Perangkat Lunak

IEEE-Standar Glossary of Software Engineering Terminology, 1990: *"Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system".* Istilah Reakayasa Perangkat Lunak (RPL) secara umum disepakati sebagai terjemahan dari istilah Software engineering. Istilah *Software Engineering* mulai dipopulerkan pada tahun 1968 pada *software engineering Conference* yang diselenggarakan oleh NATO. Sebagian orang mengartikan RPL hanya sebatas pada bagaimana membuat program komputer. Padahal ada perbedaan yang mendasar antara perangkat lunak (*software*) dan program komputer. Perangkat lunak merupakan kumpulan dari program, prosedur, dan dokumen data lain yang saling berhubungan yg merepresentasikan masalah di dunia nyata yang dikonfigurasikan dalam sebuah bentuk aplikasi yang harus dikerjakan komputer. Perangkat lunak adalah seluruh perintah yang digunakan untuk memproses informasi. Perangkat lunak dapat berupa program atau prosedur. Program adalah kumpulan perintah yang dimengerti oleh komputer sedangkan prosedur adalah perintah yang dibutuhkan oleh pengguna dalam memproses informasi (O'Brien, 1999).

Berdasarkan kajian di atas dapat disimpulkan, RPL merupakan disiplin ilmu yg membahas semua aspek produksi perangkat lunak, mulai dari tahap awal spesifikasi sistem sampai pemeliharaan sistem setelah digunakan. Perangkat Lunak yang dibuat harus mampu: tepat waktu, tepat anggaran, meningkatkan kinerja, mengoperasikan prosedur sistem dengan benar. Rekayasa perangkat lunak lebih fokus pada praktik pengembangan perangkat lunak dan mengirimkan perangkat lunak yang bermanfaat untuk pelanggan. Sehinggaperangkat lunak yang baik adalah perangkat lunak yang

dapat memenuhi kebutuhan pelanggan atau user dengan kata lain berorientasi pada pelanggan atau pemakai perangkat lunak, **bukan** berorientasi pada pembuat atau pengembang.

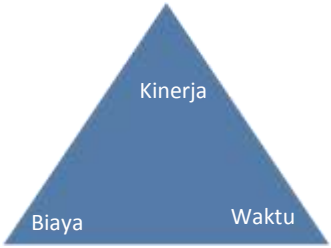
Pekerjaan yang terkait dengan rekayasa perangkat dapat dikategorikan menjadi tiga buah kategori umum tanpa melihat area dari aplikasi, ukuran proyek perangkat lunak, atau kompleksitas perangkat lunak yang akan dibuat. Setiap fase dialamatkan pada satu atau lebih pertanyaan yang diajukan sebelumnya.

Fase pendefinisian fokus pada "*what*" yang artinya harus mencari tahu atau mengidentifikasi informasi apa yang harus diproses, seperti apa fungsi dan performansi yang diinginkan, seperti apa perilaku yang diinginkan, apa kriteria validasi yang dibutuhkan untuk mendefinisikan sistem. Fase pengembangan yang fokus dengan "*how*" yang artinya selama tahap Pengembangan perangkat lunak seorang perekayasa perangkat lunak (*software engineer*) berusaha mendefinisikan bagaimana data distrukturkan dan bagaimana fungsi-fungsi yang dibutuhkan diimplementasikan di dalam arsitektur perangkat lunak, bagaimana detail procedural diimplementasikan, bagaimana karakter antarmuka tampilan, bagaimana desain ditranslasikan ke bahasa pemrograman, dan bagaimana pengujian akan dijalankan.

Fase pendukung (*support phase*) fokus pada perubahan terasosiasi pada perbaikan kesalahan (*error*), adaptasi yang dibutuhkan pada lingkungan perangkat lunak yang terlibat, dan perbaikan yang terjadi akibat perubahan kebutuhan pelanggan (*customer*). Fase pendukung terdiri dari empat tipe perubahan antara lain, koreksi (*correction*), adaptasi (*adaptation*), perbaikan (*enhancement*), dan pencegahan (*prevention*).

2. Tujuan Rekayasa Perangkat Lunak

Bidang rekayasa akan selalu berusaha menghasilkan output yang kinerjanya tinggi, biaya rendah dan waktu penyelesaian yang tepat. Tujuan RPL dapat ditunjukkan pada Gambar 1.



Gambar 1. Tujuan RPL

Secara lebih khusus kita dapat menyatakan tujuan RPL adalah:

a. Memperoleh biaya produksi perangkat lunak yang rendah.
b. Menghasilkan perangkat lunak yang kinerjanya tinggi, andal dan tepat waktu.
c. Menghasilkan perangkat lunak yang dapat bekerja pada berbagai jenis platform.
d. Menghasilkan perangkat lunak yang biaya perawatannya rendah.

3. Ruang Lingkup Rekayasa Perangkat Lunak

Berdasarkan definisi dari para ahli yang tekah dibahas sebelumnya, maka ruang lingkup RPL dapat ditunjukkan di Gambar 2.

Gambar 2. Ruang Lingkup RPL

a. Software requirements berhubungan dengan spesifikasi kebutuhan dan persyaratan perangkat lunak.
b. Software design mencakup proses penentuan arsitektur, komponen, antarmuka, dan karakteristik lain dari perangkat lunak.
c. Software construction berhubungan dengan detil pengembangan perangkat lunak, termasuk algoritma, pengkodean, pengujian, dan pencarian kesalahan.
d. Software testing meliputi pengujian pada keseluruhan perilaku perangkat lunak.
e. Software maintenance mencakup upaya-upaya perawatan ketika perangkat lunak telah dioperasikan.
f. Software configuration management berhubungan dengan usaha perubahan konfigurasi perangkat lunak untuk memenuhi kebutuhan tertentu.
g. Software engineering management berkaitan dengan pengelolaan dan pengukuran RPL, termasuk perencanaan proyek perangkat lunak.
h. Software engineering tools and methods mencakup kajian teoritis tentang alat bantu dan metode RPL.
i. Software engineering process berhubungan dengan definisi, implementasi, pengukuran, pengelolaan, perubahan dan perbaikan proses RPL.
j. Software quality menitikberatkan pada kualitas dan daur hidup perangkat lunak.

4. Jenis Perangkat Lunak

Terdapat beberapa jenis perangkat lunak sesuai dengan ruang lingkupnya, diantaranya:

a. *Software Requirements* berhubungan dengan spesifikasi kebutuhan dan persyaratan perangkat lunak
b. *Software* Desain mencakup proses penampilan arsitektur, komponen, antar muka, dan karakteristik lain dari perangkat lunak
c. *Software Construction* berhubungan dengan detail pengembangan perangkat lunak, termasuk algoritma, pengkodean, pengujian dan pencarian kesalahan
d. *Software Testing* meliputi pengujian pada keseluruhan perilaku perangkat lunak
e. *Software Maintenance* mencakup upaya-upaya perawatan ketika perangkat lunak telah dioperasikan
f. *Software Configuration Management* berhubungan dengan usaha perubahan konfigurasi perangkat lunak untuk memenuhi kebutuhan tertentu
g. *Software Engineering management* berkaitan dengan pengelolaan dan pengukuran RPL, termasuk perencanaan proyek perangkat lunak
h. *Software Engineering Tools And Methods* mencakup kajian teoritis tentang alat bantu dan metode RPL
i. *Software Engineering Process* berhubungan dengan definisi, implementasi pengukuran, pengelolaan, perubahan dan perbaikan proses RPL
j. *Software Quality* menitik beratkan pada kualitas dan daur hidup perangkat lunak

Jenis software berdasar lisensi, terdapat 2 jenis yaitu ***open source dan proprietary***. Pertama yaitu ***open source software***, *Software* yang *source codenya* terbuka dan didistribusikan dalam suatu format lisensi yang memungkinkan pihak lain secara bebas memperbanyak dan memodifikasi *source code* (informasi)

didalamnya. Pada open source software ini, kepemilikan hak cipta tetap ada, akan tetapi lisensi memungkinkan dapat digunakan oleh orang lain serta dapat memodifikasi softwarenya. Jenis-jensi lisensi *open source software* diantaranya BSD license, Apache License, GNU General Public License (GPL), Mozilla Public License, dan MIT License.

Kedua yaitu ***proprietary software***, merupakan *Software* yang *source* codenya tertutup dan didistribusikan dengan suatu format lisensi yang membatasi pihak lain untuk menggunakan, memperbanyak dan memodifikasi. Lisensi proprietary software memungkinkan orang lain menggunakan software yang kita buat dengan diikuti penyerahan royalti (uang) ke pemilik hak ciptanya.

Perangkat lunak berdasar fungsionalnya, yaitu : *interfacing*, *operating sistem*, dan program aplikasi. ***Interfacing*** yaitu perangkat lunak yang menghubungkan suatu perangkat keras tertentu, seperti hardware driver, interfaces dengan perangkat keras lain. Contoh : Driver untuk Kamera, *Handphone* atau perangkat keras lainnya, Program interface seperti Sensor Suhu dengan LM 555, PPI 8255, Komunikasi Serial RS 232. ***Operating system***, Perangkat lunak yang menjalankan sistem komputer dan merupakan *interface* dari sistem komputer dan program aplikasi yang berjalan diatasnya. Beberapa OS yang dikenal secara luas: Microsoft Windows, Linux dan variansnya, seperti Redhat, SuSE, Mandrake, Debian, Unix, FreeBSD, *Macintosh (Apple*). **Program Aplikasi**, yaitu perangkat lunak yang digunakan program ini digunakan untuk keperluan tertentu, yang tujuannya membantu pekerjaan manusia menjadi lebih mudah. Program ini yang banyak dibahas dalam pembuatan perangkat lunak. Program Aplikasi ini tergantung pada kebutuhan dari program itu sendiri, seperti*: Program Office*, *Program Graphics Design*, *Program Multimedia*, dan lain-lain.

Karakteristik perangkat lunak, diantaranya : Mempunyai daya guna yang tinggi (usability), Mempunyai kinerja sesuai fungsi yang

dibutuhkan pemakai, Mampu diandalkan (be reliable), Mudah dirawat/diperbaiki (*maintenability*), Lebih efisien, Mempunyai antarmuka yg menarik (*eye cathcing user interface*), Mempunyai siklus hidup yang cukup lama (*long life time*).

5. Tanggung Jawab Profesional Dan Etika

RPL melibatkan tanggung jawab yang lebih besar dari sekedar penerapan keahlian teknis. Rekayasawan perangkat lunak harus berlaku secara jujur dan etis jika ingin dihargai sebagai profesional. Perilaku etis lebih dari sekedar menjunjung tinggi hukum. Tanggung jawab profesional yang pertama adalah **kerahasiaan**, Rekayasawan harus menghargai kerahasiaan pegawai atau kliennya. Kedua, **Kompeten**, Rekayasawan tidak boleh memberi gambaran yang salah tentang tingkat kompetensinya, mereka tidak boleh secara sadar menerima pekerjaan yang diluar kompetensinya.

Perangkat lunak yang baik harus memberikan fungsi yang diperlukan dan kinerja bagi pengguna dan harus dipertahankan, diandalkan dan dapat diterima. ***Maintainability*** (mudah dipelihara/dirawat) *Software* harus berevolusi untuk memenuhi perubahan kebutuhan, ***Dependability*** (dapat diandalkan) Software harus dapat dipercaya, ***Efficiency*** (Daya Guna) Efisien dalam penggunaan sumber daya sistem (memori, hardware, listrik, dll), ***Acceptability*** (Dapat diterima) Perangkat Lunak harus diterima oleh pengguna seperti yang pernah dirancang. Ini berarti harus bisa dimengerti, digunakan dan kompatibel dengan sistem lainnya.

Tantangan utama yang dihadapi RPL, yaitu: (1*)* ***Heterogeneity*** (keberagaman), mengembangkan teknik untuk membangun perangkat lunak yang dapat mengatasi perbedaan platform dan lingkungan eksekusi; (2) *Delivery* (pengiriman), mengembangkan teknik yang mengakibatkan pengiriman perangkat lunak lebih cepat;

(3*) Trust* (kepercayaan), mengembangkan teknik yang menunjukkan bahwa perangkat lunak dapat dipercaya oleh para penggunanya.

- **Kesimpulan**

RPL merupakan disiplin ilmu yang membahas semua aspek produksi perangkat lunak, mulai dari tahap awal spesifikasi sistem sampai pemeliharaan sistem setelah digunakan. Perangkat Lunak yang dibuat harus mampu: tepat waktu, tepat anggaran, meningkatkan kinerja, mengoperasikan prosedur sistem dengan benar. Rekayasawan perangkat lunak harus berlaku secara jujur dan etis jika ingin dihargai sebagai profesional.

- **Evaluasi**

Setelah menyimak materi konsep rekayasa perangkat lunak, silahkan jawab pertanyaan berikut:

1. Apakah proses produksi perangkat lunak identik atau serupa dengan proses produksi pada pabrik/manufaktur pembuatan mobil? Jelaskan alasannya!
2. Bidang rekayasa perangkat lunak apakah sebagai bagian dari seni atau bagian dari teknik? Jelaskan alasannya!
3. Mengapa ada proses-proses atau tahapan-tahapan yang harus dilakukan dalam rekayasa perangkat lunak?
4. Mengapa rekayasa perangkat lunak sebaiknya fokus pada pelanggan atau pengguna?
5. Mengapa faktor sosial dari tekologi informasi sering sekali diabaikan oleh pengembang aplikasi?

- **Daftar Pustaka**

Sukamto, Rosa A., & Shalahuddin, M. 2016. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object*. Bandung: Informatika.

Kadir dan Triwahyuni. 2013. *Pengantar Teknologi Informasi*. Yogyakarta: Andi.

Pressman, R.S. 2012. Rekayasa Perangkat Lunak: Pendekatan Praktisi. Yogyakarta: Andi.

Pressman, Roger S., 2001, *Software Engineering A Practitioner's Approach 7th ed*, McGraw-Hill, New York.

Bab III
Perencanaan Perangkat Lunak

Setelah mempelajari bab **Perencanaan Perangkat Lunak**, maka diharapkan :

5. Mahasiswa mampu menyusun tujuan perencanaan proyek
6. Mahasiswa mampu menerapkan teknik pengumpulan data
7. Mahasiswa mampu menguraikan ruang lingkup perangkat lunak
8. Mahasiswa mampu menyusun estimasi proyek perangkat lunak

Proses pengembangan perangkat lunak dimulai dengan beberapa aktivitas yang secara kolektif disebut dengan project planning (perencanaan proyek). Perencanaan proyek memberikan sebuah peta jalan bagi suksesnya rekayasa perangkat lunak.

1. Tujuan Perencanaan Proyek

Tujuan perencanaan proyek perangkat lunak adalah untuk menyediakan sebuah kerangka kerja yang memungkinkan manajer membuat estimasi yang dapat dipertanggungjawabkan mengenai sumber daya, biaya dan jadwal. Tujuan perencanaan dicapai melalui suatu proses penemuan informasi yang menunjuk ke estimasi yang dapat dipertanggungjawabkan.

2. Teknik Pengumpulan Data

Beberapa teknik pengumpulan data yang dilakukan pada saat merencanaan proyek perangkat lunak, yaitu teknik wawancara, teknik observasi, dan teknik kuisioner.

a. Wawancara

Dengan wawancara, lebih mudah dalam menggali bagian sistem mana yang lebih baik, dapat menggali kebutuhan user secara lebih luas, User dapat mengungkapkan kebutuhan lebih bebas. Panduan dalam melakukan wawancara yaitu: Buat jadwal wawancara dengan narasumber, Panduan wawancara, Pertanyaan yang jelas, Catat hasil wawancara.

Pengumpulan data dengan menggunakan wawancara mempunyai beberapa keuntungan sebagai berikut.

1) Lebih mudah dalam menggali bagian sistem mana yang dianggap baik dan bagian mana yang dianggap kurang baik
2) Jika da bagian tertentu yang menurut anda perlu untuk digali lebih dalam, anda dapat langsung menanyakan kepada narasumber
3) Dapat menggali kebutuhan *user* secara lebih bebas
4) *User* dapat mengungkapkan kebutuhannya secara lebih bebas

Selain mempunyai beberapa kelebihan tersebut, teknik wawancara juga memiliki kelemahan. Berikut ini adalah beberapa kelemahan dari teknik wawancara.

1) Wawancara akan sulit dilakukan jika narasumber kurang dapat mengungkapkan kebutuhannya
2) Pertanyaan dapat menjadi tidak terarah, terlalu fokus pada hal-hal tertentu dan mengabaikan bagian lainnya

b. Observasi

Observasi dilakukan dengan melakukan Analisis dapat melihat langsung bagaimana sistem lama berjalan. Dengan observasi mampu menghasilkan gambaran lebih baik jika dibandingkan dengan teknik lain. Panduan dalam melakukan observasi, yaitu Tentukan hal-hal apa saja yang akan diobservasi, minta ijin pada yang berwenang pada bagian yang akan diobservasi, jika ada yang anda tidak mengerti, coba bertanya jangan berasumsi sendiri.

Pengumpulan data dengan menggunakan observasi mempunyai keuntungan, yaitu:

1) Analisis dapat melihat langsung bagaimana sistem lama berjalan
2) Mampu menghasilkan gambaran lebih baik jika dibanding dengan teknik lainnya

Kelemahan dengan menggunakan teknik observasi, yaitu :

1) Membutuhkan waktu cukup lama karena jika observasi waktunya sangat terbatas, maka gambaran sistem secara keseluruhan akan sulit untuk diperoleh
2) Orang-orang yang sedang diamati biasanya perilakunya akan berbeda dengan perilaku sehari-hari (cenderung berusaha terlihat baik).
3) Dapat mengganggu pekerjaan orang-orang pada bagian yang sedang diamati

c. Kuisioner

Kuisioner dilakukan dengan harapan Hasil lebih objektif karena dapat dilakukan pada banyak orang sekaligus, waktu lebih singkat. Panduan ketika akan membagikan kuisioner, yaitu : hindari pertanyaan isian, buat pertanyaan yang tidak terlalu banyak, buat pertanyaan yang singkat padat dan jelas.

Pengumpulan data dengan menggunakan kuesioner mempunya keuntungan, yaitu :

1) Hasilnya lebih obyektif, karena kuesioner dapat dilakukan kepada banyak orang sekaligus
2) Waktunya lebih singkat

Kelemahan pengumpulan data dengan menggunakan kuesioner adalah sebagai berikut:

1) Responden cenderung mals untuk mengisi kuesioner
2) Sulit untuk membuat pertanyaan yang singkat, padat, jelas dan mudah dipahami

3. Ruang Lingkup Perangkat Lunak

Penentuan ruang lingkup perangkat lunak merupakan aktivitas pertama dalam perencanaan proyek perangkat lunak. Ruang lingkup perangkat lunak menggabarkan fungsi, kinerja, batasan, interface dan reliabilitas. Fungsi yang digambarkan dalam statmen ruang lingkup dievaluasi dan disaring untuk memberikan awalan yang lebih detail pada saat estimasi dimulai. Pertimbangan kinerja melingkupi pemrosesan dan kebutuhan waktu respon. Batasan ini mengidentifikasi dari batas yang ditempatkan pada perangkat lunak oleh perangkat keras eksternal, memori, atau sistem informasi yang ada.

Teknik yang banyak dipakai secara umum untuk menjembatani jurang komunikasi antara pelanggan dan pengembang serta untuk memulai proses komunikasi adalah dengan melakukan pertemuan atau wawancara pendahuluan. Gause & weinberg mengusulkan bahwa analis harus memulai dengan mengajukan pertanyaan-pertanyaan bebas konteks, yaitu serangkaian pertanyaan yang akan membawa pada pemahaman mendasar terhadap masalah, orang yang menginginkan suatu solusi, sifat solusi yang diharapkan, dan efektivitas pertemuan itu. Beberapa pertanyaan bebas konteks pada pelanggan yang meliputi tujuan keseluruhan, serta keuntungan :

a. Siapa di belakang permintaan kerja ini?
b. Siapa yang akan memakai solusi ini?
c. Apakah yang akan menjadi keutungan ekonomi dari sebuah solusi yang sukses?
d. Adakah sumberdaya lain bagi solusi ini?

4. Estimasi Proyek Perangkat Lunak

Estimasi tidak akan pernah menjadi ilmu pasti, disebabkan banyaknya variable (manusia, teknik, lingkungan dan politik) yang

mempengaruhi biaya dan usaha akhir yang diaplikasikan untuk mengembangkannya. Beberapa pilihan untuk mencapai estimasi :

a. Menunda estimasi sampai akhir proyek
b. Mendasarkan estimasi pada proyek – proyek yang mirip yang sudah dilakukan
c. Menggunakan teknik dekomposisi yang relatif sederhana
d. Menggunakan satu atau lebih model empiris bagi estimasi usaha dan biaya perangkat

Kesimpulan

Perencanaan proyek memberikan sebuah peta jalan bagi suksesnya rekayasa perangkat lunak. Beberapa teknik pengumpulan data yang dilakukan pada saat merencanaan proyek perangkat lunak, yaitu teknik wawancara, teknik observasi, dan teknik kuisioner. Ruang lingkup perangkat lunak menggabarkan fungsi, kinerja, batasan, interface dan reliabilitas.

Evaluasi

Setelah menyimak materi perancanaan perangkat lunak, silahkan jawab pertanyaan berikut:

1. Kegiatan apa saja yang dilakukan pada saat perencanaan proyek?
2. Teknik yang paling sesuai untuk menjembatani komunikasi dengan user adalah? Jelaskan alasannya!
3. Sebutkan jenis-jenis kebutuhan pengembangan sistem informasi!
4. Hal-hal apa saja yang dilakukan pada tahap desain sistem?

Daftar Pustaka

Kendall, J.E. & Kendall, K.E. 2010. Analisis dan Perancangan Sistem. Jakarta: Indeks.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Sukamto, Rosa A., & Shalahuddin, M. 2016. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object*. Bandung: Informatika

Pressman, R.S. 2012. Rekayasa Perangkat Lunak: Pendekatan Praktisi. Yogyakarta: Penerbit Andi.

Pressman, Roger S., 2001, *Software Engineering A Practitioner's Approach 7th ed*, McGraw-Hill, New York.

Bab IV
Software Development Life Cycle (SDLC)

Setelah mempelajari bab ***Software Development Life Cycle*** **(SDLC)**, maka diharapkan :

3. Mahasiswa mampu menjelaskan definisi SDLC
4. Mahasiswa mampu menerapkan model SDLC

1. Definisi SDLC

SDLC *atau Software Development Life Cycle* merupakan proses pengembangan atau mengubah suatu system perangkat lunak dengan menggunakan atau mengubah suatu system perangkat lunak dengan menggunakan model-model dan metodologi yang digunakan orang untuk mengembangkan system perangkat lunak. SDLC juga merupakan pola yang diambil untuk mengembangkan sistem perangkat lunak, yang terdiri dari tahap-tahap: rencana(*planning*), analisis (*analysis*), desain (*design*), implementasi (*implementation*), uji coba (*testing)* dan pengelolaan (*maintenance*).

Gambar 4.1. Model siklus pengembangan sistem

Dalam rekayasa perangkat lunak, konsep SDLC mendasari berbagai jenis metodologi pengembangan perangkat lunak. Metodologi-metodologi ini membentuk suatu kerangka kerja untuk perencanaan dan pengendalian pembuatan sistem informasi, yaitu

proses pengembangan perangkat lunak. Terdapat beberapa model SDLC yang dapat digunakan, semuanya mempunyai kelebihan dan kekurangan masing-masing pada tiap tahapannya. Hal yang paling penting yaitu mengenali type pelanggan/ customer dan memilih menggunakan model SDLC yang sesuai dengan karakter pelanggan dan sesuai dengan karakter pengembang.

2. Model SDLC

a. Model Waterfall

Model air terjun (Waterfall Model) adalah pendekatan klasik dalam pengembangan perangkat lunak yang menggambarkan metode pengembangan linier dan berurutan. Ini terdiri dari lima hingga tujuh fase, setiap fase didefinisikan oleh tugas dan tujuan yang berbeda, di mana keseluruhan fase menggambarkan siklus hidup perangkat lunak hingga pengirimannya. Setelah fase selesai, langkah pengembangan selanjutnya mengikuti dan hasil dari fase sebelumnya mengalir ke fase berikutnya.

Gambar 4.2 Model Waterfall

1) *Requirement Gathering and analysis* — Mengumpulkan kebutuhan secara lengkap kemudian dianalisis dan didefinisikan kebutuhan yang harus dipenuhi oleh program yang akan dibangun. Fase ini harus dikerjakan secara lengkap untuk bisa menghasilkan desain yang lengkap.
2) Desain ,dalam tahap ini pengembang akan menghasilkan sebuah sistem secara keseluruhan dan menentukan alur perangkat lunak hingga algoritma yang detail.
3) Implementasi adalah Tahapan dimana seluruh desain diubah menjadi kode kode progam . Kode progam yang dihasilkan masih berupa modul-modul yang akan diintregasikan menjadi sistem yang lengkap.
4) *Integration & Testing*, Di tahap ini dilakukan penggabungan modul-modul yang sudah dibuat dan dilakukan pengujian ini dilakukan untuk mengetahui apakah software yang dibuat telah sesuai dengan desainnya dan fungsi pada software terdapat kesalahan atau tidak.
5) Verifikasi adalah klien atau pengguna menguji apakah sistem tersebut telah sesuai dengan yang disetujui.
6) *Operation & Maintenance* yaitu instalasi dan proses perbaikan sistem sesuai yang disetujui.

Keunggulan Model pendekatan pengembangan software metode waterfall adalah pencerminan kepraktisan rekayasa , yang bisa membuat kualitas software tetap terjaga. Jenis model yang bersifat lengkap sehingga proses pemeliharaannya lebih mudah. Karena struktur logis dari model, kesalahan konseptual seringkali dapat dihindari. Model ini mengarah pada dokumentasi teknis yang luas, yang merupakan kelegaan bagi programmer dan pengembang baru dan juga berguna dalam tahap pengujian. Kemajuan proyek

dapat dipantau menggunakan tonggak sejarah. Total biaya dapat diperkirakan dengan akurasi relatif jika tidak ada konflik.

Kelemahan model waterfall ini adalah lambatnya proses pengembangan perangkat lunak. Dikarenakan proses yang satu tidak bisa diloncat-loncat maka dari itu model ini sangat memakan waktu dalam mengembangkannya. Kelemahan yang lain kinerja tidak optimal dan efisien. Konflik, bug, dan kesalahan program terkadang menyebabkan kenaikan biaya dan waktu yang cukup lama. Hal yang sama berlaku jika klien tidak puas. Spesifikasi yang awalnya dibuat seringkali sulit untuk dipahami oleh klien karena lebih abstrak daripada apa yang seharusnya dilakukan oleh perangkat lunak. Terutama dalam proyek-proyek outsourcing, ini bisa menjadi kerugian yang menentukan, karena tanggal rilis harus ditunda dan pasar mungkin telah berubah selama waktu ini.

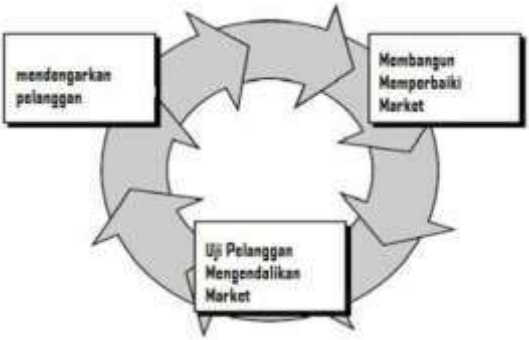
Model waterfall adalah model SDLC yang paling sederhana, model ini hanya cocok untuk pengembangan perangkat lunak dengan spesifikasi yang tidak berubah-ubah.

b. Model Prototype

Prototyping adalah proses merancang sebuah prototype dimana prototype sendiri adalah sebuah model dari sebuah model produk yang mungkin belum memiliki semua fitur produk sesungguhnya namun sudah memiliki fitur – fitur utama dari produk sesungguhnya dan biasa digunakan untuk keperluan testing/uji coba untuk bahan uji coba sebelum berlanjut ke fase pembuatan produk sesungguhnya. Dengan metode prototyping ini pengembang dan pelanggan dapat saling berinteraksi selama proses pembuatan suatu produk.

Prototyping perangkat lunak adalah salah satu metode siklus hidup sistem yang didasarkan pada konsep model bekerja (working model). Tujuannya adalah mengembangkan model menjadi sistem final. Artinya sistem akan dikembangkan lebih cepat dari pada

metode tradisional dan biayanya menjadi lebih rendah. Ada banyak cara untuk melakukan prototyping, begitu pula dengan penggunaanya.



Gambar 4.3 Ilustrasi Model Prototype menurut Roger S. Pressman

Tahapan pengembangan model *prototype* yaitu:

1) Mendengarkan pelanggan, pada tahap ini dilakukan pengumpulan kebutuhan dari system dengan cara mendengar keluhan dari pelanggan. Untuk membuat suatu system yang sesuai kebutuhan, maka harus diketahui terlebih dahulu bagaimana system yang sedang berjalan untuk kemudian mengetahui masalah yang terjadi.
2) Merancang dan Membuat *Prototype*, pada tahap ini, dilakukan perancangan dan pembuatan prototype system. Prototype yang dibuat disesuaikan dengan kebutuhan system yang telah didefinisikan sebelumnya dari keluhan pelanggan atau pengguna.
3) Uji coba Pada tahap ini, *Prototype* dari system di uji coba oleh pelanggan atau pengguna. Kemudian dilakukan evaluasi kekurangan-kekurangan dari kebutuhan pelanggan. Pengembangan kemudian kembali mendengarkan keluhan dari pelanggan untuk memperbaiki Prototype yang ada.

Kelebihan model *prototype,* Adanya komunikasi yang baik antara pengembang dan pelanggan. Pengembangan dapat bekerja lebih baik dalam menentukan kebutuhan pelanggan. Lebih menghemat waktu dalam pengembangan system. Penerapan menjadi lebih mudah karena pemakai mengetahui apa yang diharapkannya. Pelanggan ikut dalam pengembangan sistem yang akan memudahkan pengembang mengetahui produk yang diharapkan pelanggan.

Kekurangan model *prototype* diantaranya Resiko tinggi yaitu untuk masalah-masalah yang tidak terstruktur dengan baik, ada perubahan yang besar dari waktu ke waktu, dan adanya persyaratan data yang tidak menentu. Interaksi pemakai penting. Sistem harus menyediakan dialog on-line antara pelanggan dan komputer. Hubungan pelanggan dengan komputer yang disediakan mungkin tidak mencerminkan teknik perancangan yang baik. Kurang fleksibel jika terjadi perubahan. Walaupun pemakai melihat berbagai perbaikan dari setiap versi prototype, tetapi pemakai mungkin tidak menyadari bahwa versi tersebut dibuat tanpa memperhatikan kualitas dan pemeliharaan jangka panjang.

Model *prototype* sesuai jika diterapkan untuk menggali spesifikasi kebutuhan pelanggan secara lebih detal tetapi beresiko tinggi terhadap membengkaknya biaya dan waktu proyek.

c. Model Rapid Application Development (RAD)

Metode Rapid Application Development (RAD) adalah model pengembangan perangkat lunak yang pengembangannya tergolong dalam teknik incremental (bertingakat). *Rapid Application Development* (RAD) adalah strategi siklus hidup yang ditujukan untuk menyediakan pengembangan yang jauh lebih cepat dan mendapatkan hasil dengan kualitas yang lebih baik dibandingkan dengan hasil yang dicapai melalui siklus tradisional (McLeod, 2002).

Dalam proses pengembangannya model ini menggunakan metode iterative atau berulang dimana working model sistem dikonstruksikan di awal tahap pengembangan dengan tujuan menetapkan kebutuhan (requirement) user. Model ini membutuhkan waktu singkat dibanding metode lain yaitu sekitar 30-90 hari. Siklus kerja nya yang pendek juga membuat pengembangan sistem dengan model ini bisa diselesaikan dengan cepat. Tetapi terdapat kekurangan dari penggunaan model RAD sehingga kurang efektif penerapannya untuk proyek besar.

Pemaparan konsep yang lebih spesifik lagi dijelaskan oleh Pressman (2005) dalam bukunya, "*Software Engineering: A Practition's Approach*". Ia mengatakan bahwa RAD adalah proses model perangkat lunak inkremental yang menekankan siklus pengembangan yang singkat. Model RAD adalah sebuah adaptasi "kecepatan tinggi" dari model waterfall, di mana perkembangan pesat dicapai dengan menggunakan pendekatan konstruksi berbasis komponen. Jika tiap-tiap kebutuhan dan batasan ruang lingkup projek telah diketahui dengan baik, proses RAD memungkinkan tim pengembang untuk menciptakan sebuah "sistem yang berfungsi penuh" dalam jangka waktu yang sangat singkat.

Gambar 4.4 Ilustrasi model RAD menurut Kendal 2010

RAD digunakan pada aplikasi sistem konstruksi, maka menekankan fase-fase. Ada tiga fase dalam RAD yaitu (kendall dan kendall, 2008) :

1) *Requirement planning*, dalam tahap ini diketahui apa saja yan menjadi kebutuhan sistem yaitu dengan mengidentifikasikan kebutuhan informasi dan masalah yang dihadapi untuk menentukan tujuan, batasan-batasan sistem, kendala dan juga alternatif pemecahan masalah. Analisis digunakan untuk mengetahui perilaku sistem dan juga untuk mengetahui aktivitas apa saja yang ada dalam sistem tersebut.
2) *Design workshop*, yaitu mengidentifikasi solusi alternatif dan memilih solusi yang terbaik. Kemudian membuat desain proses bisnis dan desain pemrograman untuk data-data yang telah didapatkan dan dimodelkan dalam arsitektur sistem informasi. Tools yang digunakan dalam pemodelan sistem biasanya menggunakan unified modeling language (uml).
3) *Implentation,* setelah design workshop dilakukan, selanjutnya sistem diimplementasikan (coding) ke dalam bentuk yang dimengerti oleh mesin yang diwujudkan dalam bentuk program atau unit program. Tahap implementasi sistem merupakan tahap meletakkan sistem supaya siap untuk dioperasikan.

Kelebihan Metode *Rapid Application Development* (RAD), Sangat berguna dilakukan pada kondisi user tidak memahami kebutuhan kebutuhan apa saja yang digunakan pada proses pengembangan perangkat lunak. Metode RAD dapat dilakukan dengan waktu yang singkat yakni sekitar 30-90 hari. Biaya yang dikeluarkan menjadi lebih rendah karena waktu pengembangan yang singkat dan menggunakan komponen yang sudah ada. Proses pengiriman menjadi lebih mudah, hal ini dikarenakan proses pembuatan lebih banyak menggunakan potongan-potongan script.

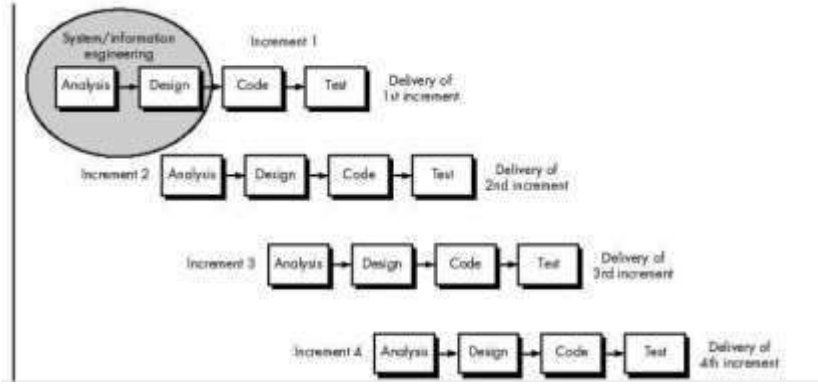
Mudah untuk diamati sehingga user lebih mengerti akan sistem yang dikembangkan. Lebih fleksibel karena pengembang dapat melakukan proses desain ulang pada saat yang bersamaan. Bisa mengurangi penulisan kode yang kompleks karena menggunakan wizard. Keterlibatan user semakin meningkat karena merupakan bagian dari tim secara keseluruhan. Mampu meminimalkan kesalahan-kesalahan dengan menggunakan alat-alat bantuan (*case tools*). Mempercepat waktu pengembangan sistem secara keseluruhan karena cenderung mengabaikan kualitas. Tampilan yang lebih standar dan nyaman dengan bantuan *software-software* pendukung.

Kekurangan *Rapid Application Development*, Beberapa hal yang perlu di perhatikan dalam implementasi pengembangan menggunakan model RAD : Membutuhkan SDM atau sumber daya manusia yang ahli untuk proyek berskala besar. RAD menuntut pengembang dan pelanggan memiliki komitmen dalam aktivitas *rapid fire* yang diperlukan untuk melengkapi sebuah sistem dalam waktu yang singkat. Jika komitmen tersebut tidak ada maka proyek RAD akan gagal. Dengan melakukan pembelian belum tentu bisa menghemat biaya dibandingkan dengan mengembangkan sendiri. Membutuhkan biaya tersendiri untuk membeli peralatan-peralatan penunjang seperti misalnya *software* dan *hardware*. Kesulitan melakukan pengukuran mengenai kemajuan proses. Kurang efisien karena apabila melakukan pengkodean dengan menggunakan tangan bisa lebih efisien. Ketelitian menjadi berkurang karena tidak menggunakan metode yang formal dalam melakukan pengkodean. Lebih banyak terjadi kesalahan apabila hanya mengutamakan kecepatan dibandingkan dengan biaya dan kualitas. Fasilitas-fasilitas banyak yang dikurangi karena terbatasnya waktu yang tersedia. Sistem sulit diaplikasikan di tempat yang lain. Fasilitas yang tidak perlu terkadang harus disertakan, karena menggunakan komponen yang sudah jadi, sehingga hal ini membuat biaya semakin meningkat.

d. Model Iteratif

Metode yang merupakan pengembangan dari *prototyping* model dan digunakan ketika requirement dari software akan terus berkembang dalam tahapan-tahapan pengembangan aplikasi tersebut. Sedikit pengertian tentang requirement software dari developer yang diterapkan pada tahap pertama iterasi, akan mendapatkan tanggapan dari user. Ketika requirement menjadi jelas, tahapan iterasi selanjutnya akan dilaksanakan.

Pada setiap iterasi, modifikasi desain yang dibuat dan kemampuan fungsional baru ditambahkan. Ide dasar di balik metode ini adalah untuk mengembangkan sistem melalui siklus berulang (iteratif) dan dalam porsi yang lebih kecil pada waktu (incremental).



Gambar 4.5 Ilustrasi model Iteratif

Pengembangan berulang dan Incremental adalah kombinasi dari kedua desain iteratif atau metode iteratif dan incremental membangun model untuk pembangunan. “Selama pengembangan perangkat lunak, lebih dari satu iterasi dari siklus pengembangan perangkat lunak mungkin berlangsung pada saat yang sama.” dan

“Proses ini dapat digambarkan sebagai” akuisisi evolusi “atau” inkremental membangun “pendekatan.”

Keuntungan dari Iterative model, User dapat mencoba sistem yg sudah dikembangkan dan kemudian dapat memberikan masukkan > keterlibatan user semakin intens dampak positif dalam pengembangan. Prototype relatif lebih mudah dibangun dan tidak memerlukan waktu yang lama. Dengan prototype, kesalahan & kelalaian dalam pengembangan dapat segera diketahui

Kelemahan dari Iterative model, Setiap iterasi bergantung prototype sebelumnya solusi final umumnya terjadi apabila ada perbedaan yg nyata pada prototype sebelumnya. Formal end-of-phasemungkin tidak terjadi, karena sangat sulit menentukan scope dari suatu prototype > proyek tidak pernah selesai. Dokumentasi seringkali tdk lengkap > fokus pada pembuatan prototype. Isu2 mengenai system backup & recovery, system performance dan system security, kurang/tidak diperhatikan dan sering terlupakan

e. Model Spiral

Model Spiral, merupakan model pengembangan system yang digambarkan berupa spiral. Model spiral ini tidak merepresentasikan rangkaian tahapan dengan penelusuran balik (back-tracking), tidak ada fase-fase tahapan yang tetap seperti spesifikasi atau perancangan. Setiap untaian pada pada spiral menunjukkan fase software process. Model spiral adalah model proses software yang evolusioner yang merangkai sifat iteratif dari prototipe dengan cara kontrol dan aspek sistematis dari model sekuensial linier. Model pengembangan perangkat lunak dengan metode spiral memiliki dua model yaitu prototyping dan waterfall. Model ini dikenal dengan sebutan Spiral Boehm.

Gambar 4.6 Ilustrasi model spiral

Fase-fase dari proses pengembangan perangkat lunak pada model spiral diwakili oleh tiap putaran. Dapat disimpulkan, putaran paling dalam berkaitan dengan kelayakan sistem, putaran berikutnya yaitu definisi kebutuhan, dilanjutkan dengan perancangan sistem, dan seterusnya. Putaran-putaran/loop pada model spiral dibagi menjadi empat sektor, yaitu:

1) *Objective Setting* (Penetapan Tujuan)

   Hal-hal yang dilakukan pada tahap ini adalah mengidentifikasi kendala pada proses dan produk, tujuan spesifik untuk projek, membuat rencana pengelolaan yang lebih rinci, merencanakan resiko dalam projek, strategi alternative.

2) *Risk Assesment and Reduction* (Penilaian dan pengurangan resiko)

   Tahapan pada risk manajemen reduction dilakuakan melihat hasil identifikasi pada detail analisis pada resiko projek di tahap sebelumnya. Sebagai contoh apabila ada

resiko bahwa requirement tidak pantas atau kurang, maka akan dibuat prototype sistem.

3) *Development and Validation* (Pengembangan dan Validasi)

   Setelah tahap *risk assesment*, dapat dipilih model pengembangan sistem. Sebagai contoh, pembuatan prototype lembaran (throwaway) akan menjadi pendekatan pengembangan yang paling baik jika resiko user interface lebih besar atau dominan. Jika resiko utama yang diidentifikasi adalah integrasi subsistem, maka waterfall model mungkin adalah model pembangunan terbaik untuk digunakan.

4) *Planning* (Perencanaan)

   Pada tahap planning, dilakukan review pada proyek dan pengambilan keputusan terkait kelanjutan tahapan pada putaran spiral selanjutnya. Apabilan telah diputuskan untuk dilanjutkan, rencana akan disusun untuk fase selanjutnya dari proyek.

Perbedaan utama model perangkat lunak lainnya dengan model spiral adalah penjelasan eksplisit mengenai resiko di tiap tahapan proses perangkat lunak yang dilalui (Ian Sommerville). Pada model spiral, Perangkat Lunak dikembangkan dalam seri dari versi evolusioner. Selama iterasi awal, versi sebelumnya akan menjadi protoype. Pada iterasi selanjutnya, terdapat kenaikan versi lebih komplit dari sistem yang diproduksi. Model spiral dibagi menjadi sebuah aktivitas *framework* yang didefinisikan oleh tim pengembang. Tiap aktivitas framework merepresentasikan 1 segmen dari jalur spiral. Sebagai permulaan proses evolusioner, tim pengembang melakukan aktivitas yang disiratkan oleh sirkuit spiral searah jarum jam, dimulai dari tengah. Resiko dipertimbangkan dari tiap evaluasi yang dibuat.

Spesifikasi produk dihasilkan di sirkuit pertama pada model spiral. Selanjutnya yaitu digunakan untuk mengembangkan sebuah prototype dan secara progresif menjadi versi Perangkat lunak yang lebih canggih. Penyesuaian terhadap rencana proyek dihasilkan pada tiap lintasan yang melewati. Biaya dan jadwal pengembangan Perangkat Lunak disesuaikan berdasarkan feedback dari customer setelah diantarkan. Model spiral dapat diadaptasi untuk sepanjang hidup Perangkat Lunak, berbeda dengan model proses Perangkat Lunak yang lain, yang berhenti ketika Perangkat Lunak sampai kepada customer.

Sirkuit pertama dalam spiral model mempresentasikan konsep pengembangan proyek yang dimulai dari tengah spiral dan berlanjut untuk banyak iterasi sampai konsep pengembangan selesai. Spiral model akan tetap digunakan sampai Perangkat Lunak tidak digunakaan. Jika dalam beberapa waktu proses terbengkalai, tetapi kapanpun perubahan diinisialisasi, proses dimulai pada entri poin yang sesuai.

Model spiral merupakan pendekatan yang realistik untuk mengembangkan sistem skala besar, karena dikembangkan sebagai proses yang berprogres, developer dan customer lebih baik mengerti dan memberi feedback pada tiap level evolusioner.

- **Kesimpulan**

SDLC merupakan pola yang diambil untuk mengembangkan sistem perangkat lunak, yang terdiri dari tahap-tahap: rencana(planning), analisis (analysis), desain (design), implementasi (implementation), uji coba (testing) dan pengelolaan (maintenance). Model SDLC mempunyai kelebihan dan kekurangan masing-masing pada tiap tahapannya. Hal yang paling penting yaitu mengenali type pelanggan/ customer dan memilih menggunakan model SDLC yang sesuai dengan karakter pelanggan dan sesuai dengan karakter

pengembang. Beberapa model SDLC adalah model waterfall, model prototype, model RAD, model iterative, dan model spiral.

Evaluasi

Setelah menyimak materi SDLC, silahkan jawab pertanyaan berikut:

1. Apa risiko yang dihadapi jika pengembangan aplikasi tidak mengikuti tahapan-tahapan SDLC?
2. Jelaskan pendapat kalian alas an munculnya SDLC!
3. Mengapa model Waterfall dianggap sebagai model yang paling sederhana dan hanya cocok digunakan untuk aplikasi skala kecil?
4. Mengapa metode iterative cocok digunakan untuk pengembang dengan turnover staf yang tinggi?

Daftar Pustaka

Kendall, Kenneth E. and Kendall, Julie E., 2011, *System Analysis and Design 8th ed*, Prentice Hall, New Jersey.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Pressman, R.S. 2012. Rekayasa Perangkat Lunak: Pendekatan Praktisi. Yogyakarta: Penerbit Andi.

Pressman, Roger S., 2001, *Software Engineering A Practitioner's Approach 7th ed*, McGraw-Hill, New York.

Simarmata, Janner. 2010. *Rekayasa Perangkat Lunak*. Yogyakarta: Andi

Sommerville, Ian., 2011, *Software Engineering 9th ed*, Pearson Education, Boston.

Sukamto, Rosa A., & Shalahuddin, M. 2016. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object*. Bandung: Informatika

Bab V
Basis Data

Setelah mempelajari bab **Basis Data**, maka diharapkan :

4. Mahasiswa mampu menjelaskan definisi basis data
5. Mahasiswa mampu menerapkan ERD
6. Mahasiswa mampu menerapkan studi kasus ERD

1. Definisi Basis Data

Basis data terdiri dari dua kata yaitu basis dan data, Basis dapat dikatakan gudang, markas, atau tempat berkumpul. Sedangkan sata dapat diartikan representasi dari fakta dunia yang mewakili sebuah obyek (manusia, peristiwa, barang, keadaan dsb) yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi atau kombinasinya. Elmasri menyampaikan bahwa istilah basis data lebih dibatasi pada arti implisit yang khusus mempunyai beberapa pengertian, yaitu :

a. Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implicit. Sehingga apabila data terkumpul secara acak dan tanpa mempunyai arti, tidak dapat disebut basis data.
b. Basis data dapat digunakan oleh beberapa pemakai dan beberapa aplikasi yang sesuai dengan kepentingan pemakai.
c. Basis data merupakan penyajian suatu aspek dari dunia nyata (*real word* atau *miniworld*). Misalnya basis data perbankan, perpustakaan, pertanahan, perpajakan, dan lain-lain.
d. Basis data perlu dirancang, dibangun dan data dikumpulkan untuk suatu tujuan tertentu.

Dari beberapa pengertian tersebut, dapat diambil kesimpulan bahaw sistem basis data merupakan sistem terkomputerisasi yang berujuan untuk memelihara data yang telah diolah dan

mempercepat proses saat dibutuhkan. Dengan kata lain basis data merupakan media untuk menyimpan data agar dapat diakses dengan mudah dan cepat. Kebutuhan basis data dalam sistem informasi meliputi memasukkan, menyimpan, mengambil kembali data dan membuat laporan berdasarkan data yang telah disimpan.

2. Entity Relatioship Diagram

Pemodelan awal basis data yang paling banyak digunakan adalah *Entity Relationship Diagram* (ERD). ERD digunakan untuk pemodelan basis data relational. Diagram relasi entitas atau ERD merupakan suatu diagram dalam bentuk gambar atau simbol yang mengidentifikasi tipe dari entitas di dalam suatu sistem yang diuraikan dalam data dengan atributnya, dan menjelaskan hubungan atau relasi diantara entitas tersebut. Atau dapat dikatakan bahwa ERD adalah model jaringan yang menggunakan susunan data yang disimpan dalam sistem secara abstrak. ERD menekankan pada struktur dan relationship data.

Untuk dapat membuat entity relasional diagram, maka komponen yang harus terpenuhi adalah:

a. Obyek Data, Atribut dan Hubungan.

Obyek Data Adalah representasi dari hampir semua informasi gabungan yang harus dipahami oleh perangkat lunak. Objek data dapat berupa entitas eksternal (misalkan semua yang menghasilkan informasi), suatu benda (misal laporan atau tampilan), peristiwa (misalnya proses meminjam) atau event, peran (misalnya peminjam), unit organisasi atau suatu struktur. Sebagai contoh : orang atau mobil dapat dipandang sebagai objek data bila salah satu dari mereka dapat didefinisikan dalam bentuk atribut. Deskripsi objek data menghubungkan objek data dengan semua atributnya. Obyek data dihubungkan satu dengan yang lainnya,

misalkan seorang dapat memiliki mobil, dimana hubungan “memiliki” mengkonotasikan suatu hubungan khusus antara seorang dengan mobil.

Atribut Menentukan property suatu obyek data dan mengambil salah satu dari tiga karakteristik yang berbeda. Atribut dapat digunakan untuk:

1) Menamai sebuah contoh dari obyek data
2) Menggambarkan contoh
3) Membuat referensi ke contoh yang lain pada tabel yang lain.

Hubungan Obyek data disambungkan satu dengan lainnya dengan berbagai macam cara. Andaikan ada dua objek data, buku dan toko buku, obyek tersebut dapat diwakilkan dengan menggunakan dua notasi sederhana, dibangun suatu hubungan anatar buku dengan toko buku karena kedua obyek data tersebut berhubungan. Hubungan tersebut dapat berupa :

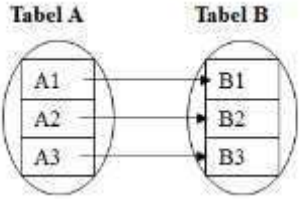
1) Toko buku memesan buku
2) Toko buku menampilkan buku
3) Toko buku menjual buku

Hubungan memesan, menampilkan, menjual mendefinisikan hubungan yang relevan antara buku dan toko buku. Penting untuk dicatat bahwa hubungan obyek mempunyai dua arah, dimana mereka dpaat dibaca dari dua arah, misalnya :toko buku memesan buku atau buku dipesan oleh toko buku.

b. Kardinalitas dan Modalitas Kardinalitas

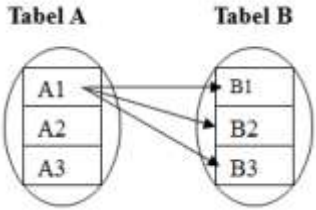
Model data harus dapat merepresentasikan jumlah peristiwa dari obyek di dalam hubungan yang diberikan.

1) Satu ke satu (1:1) Misalnya: seorang suami hanya dapat memiliki satu istri, dan seorang istri hanya mempunyai satu suami. Contoh ilustrasi dapat ditunjukkan pada Gambar 5.1



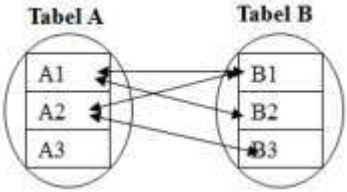
Gambar 5.1 Relasi Satu ke Satu

2) Satu ke banyak (1:N) Misalnya: seorang ibu kandung dapat memiliki banyak anak, tetapi seorang anak hanya dapat memiliki satu ibu kandung. Contoh ilustrasi dapat ditunjukkan pada Gambar 5.2



Gambar 5.2 Relasi Satu ke Banyak

3) Banyak ke banyak (M:N) Misalnya: seorang paman dapat memiliki banyak keponakan, sementara itu seorang keponakan dapat memiliki banyak paman. Contoh ilustrasi dapat ditunjukkan pada Gambar 5.3



Gambar 5.3 Relasi Banyak ke Banyak

Modalitas dari suatu hubungan adalah nol bila tidak ada kebutuhan eksplisit untuk hubungan yang terjadi atau hubungan itu bersifat opsional. Modalitas bernilai satu jika suatu kejadian dari hubungan merupakan perintah.

Untuk menggambarkan ERD setidaknya ada empat langkah yang harus dilakukan oleh perancang basis data yaitu:

a. Menemukan atau mendefinisikan Entitas
b. Menemukan atau mendefinisikan atribute
c. Menemukan atau mendefinisikan Relasi
d. Menggambarkan ERD menggunakan notasi-notasi standar

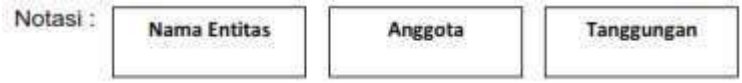
a. Entitas (*Entity*)

Entitas merupakan obyek yang mewakili sesuatu dalam dunia nyata dan dapat dibedakan antara satu dengan lainnya (unique). Setiap entitas memiliki beberapa atribut yang mendeskripsikan karakteristik dari objek tersebut. Atribut Dapat berupa:

1) Fisik (manusia, pegawai, dsb)
2) Abstrak/konsep (pekerjaan, mata kuliah, department, dsb)
3) Kejadian (penjualan, pembelian, peminjaman, dsb)

Entitas dapat dibedakan menjadi dua jenis yaitu **entitas kuat** dan **entitas lemah**. Entitas kuat yaitu keberadaannya tidak tergantung pada entitas yang lain, sementara e**ntitas lemah** adalah

yang keberadaannya tergantung pada entitas lain. Notasi umum entitas kuat dengan nama entitas Anggota dan entitas lemah dengan nama entitas tanggungan ditunjukkan pada Gambar 5.4. Entitas tanggungan disebut sebagai **entitas lemah** karena jika data seorang pegawai dihapus maka data tanggungannya juga akan terhapus. Keberadaan data tanggungan tergantung pada data di pegawai. Lebih lanjut dapat dilihat pada Gambar 5.5.



Gambar 5.4 Contoh Notasi

Contoh :

Entitas	Atribut
Petugas	NIP, Nama, Alamat, Agama, Jenis kelamin
Departemen	No, Nama, Lokasi

Gambar 5.5 Contoh Penggunaan Simbol Entitas dan Atribut

Urutan langkah untuk mendefinisikan entitas dalam sistem adalah:

a. Membuat gambaran cerita (ilustrasi) terkait sistem
b. Menandai setiap objek yang diwakili oleh kata benda yang ada dalam ilustrasi
c. Pada setiap objek, pastikan memiliki karakteristik yang akan dijadikan sebagai atribut
d. Menentukan objek yang merupakan entitas (Jika memang ia memiliki karakteristik jadikan ia sebagai entitas)

e. Menggambarkan entitas beserta atributnya menggunakan notasi simbol yang telah ditentukan.

3. Studi Kasus ERD

Contoh cara menentukan entitas:

Langkah 1. *Deskripsi tentang gambaran sistem (misalnya gambaran tentang system informasi perpustakaan):*

Departemen A mempunyai perpustakaan, perpustakaan ini dijaga oleh dua orang pegawai. Anggota perpustakaan ini adalah seluruh pegawai departemen yang sudah terdaftar menjadi anggota perpustakaan. Koleksi buku yang dimiliki ada sekitar 200 eksemplar, dengan berbagai jenis buku, pengarang dan penerbit. Untuk melakukan peminjaman buku, anggota melakukan proses peminjaman ke petugas dengan memberikan kartu anggota perpustakaan, lama peminjaman adalah dua minggu, proses pengembalian dilakukan dengan memberikan buku dan kartu anggota untuk pengecekan. Jika anggota terlambat mengembalikan maka akan dikenakan denda Rp.1000, 00 per hari.

Langkah 2. *Menandai setiap objek yang diwakili oleh kata benda yang ada dalam ilustrasi*

Departemen A mempunyai perpustakaan, perpustakaan ini dijaga oleh dua orang **pegawai. Anggota** perpustakaan ini adalah seluruh pegawai departemen yang sudah terdaftar menjadi anggota perpustakaan. Koleksi buku yang dimiliki ada sekitar 200 eksemplar, dengan berbagai **jenis buku, pengarang** dan **penerbit**. Untuk melakukan peminjaman buku, anggota melakukan proses **peminjaman** ke petugas dengan memberikan kartu anggota perpustakaan, lama peminjaman adalah dua minggu, proses

pengembalian dilakukan dengan memberikan buku dan kartu anggota untuk pengecekan. Jika anggota terlambat mengembalikan maka akan dikenakan **denda** Rp.1000, 00 per hari.

Langkah 3. *Untuk setiap objek tersebut yakinkan bahwa ia memiliki karakteristik yang nanti disebut sebagai atribut*

Pegawai : id, nama, Username, Password
Anggota : No_anggota, Nama, alamat, Nomor_telp, jenis_kelamin
Buku : Kode, kategori,Judul,Jumlah_halaman, ISBN, Pengarang, penerbit
Jenis buku : Kode, Jenis
Pengarang : Kode, Nama
Penerbit : Kode, Nama
Peminjaman : Kode_peminjaman, Id_Petugas, No_anggota, Kode_Buku, Tgl_pinjam, Tgl_kembali
Pengembalian : Kode_pengembalian, No_anggota, tgl_kembali, denda

b. Atribut

Atribut merupakan karakteristik atau sifat-sifat pada entitas. Nama atribut ini identik dengan nama kolom atau field pada suatu table dalam basis data. Candidat Key adalah merupakan superkey yang jumlah atributnya paling sedikit.

Misalnya *candidat key* untuk entitas petugas antara lain:

1) Id_Pegawai
2) Nama (harus dipastikan bahwa ktidak ada nama yang sama antara satu baris dengan baris yang lain)

Primary key merupakan suatu *candidat key* yang dipilih menjadi kunci utama karena sering dijadikan acuan untuk mencari

informasi, ringkas, menjadi keunikan suatu baris. Misalnya kodeBuku antara buku yang satu dengan buku yang lain pasti berbeda, dalam hal ini kodeBuku dapat digunakan sebagai suatu key. Gambar 5.6 menjelaskan simbol atau notasi primary key.

Gambar 5.6 Primary key

c. Relasi

Relasi menyatakan hubungan antara dua atau beberapa entitas.Setiap relasi mempunyai batasan (constraint) terhadap kemungkinan kombinasi entitas yang berpartisipasi. Batasan tersebut ditentukan dari situasi yang diwakili relasi tersebut. Ragam atau jenis relasi dibedakan menjadi beberapa macam antara lain adalah.

a) Relasi Binary

Relasi ini merupakan relasi antara 2 entitas. Relasi ini dibedakan menjadi :

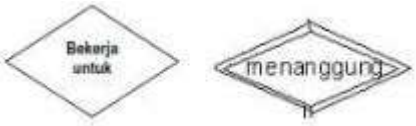
1) Relasi One-to-one (notasi 1:1)
2) Relasi One-to-many (notasi 1:N) atau many-to-one (notasi N:1)
3) Relasi Many-to-many (notasi M:N)

b) Relasi Ternary

Relasi ini merupakan relasi antara 3 entitas atau lebih. Dalam Relasi One-to-one (1:1) setiap atribute dari satu entitas berpasangan dengan satu attribute dari entitas yang direlasikan.Dalam relasi One-to-many (1:N) atau many-to- one (N:1) satu atribute berelasi dengan beberapa atribute dari entitas yang direlasikan.Dalam Many-to-many (M:N) satu atribute berelasi dengan beberapa atribute dari entitas yang direlasikan, begitu pula sebaliknya.

Notasi Relasi

Relasi dapat dikategorikan menjadi relasi kuat dan relasi lemah. gambar 5.7 menunjukkan notasi umum untuk relasi kuat dan relasi lemah.



Gambar 5.7 Relasi kuat dan relasi lemah

Langkah-langkah yang digunakan untuk mengidentifikasi relasi yaitu :

1) Tandai setiap hubungan dari gambaran sistem yang diwakili oleh kata kerja yang ada di dalam ilustrasi gambaran sistem beserta entitas yang berhubungan
2) Lakukan identifikasi rasio kardinalitas dari setiap hubungan
3) Lakukan identifikasi batasan partisipasi dari setiap hubungan/relasi yang ada beserta kemungkinan atribut yang muncul dari setiap hubungan

Gambarkan hubungan tersebut dalam bentuk notasi diagram dan gabungkan dengan notasi Entitas dan atribut yang dibuat sebelumnya. Misalnya tentukan relasi untuk sistem informasi perpustakaan dengan melihat deskripsi sistem di atas.

Langkah Penyelesaian ditunjukkan pada langkah-langkah berikut:

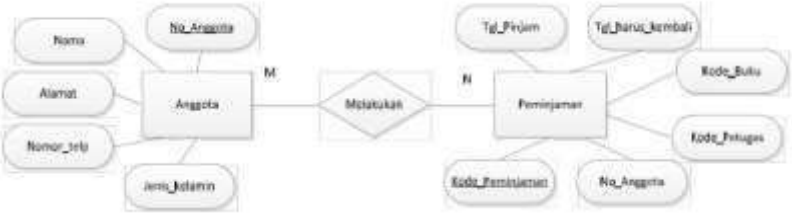
Langkah 1:

Tandai dan tentukan setiap hubungan yang ada pada gambaran sistem yang diwakili oleh kata kerja yang ada di dalam ilustrasi dan entitas yang berhubungan. Identifikasi hubungan antara entitas

Tabel Entitas dan Relasi

Entitas 1	Hubungan	Entitras 2
Anggota	Melakukan	Peminjaman
Pegawai	Mengelola	Peminjaman
Pengarang	Menulis	Buku
Penerbit	Menerbitkan	Buku
Anggota	Melakukan	Pengembalian
Pegawai	Mengelola	Pengembalian

Dari tabel di atas maka berikut relasinya :



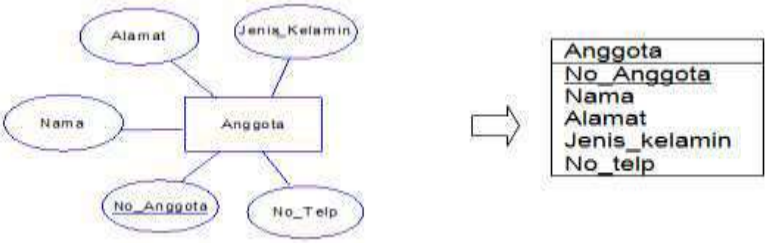
Gambar 5.8 Notasi hubungan entitas dan relasi

Langkah 2.

Mapping ER diagram ke tabel. Didalam data base yang menjadi pusat perhatian dan intisari dari sistem database itu adalah table dan relasinya. Tabel ini sama artinya dengan entitas , setiap orang bisa membuat table tetapi membuat table yang baik tidak semua orang dapat melakukannya. Kebutuhan akan membuat tabel yang baik ini maka muncul teori beberapa teori atau metode yaitu mapping ERD to table.

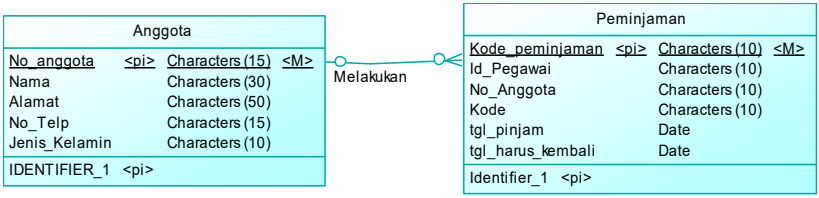
Contoh:

Dari relasi yang telah dibuat pada sistem informasi perpustakaan, pada setiap entitasnya pilih satu atribut kunci sebagai primary key (atribut harus uniqe).



Gambar 5.9. Maping Notasi ke Diagram ER

Dengan cara yang sama dapat dilakukan mapping ERD to table pada semua entitas. Setelah semua entitas selesai dibuat, tentukan relasi antar entitas sesuai dengan tabel hubungan antar entitas di atas. Misalnya relasi antara entitas anggota dan Peminjamanya adalah melakukan seperti yang ditunjukkan pada Gambar 5.10



Gambar 5.10 Entity Relationship Diagram

Kesimpulan

Basis data merupakan kumpulan data dari berbagai sumber yang secara logika mempunyai arti implicit. Sehingga apabila data terkumpul secara acak dan tanpa mempunyai arti, tidak dapat disebut basis data. Pemodelan awal basis data yang paling banyak digunakan adalah Entity Relationship Diagram (ERD). ERD digunakan untuk pemodelan basis data relational. Diagram relasi entitas atau ERD adalah suatu diagram dalam bentuk gambar atau simbol yang mengidentifikasi tipe dari entitas di dalam suatu sistem yang

diuraikan dalam data dengan atributnya, dan menjelaskan hubungan atau relasi diantara entitas tersebut.

Evaluasi

Setelah menyimak materi basis data, silahkan jawab pertanyaan berikut:

1. Apa yang dimaksud dengan basis data?
2. Apakah fungsi basis data pada suatu sistem informasi?
3. Jelaskan yang kalian ketahui tentang ERD!
4. Jelaskan dengan singkat, langkah menyusun ERD!
5. Buatlah ERD untuk sistem informasi atau aplikasi yang berhubungan dengan dunia pendidikan!

Daftar Pustaka

Elmasri dan Navathe. 2007. *Fundamentals of Database Systems, Fifth Edition. Boston: Pearson Education, Inc. Addison Wesley*

Fatansyah. 2012. Basis Data. Bandung:Informatika

Indrajani. 2015. *Database Design.* Jakarta: Elex Media Komputindo

Kadir, Abdul. 2008. *Belajar Database menggunakan MySQL*. Yogyakarta : Andi Offset

Sukamto, R. A., & Shalahuddin, M. 2016. Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object. Bandung: Informatika

Bab VI
Data Flow Diagram (DFD)

Setelah mempelajari bab ***Data Flow Diagram***, maka diharapkan :

5. Mahasiswa mampu menjelaskan definisi DFD
6. Mahasiswa mampu menjelaskan komponen DFD
7. Mahasiswa mampu menerapkan tingkatan level DFD
8. Mahasiswa mampu menerapkan tahapan pembuatan DFD
9. Mahasiswa mampu menerapkan studi kasus DFD

1. Definisi Data Flow Diagram

Data Flow Diagram (DFD) atau dalam bahasa Indonesia menjadi Diagram Alir Data adalah representasi grafik yang menggambarkan aliran informasi dan transformasi informasi yang diaplikasikan sebagai data yang mengalir dari input dan output. Penjelasan lain terkait DFD yaitu suatu cara atau metode untuk membuat rancangan sebuah sistem yang mana berorientasi pada alur data yang bergerak pada sebuah sistem nantinya. Dalam pembuatan Sistem Informasi, DFD sering digunakan. DFD dibuat oleh para analis untuk membuat sebuah sistem yang baik. Dimana DFD ini nantinya diberikan kepada para programmer dimana para programmer melakukan sebuah coding sesuai dengan DFD yang dibuat oleh para analis sebelumnya.

DFD merupakan sebuah teknik analisis yang digunakan untuk menggambarkan aliran input dalam sebuah sistem yang dioleh oleh proses dan menghasilkan suatu keluaran (*output*). Diagram ini menggambarkan apa yang terjadi dalam sebuah sistem. DFD disajikan dalam bentuk gambar yang berisi simbol atau notasi yang digunakan untuk memahami mekanisme aliran data dalam suatu sistem. DFD merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi. DFD dapat pula digunakan untuk menggambarkan suatu analisa maupun rancangan

sistem yg mudah dikomunikasikan oleh profesional sistem (sistem analist) kepada pemakai (*user*) maupun pembuat program (*programmer*).

Fungsi dan manfaat DFD antara lain adalah

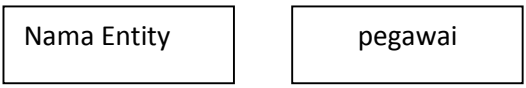
a. DFD membantu para analis sitem meringkas informas tentang sistem, mengetahui hubungan antar sub-sub sistem, membantu perkembangan aplikasi secara efektif.
b. DFD berfungsi sebagai alat komunikasi yang baik antara pemakai dan analis sistem.
c. DFD dapat menggambarkan sejumlah batasan otomasi untuk pengembangan alternative sistem fisik.

2. Komponen DFD

Terdapat beberapa notasi atau simbol yang digunakan dalam DFD. Notasi tersebut merupakan karakteristik dari suatu system. Notasi dan simbol-simbol yang sering dipakai antara lain adalah : yaitu :

a. *Terminator* atau *External Entity*.

   Terminator disimbolkan dalam bentuk persegi panjang, yang mewakili entity luar dimana sistem berkomunikasi. Biasanya notasi ini melambangkan orang atau kelompok orang misalnya organisasi diluar sistem, grup, departemen, perusahaan pemerintah, dan berada di luar kontrol sistem yang dimodelkan. Pada sejumlah kasus dapat merupakan sistem lain, sebagai contoh adalah entity: pegawai, mahasiswa, peserta didik dll. Gambar dibawah ini menjelaskan notasi atau simbol Terminator (External Entity)



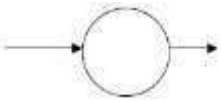
b. Proses

Komponen proses menggambarkan suatu transformasi input menjadi output. Penamaan proses disesuaikan dgn proses atau kegiatan yang sedang dilakukan Pemberian nama suatu proses menggunakan kata kerja transistif yaitu kata kerja yang membutuhkan objek. Komponen Proses disimbolkan dalam bentuk lingkaran dengan nama proses ditulis didalam lingkaran. Terdapat beberapa hal yang harus diperhatikan tentang komponen proses dalam diagram alur data antara lain adalah:

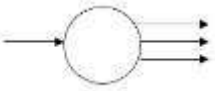
1) Setiap komponen proses harus memiliki input dan output.
2) proses dapat dihubungkan dgn komponen terminator, data store atau alur data.
3) Sistem, bagian, divisi atau departemen yang sedang dianalisis oleh profesional sistem dapat digambarkan dengan komponen proses.

Terdapat 4 kemungkinan yang dapat terjadi dalam sutau proses sehubungan dengan pengolahan komponen input menjadi output yaitu :

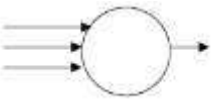
1) Satu input dan satu output
2) Satu input dan banyak output
3) Banyak input dan satu output
4) Banyak input dan banyak output

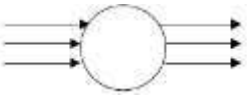


1) Satu input dan satu output



2) Satu input dan banyak output



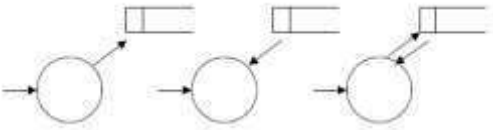


3) Banyak input dan satu output
4) banyak input dan banyak output

c. Penyimpanan Data (*Data Store*)

Data store digunakan untuk memodelkan kumpulan data atau paket data. Penyimpanan data kadangkala didefinisikan sebagai suatu mekanisme diantara dua proses yang dibatasi oleh jangka waktu tertentu. Data store dapat berupa fie atau basis data yang tersimpan dalam unit penyimpanan seperti disket, harddisk. Alur data digunakan untuk menerangkan perpindahan data / paket data dari satu bagian ke bagian lainnya. Alur data dapat berupa kata, pesan, formulir atau informasi. Data store disimbolkan dengan garis sejajar. Terdapat beberapa hal Yang harus diperhatikan dalam pendefinisian data store anatra lain ialah :

1) Alur data dari proses menuju data store. Hal ini berarti bahwa data store berfungsi sebagai tujuan atau tempat penyimpanan dari suatu proses (proses write).
2) Alur data dari data store ke proses. Hal ini berarti data store berfungsi sebagai sumber atau suatu proses memerlukan data (proses read).
3) Alur data dari proses menuju data store dan sebaliknya. Hal ini berarti alur data berfungsi Sebagai sumber dan tujuan (proses update)



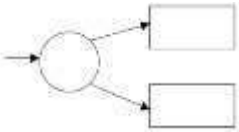
d. Aliran data (*flow*)

Alur data digunakan untuk menerangkan perpindahan data atau paket data dari satu bagian ke bagian lainnya. Alur data

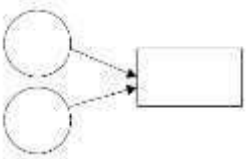
menunjukkan aliran data yang dapat berupa masukkan untuk sistem atau hasil proses system. Terdapat beberapa konsep tentang alur data antara lain adalah :

1) *Packets of data*. Apabila ada 2 data atau lebih yg mengalir dari 1 sumber yang sama menuju pada tujuan yang g sama pula dan mempunyai hubungan maka digambarkan dengan 1 alur data.

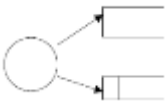
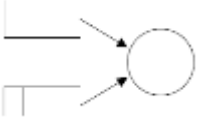
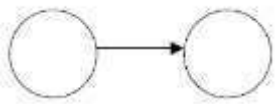
2) *Diverging data flow*. Apabila ada sejumlah paket data yg berasal dari sumber yg sama menuju pada tujuan yang berbeda atau paket data yg kompleks maka diagram dibagi menjadi beberapa elemen data yg dikirim ke tujuan yg berbeda.



3) *Converging data flow*. Alur data yang mengumpul menunjukkan beberapa alur data yang berbeda dari sumber data yang berbeda bergabung bersama-sama menuju tujuan yang sama.



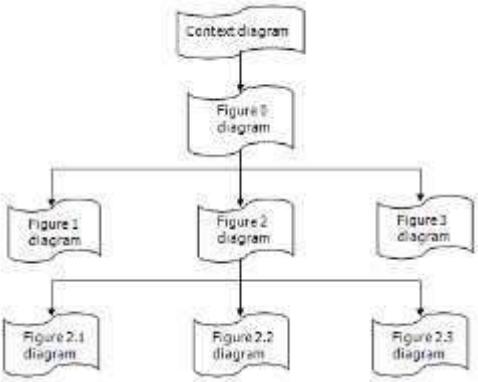
4) Sumber dan Tujuan. Arus data harus dihubungkan pada proses, baik dari maupun menuju proses.

		
Dari proses ke bukan proses	Dari bukan proses menuju proses	Dari proses ke proses

3. Tingkatan atau Level DFD

DFD dapat dibagi menjadi beberapa level yang lebih detail untuk merepresentasikan aliran informasi atau fungsi yang lebih detail. Dalam penerapannya tidak ada aturan yang baku tentang tingkatan atau level DFD. Secara umum terdapat beberapa tingkatan yang sering digunakan antara laian adalah sebagai berikut

a. DFD level 0 atau biasa disebut diagram konteks. Diagram konteks menggambarkan secara global aliran informasi dan data yang akan dilakukan oleh system. Diagram konteks ini merupakan level tertinggi (*top level*) yang menggambarkan hubungan antar system dengan entitas diluar system dan merupakan gambaran system secara keseluruhan. Komponen yang ada dalam diagram ini biasanya komponen proses, external entity dan alur data.
b. DFD level 1, diagram ini menjelaskan lebih detail dari diagram konteks. Diagram ini merupakan dekomposisi diagram konteks. Beberapa proses dalam diagram konteks dapat dijelaskan lebih rinci.
c. DFD level 2, diagram ini merupakan dekomposisi dari diagram level 1.
d. Diagram level 3 dan seterusnya, diagram ini merupakan dekomposisi dari diagram level 2.

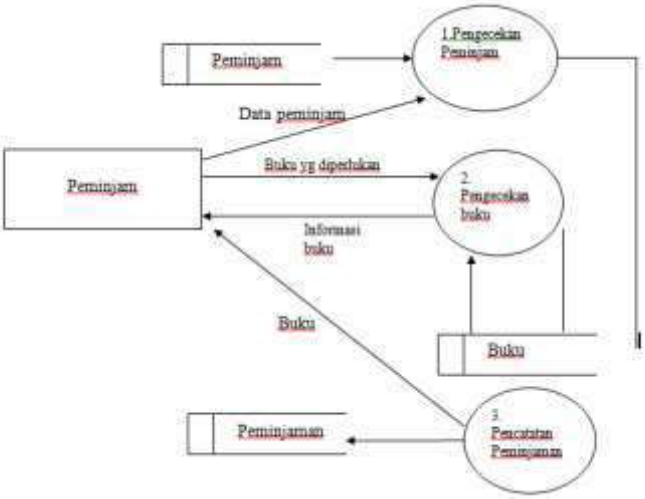


Adapun langkah-langkah yang dilakukan dalam pembuatan diagram alur data adalah sebagai berikut:

a. Membuat diagram konteks
   1) Tentukan nama sistemnya.
   2) Tentukan batasan sistemnya.
   3) Tentukan terminator apa saja yg ada dalam sistem.
   4) Tentukan apa yg diterima dan diberikan terminator dari/pada sistem.
   5) Gambarkan diagram *context*

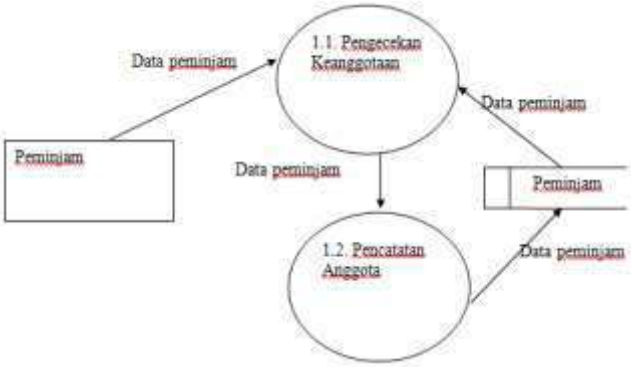
b. Membuat diagram level 1
   1) Tentukan proses utama yg ada pada sistem.

2) Tentukan apa yg diberikan/diterima masing-masing proses pada/dari sistem sambil memperhatikan konsep keseimbangan (alur data yg keluar/masuk dari suatu level harus sama dgn alur data yg masuk/keluar pada level berikutnya)
3) Apabila diperlukan, munculkan data store (master) sebagai sumber maupun tujuan alur data.
4) Gambarkan diagram level zero.
5) Hindari perpotongan arus data
6) Beri nomor pada proses utama (nomor tidak menunjukkan urutan proses).



c. Membuat diagram level 2
1) Tentukan proses yg lebih kecil (sub-proses) dari proses utama yg ada di level zero.
2) Tentukan apa yg diberikan/diterima masing-masing sub-proses pada/dari sistem dan perhatikan konsep keseimbangan.

3) Apabila diperlukan, munculkan data store (transaksi) sbg sumber maupun tujuan alur data.
4) Gambarkan DFD level Satu
5) Hindari perpotongan arus data.
6) Beri nomor pada masing-masing sub-proses yg menunjukkan dekomposisi dari proses sebelumnya. Misalnya 1.1, 1.2, 2.1



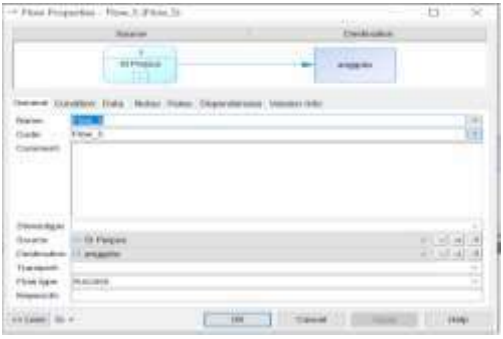
4. Studi Kasus Pembuatan DFD

Terdapat 3 pihak yang berkepentingan dalam sistem informasi ini yaitu anggota, pimpinan dan petugas. DFD Level 0 menyatakan tugas dan hak serang user dari sistem informasi ini, Setiap proses yang perlu dijelaskan diperlukan level yang lebih tinggi yang merupakan subproses dari proses yang ad**a.**

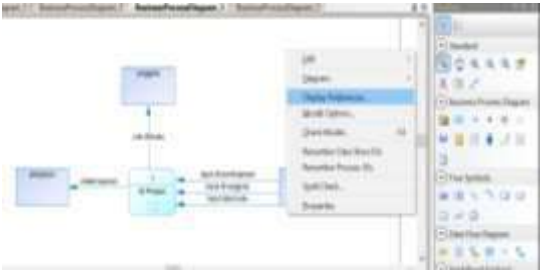
Langkah 1 . Buka Power designer 16.5
Langkah 2. Pilih New Model – Information – Data Flow diagram

Langkah 4. Buat DFD Level Konteks (Diagram 0)

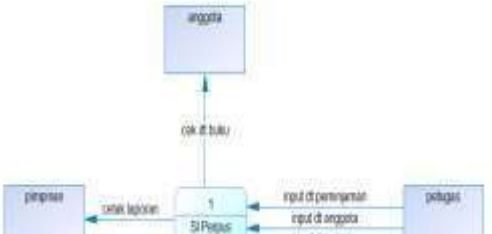
Langkah 5. Hubungkan Antar Eksternal Entity dengan Proses, dan beri nama pada masing-masing flow sesuai dengan hubungan antar entity ke proses



NB: Untuk menampilkan teks name pada data flow
Klik kanan pada area kosong di lembar kerja, pilih display preferences.



- ✓ Pilih flow dan centang pada name dan stereotype, klik set as default
- ✓ **Ulangi pada *resource flow* dan centang name dan *stereotype*, klik *set as default***



Langkah 6. Dekomposisi diagram level konteks untuk membuat Diagram level 1

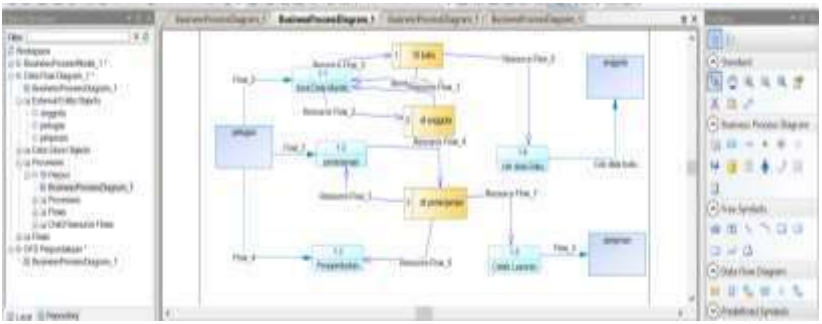
✓ **Klik kanan pada proses, pilih Decompose proses**

Klik pada menu kiri, pilih *Processes*, SI Perpus yang telah dibuat dan klik Business Process Diagram, sampai muncul eksternal entity seperti pada gambar di atas.

Langkah 7. Susun Diagram Level 1 pada DFD

Pada level 1, mulai disusun semua proses secara umum yang ada pada system (sesuai kebutuhan) yang ada pada diagram level konteks yang telah dibuat dan data *store* yang digunakan.

Hubungkan setiap *Entity*, proses, dan data store



- **Kesimpulan**

DFD merupakan sebuah teknik analisis yang digunakan untuk menggambarkan aliran input dalam sebuah sistem yang dioleh oleh proses dan menghasilkan suatu keluaran (output). DFD disajikan dalam bentuk gambar yang berisi simbol atau notasi yang digunakan untuk memahami mekanisme aliran data dalam suatu sistem. DFD merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi. DFD dapat dibagi menjadi beberapa level yang lebih detail untuk merepresentasikan aliran informasi atau fungsi yang lebih detail. Tingkatan level pada DFD yaitu DFD level konteks atau nol, level 1, level 2, dan seterusnya.

Evaluasi

Setelah menyimak materi *data flow diagram*, silahkan jawab pertanyaan berikut:

1. Apa yang kalian ketahui tentang DFD?
2. Jelaskan yang kalian ketahui tentang diagram konteks!
3. Bagaimana tahapan melakukan perancangan sistem menggunakan DFD?
4. Buatlah perancangan pemrograman terstruktur menggunakan DFD dan kamus data untuk studi kasus sistem informasi apotek!

Daftar Pustaka

Dennis, Alan., Wixom, Barbara H. and Roth, Roberta M., 2012, *System Analysis & Design 5th ed*, John Wiley & Sons, New Jersey.

Kendall, J.E. & Kendall, K.E. 2010. Analisis dan Perancangan Sistem. Jakarta: Indeks.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Mc.,Leod, R. Jr. 2002. System Development: A Project Management Approach. New York: Leigh Publishing LLC.

Simarmata, Janner. 2010. *Rekayasa Perangkat Lunak*. Yogyakarta: Andi

Sommerville, Ian., 2011, *Software Engineering 9th ed*, Pearson Education, Boston.

Sukamto, Rosa A., & Shalahuddin, M. 2016. Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object. Bandung: Informatika

Bab VII
Pemodelan UML

Setelah mempelajari bab **Pemodelan UML** , maka diharapkan :

6. Mahasiswa mampu menjelaskan kompleksitas pengembangan perangkat lunak
7. Mahasiswa mampu menjelaskan pemodelan perangkat lunak
8. Mahasiswa mampu menerapkan UML
9. Mahasiswa mampu menerapkan Use Case Diagram
10. Mahasiswa mampu menerapkan Activity Diagram

1. Kompleksitas Pengembangan Perangkat Lunak

Mengelola pengembangan perangkat lunak bukanlah hal yang mudah dikarenakan harus menyatukan ide dari beberapa orang yang mempunyai pemikiran berbeda. Kompleksitas sebuah perangkat lunak dapat dilihat dari hal-hal berikut:

a. Kompleksitas domain atau permasalahan perangkat lunak.
b. Kesulitan mengelola proses pengembangan perangkat lunak
c. Kemungkinan fleksibilitas perubahan perangkat lunak
d. Permasalahan karakteristik bagian-bagian perangkat lunak secara diskrit.

2. Pemodelan Perangkat Lunak

Pemodelan dalam suatu rekayasa perangkat lunak merupakan suatu hal yang dilakukan di tahapan awal. Di dalam suatu rekayasa dalam perangkat lunak sebenarnya masih memungkinkan tanpa melakukan suatu pemodelan. Namun hal itu tidak dapat lagi dilakukan dalam suatu industri perangkat lunak. Pemodelan delam perangkat lunak merupakan suatu yang harus dikerjakan di bagian awal dari rekayasa, dan pemodelan ini akan mempengaruhi perkerjaan-pekerjaan dalam rekayasa perangkat lunak tersebut.

Di dalam suatu industri dikenal berbagai macam proses, demikian juga halnya dengan industri perangkat lunak. Perbedaan proses yang digunakan akan menguraikan aktivitas-aktivitas proses dalam cara-cara yang berlainan. Perusahaan yang berbeda

menggunakan proses yang berbeda untuk menghasilkan produk yang sama. Tipe produk yang berbeda mungkin dihasilkan oleh sebuah perusahaan dengan menggunakan proses yang berbeda. Namun beberapa proses lebih cocok dari lainnya untuk beberapa tipe aplikasi. Jika proses yang salah digunakan akan mengurangi kualitas kegunaan produk yang dikembangkan.

Pada rekayasa perangkat lunak, banyak model yang telah dikembangkan untuk membantu proses pengembangan perangkat lunak. Model-model ini pada umumnya mengacu pada model proses pengembangan sistem yang disebut *System Development Life Cycle* (SDLC) seperti yang telah dibahas dibab sebelumnya. Pemodelan perangkat lunak digunakan untuk mempermudah langkah berikutnya dari pengembangan sebuah sistem informasi sehingga lebih terencana. Salah satu perangkat pemodelan adalah *Unified Modelling Language* (UML).

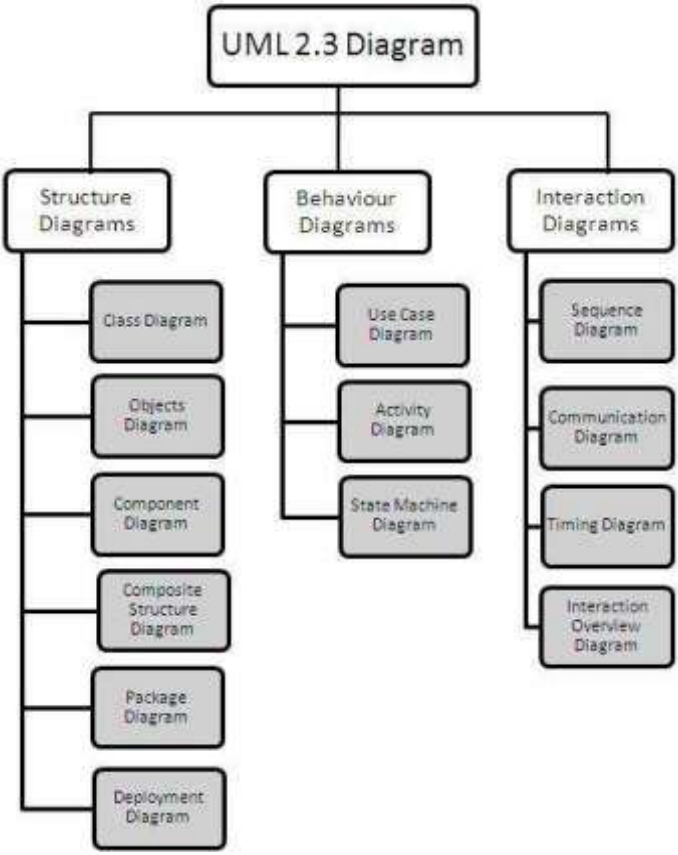
3. *Unified Modelling Language* (UML)

UML merupakan salah satu standart bahasa yang banyak digunakan di dunia industry untuk mendefinisikan *requirement*, membuat analisis & desain, sert amenggambarkan arsitektur dalam pemrograman berorientasi objek. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan mendokumentasikan sistem perangkat lunak. Pendekatan analisa & rancangan dengan menggunakan model *object oriented* mulai diperkenalkan sekitar pertengahan 1970 hingga akhir 1980 dikarenakan pada saat itu aplikasi *software* sudah meningkat dan mulai komplek. Jumlah yang menggunakaan metoda *object oriented* mulai diuji cobakandan diaplikasikan antara 1989 hingga 1994, seperti halnya oleh Grady Booch dari Rational Software Co., dikenal dengan OOSE *(Object-Oriented Software Engineering*), serta James Rumbaugh dari

General Electric, dikenal dengan OMT (Object Modelling Technique). Kelemahan saat itu disadari oleh Booch maupun Rumbaugh adalah tidak adanya standar penggunaan model yang berbasis *object oriented*, ketika mereka bertemu ditemani rekan lainnya Ivar Jacobson dari *Objectory* mulai mendiskusikan untuk mengadopsi masing-masing pendekatan metoda *object oriented* untuk membuat suatu model bahasa yang uniform / seragam yang disebut UML (Unified Modeling Language) dan dapat digunakan oleh seluruh dunia.

Secara resmi bahasa UML dimulai pada bulan oktober 1994, ketika Rumbaugh bergabung *Booch* untuk membuat sebuah project pendekatan metoda yang uniform/seragam dari masing-masing metoda mereka. Saat itu baru dikembangkan draft metoda UML version 0.8 dan diselesaikan serta di release pada bulan oktober 1995. Bersamaan dengan saat itu, Jacobson bergabung dan UML tersebut diperkaya ruang lingkupnya dengan metoda OOSE sehingga muncul release version 0.9 pada bulan Juni 1996. Hingga saat ini sejak Juni 1998 UML version 1.3 telah diperkaya dan direspons oleh OMG (Object Management Group), Anderson Consulting, Ericsson, Platinum Technology, ObjectTime Limited, dll serta di pelihara oleh OMG yang dipimpin oleh Cris Kobryn. UML adalah standar dunia yang dibuat oleh Object Management Group (OMG), sebuah badan yang bertugas mengeluarkan standar-standar teknologi objectoriented dan software component.

UML versi 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram dapat dilihat pada Gambar 7.1



Gambar 7.1 Bagan UML

Berdasarkan Gambar 7.1 berikut penjelasan singkat dari pembagian kategori tersebut:

a. Structure diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan strutur statis dari sistem yang dimodelkan.
b. Behavior diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada suatu sistem.

c. Interaction diagram, merupakan kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

Pada buku ajar ini akan difokuskan pada pemodelan UML dengan *use case* dan *activity diagram*. UML tidak hanya merupakan bahasa pemograman visual saja, namun juga dapat secara langsung dihubungkan ke berbagai bahasa pemograman, seperti JAVA, C++, Visual Basic, atau bahkan dihubungkan secara langsung ke dalam sebuah object-oriented database. Begitu juga mengenai pendokumentasian dapat dilakukan seperti; requirements, arsitektur, design, source code, project plan, tests, dan prototypes. Untuk dapat memahami UML membutuhkan bentuk konsep dari sebuah bahasa model, dan mempelajari 3 (tiga) elemen utama dari UML seperti building block, aturan-aturan yang menyatakan bagaimana building block diletakkan secara bersamaan, dan beberapa mekanisme umum (common).

a. *Building blocks*

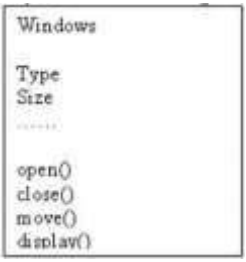
3 (tiga) macam yang terdapat dalam *building block* adalah katagori benda/Things, hubungan, dan diagram. Benda/things adalah abstraksi yang pertama dalam sebuah model, hubungan sebagai alat komunikasi dari bendabenda, dan diagram sebagai kumpulan / group dari benda-benda/things.

1) Benda/Things

Adalah hal yang sangat mendasar dalam model UML, juga merupakan bagian paling statik dari sebuah model, serta menjelaskan elemen-elemen lainnya dari sebuah konsep dan atau fisik. Bentuk dari beberapa benda/thing adalah sebagai berikut:

Pertama, sebuah kelas yang diuraikan sebagai sekelompok dari object yang mempunyai atribut, operasi, hubungan yang

semantik. Sebuah kelas mengimplementasikan 1 atau lebih interface. Sebuah kelas dapat digambarkan sebagai sebuah persegi panjang, yang mempunyai sebuah nama, atribute, dan metoda pengoperasiannya, seperti terlihat dalam Gambar 7.2.



Gambar 7.2 Sebuah kelas dari Model UML

Kedua, menggambarkan interface/antarmuka merupakan sebuah antarmuka yang menghubungkan dan melayani antar kelas dan atau elemen. Interface mendefinisikan sebuah set/kelompok dari spesifikasi pengoperasian, umumnya digambarkan dengan sebuah lingkaran yang disertai dengan namanya. Sebuah antar-muka berdiri sendiri dan umumnya merupakan pelengkap dari kelas atau komponen, seperti dalam Gambar 7.3.

Gambar 7.3 sebuah interface / antar muka

Ketiga, collaboration yang didefinisikan dengan interaksi dan sebuah kumpulan / kelompok dari kelas-kelas/elemen-elemen yang bekerja secara bersama-sama. Collaborations mempunyai struktura dan dimensi. Pemberian sebuah kelas memungkinkan berpartisipasi didalam beberapa collaborations dan digambarkan dengan sebuah 'elips' dengan garis terpotong-potong.

Gambar 7.4 Collaborations

Keempat, sebuah ‘use case’ adalah rangkaian/uraian sekelompok yang saling terkait dan membentuk sistem secara teratur yang dilakukan atau diawasi oleh sebuah aktor. ‘use case’ digunakan untuk membentuk tingkah-laku benda/ things dalam sebuah model serta di realisasikan oleh sebuah collaboration. Umumnya ‘use case’ digambarkan dengan sebuah ‘elips’ dengan garis yang solid, biasanya mengandung nama, seperti terlihat dalam Gambar 7.5.

Gambar 7.5 Use case

Kelima, sebuah node merupakan fisik dari elemen-elemen yang ada pada saat dijalankannya sebuah sistem, contohnya adalaha sebuah komputer, umumnya mempunyai sedikitnya memory dan processor. Sekelompok komponen mungkin terletak pada sebuah node dan juga mungkin akan berpindah dari node satu ke node lainnya. Umumnya node ini digambarkan seperti kubus serta hanya mengandung namanya, seperti terlihat dalam Gambar 7.6.

Gambar 7.6 Nodes

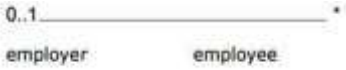
2) Hubungan/ *Relationship*

Ada 4 macam hubungan didalam penggunaan UML, yaitu; *dependency, association, generalization, dan realization.*

a) *Dependency,* yaitu hubungan semantik antara dua benda/things yang mana sebuah benda berubah mengakibatkan benda satunya akan berubah pula. Dependency digambarkan sebuah panah dengan garis terputus-putus seperti terlihat dalam Gambar 7.7.

Gambar 7.7 Dependency

b) *Association*, yaitu hubungan antar benda struktural yang terhubung diantara obyek. Kesatuan obyek yang terhubung merupakan hubungan khusus, yang menggambarkan sebuah hubungan struktural diantara seluruh atau sebagian. Assosiation digambarkan dengan sebuah garis yang dilengkapi dengan sebuah label, nama, dan status hubungannya seperti terliahat dalam Gambar 7.8



Gambar 7.8 Association

c) *Generalization*, yaitu menggambarkan hubungan khusus dalam obyek anak/child yang menggantikan obyek parent / induk . Dalam hal ini, obyek anak memberikan pengaruhnya dalam hal struktur dan tingkah lakunya kepada obyek induk. Digambarkan dengan garis panah seperti terlihat dalam Gambar 7.9.

Gambar 7.9 Generalization

d) *Realization*, yaitu hubungan semantik antara pengelompokkan yang menjamin adanya ikatan diantaranya. Hubungan ini dapat diwujudkan diantara interface dan kelas atau elements, serta antara use cases dan collaborations. Model dari sebuah hubungan realization seperti terlihat dalam Gambar 7.10.

Gambar 7.10 Realizations

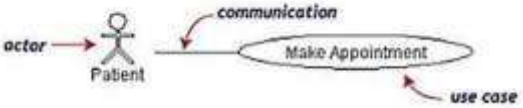
3) Diagram

UML sendiri terdiri atas pengelompokkan diagram-diagram sistem menurut aspek atau sudut pandang tertentu. Diagram adalah yang menggambarkan permasalahan maupun solusi dari permasalahan suatu model. UML mempunyai 9 diagram, yaitu; *use-case, class, object, state, sequence, collaboration, activity, component, dan deployment diagram*. Diagram pertama adalah use case menggambarkan sekelompok use case dan aktor yang disertai dengan hubungan diantaranya.

4. Use Case Diagram

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah system, menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Use case diagram menekankan kepada “apa” yang diperbuat oleh sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-create sebuah daftar belanja, dan sebagainya. Seorang atau sebuah aktor adalah sebuah entitas dapat

berupa manusia atau mesin yang berinteraksi dengan system untuk melakukan pekerjaan-pekerjaan tertentu.



Gambar 7.11 Use Case Diagram

a. Manfaat Use Diagram

Use case diagram akan sangat membantu kita dalam menyusun kebutuhan-kebutuhan (*requirement analisis)* sebuah sistem. Use case diagram akan di gunakan untuk mengkomunikasikan rancangan sistem dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

b. *Relationship* dalam *Use Case Diagram*

Didalam use case diagram terdapat dua komponen penting yang saling berinterakasi dan berkomunikasi. Komponen tersebut adalah **aktor** dan **use case**. Ragam relasi yang terjadi diantara komponen-komponen tersebut adalah :

1) *Assosiations*

   Relasi ini diigunakan untuk menghubungkan para *actor* dan *case* di dalam suatu diagram. asosiasi ini dapat digambar dalam dua arah, tergantung dari kedudukan aktornya. jika aktornya adalah sebagai aktor utama (primery actor) maka arah relasi adalah dari *actor* menuju *case*. Jika aktornya adalah sebagai aktor tambahan (*secondary actor)* maka arah relasi dari *case* menuju *actor.*

2) *Dependency*

   Relasi ini menyatakan suatu hubungan semantik antara dua unsur-unsur modeling yang mana perubahan satu elemen atau unsur model (unsur yang mengalir masuk) dapat mempengaruhi

unsur modeling yang lain (unsur yang dependent). Relasi ini menghubungkan antara use case satu dengan use case yang lain.

3) *Generalization/inheritance*

Relasi ini menyatakan hubungan hirarkis, turunan atau menyatakan suatu bagian dalam satu komponen. Sehingga aktor dapat diturunkan menjadi aktor-aktor yang lain atau use case dapat didekomposisi atau diturunkan menjadi use case yang lain. Relasi ini menunjukkan bahwa actor yang satu merupakan spesialisasi dari actor yang lain. Demikian juga use case satu merupakan spesialisasi dari usecase yang lain.

c. Karakteristik Use Case Diagram

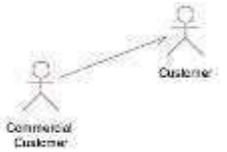
Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

d. Komponen *Use Case Diagram*

1) Aktor

Pada dasarnya *actor* bukanlah bagian dari *use case diagram*, namun untuk dapat terciptanya suatu *use case diagram* diperlukan beberapa *actor. Actor* tersebut mempresentasikan seseorang atau sesuatu (seperti perangkat, sistem lain) yang berinteraksi dengan sistem. Sebuah *actor* mungkin hanya memberikan informasi inputan pada sistem, hanya menerima informasi dari sistem atau keduanya

menerima, dan memberi informasi pada sistem. A*ctor* hanya berinteraksi dengan *use case,* tetapi tidak memiliki kontrol atas *use case*. *Actor* digambarkan dengan *stick man* . *Actor* dapat digambarkan secara secara umum atau spesifik, dimana untuk membedakannya kita dapat menggunakan *relationship.*



Gambar 7.12 Contoh aktor pada Use Case

Ada beberapa kemungkinan yang menyebabkan *actor* tersebut terkait dengan sistem antara lain:

a) Yang berkepentingan terhadap sistem dimana adanya arus informasi baik yang
b) diterimanya maupun yang dia inputkan ke sistem.
c) Orang ataupun pihak yang akan mengelola sistem tersebut.
d) *External resource* yang digunakan oleh sistem.
e) Sistem lain yang berinteraksi dengan sistem yang akan dibuat.

2) Use Case

Use case adalah gambaran fungsionalitas dari suatu sistem, sehingga *customer* atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

Catatan : *Use case diagram* adalah penggambaran sistem dari sudut pandang pengguna

sistem tersebut (*user*), sehingga pembuatan *use case* lebih dititikberatkan pada fungsionalitas yang ada pada sistem, bukan berdasarkan alur atau urutan kejadian.

Cara menentukan *Use Case* dalam suatu sistem:

a) Pola perilaku perangkat lunak aplikasi.
b) Gambaran tugas dari sebuah *actor*.

c) Sistem atau “benda” yang memberikan sesuatu yang bernilai kepada *actor*.
d) Apa yang dikerjakan oleh suatu perangkat lunak (*bukan bagaimana cara mengerjakannya).

Gambar 7.13 Contoh Use Case

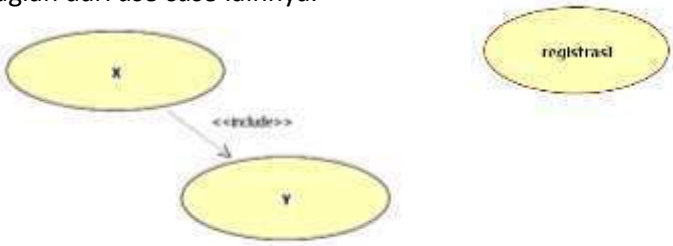
3) Relasi dalam Use Case

Ada beberapa relasi yang terdapat pada *use case diagram*:

a) *Association*, menghubungkan link antar element.
b) *Generalization*, disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.
c) *Dependency*, sebuah element bergantung dalam beberapa cara ke element lainnya.
d) *Aggregation*, bentuk assosiation dimana sebuah elemen berisi elemen lainnya.

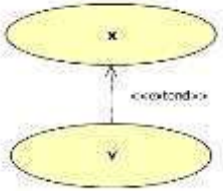
Sedangkan tipe relasi/ *stereotype* yang mungkin terjadi pada *use case diagram*:

a) **<<include>>** , yaitu kelakuan yang harus terpenuhi agar sebuah *event* dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya.



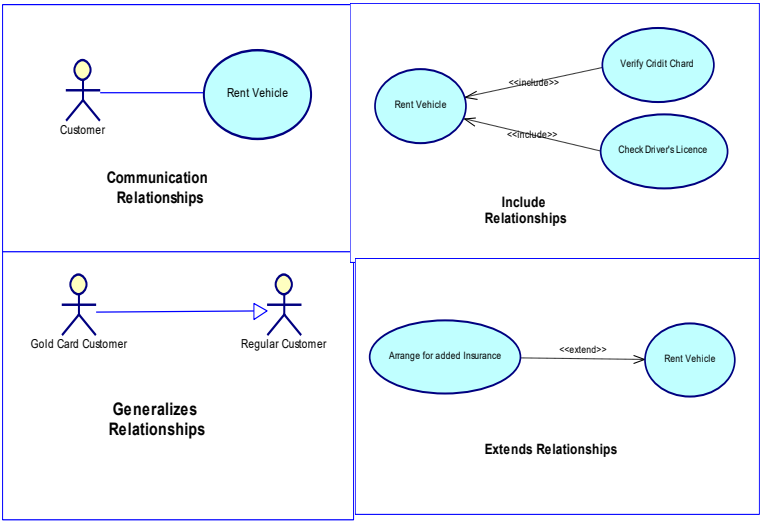
Gambar 7.14 Contoh relasi include

b) **<<extends>>**, kelakuan yang hanya berjalan di bawah kondisi tertentu seperti menggerakkan alarm.



Gambar 7.15 Contoh relasi extends

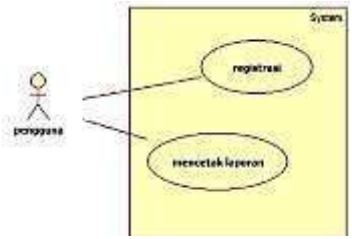
c) <<**communicates**>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah *communicates association* . Ini merupakan pilihan selama asosiasi hanya tipe *relationship* yang dibolehkan antara *actor* dan *use case*.



Gambar 7.16 Contoh seluruh relasi pada use case

4) Use *Case Diagram*

Use Case Diagrm adalah gambaran graphical dari beberapa atau semua *actor, use case,* dan interaksi diantaranya yang memperkenalkan suatu sistem.

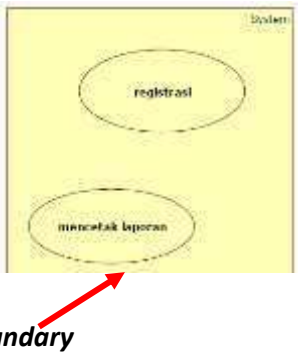


Gambar 7.17 Contoh Use Case Diagram

5) Batasan Sistem (*System Boundary*)

Dalam batasan sistem hal-hal yang penting yaitu:

a) *System boundary*.
b) Mendefinisikan batas antara actor dan sistem.
c) Dinotasikan bujur sangkar/persegi.
d) Semua use case tercakup di dalamnya.

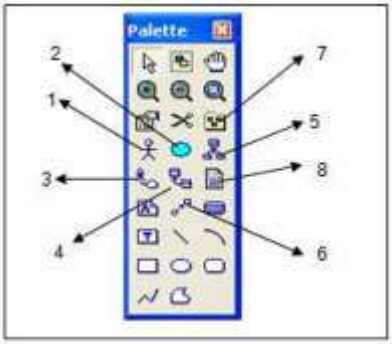


Gambar 7.18 Contoh System Boundary

e. Simbol-simbol (notasi) Use Case Diagram

Use case diagram menggunakan beberapa notasi atau simbol yang digunakan untuk menggambarkan fungsionalitas suatu sistem. Beberapa tools software menggunakan beberap notasi yang agak berbeda akan tetapi fungsionalitasnya sama. Beberapa notasi use case diagram diperlihatkan dalam Gambar 19.

1. Actor
2. Use case
3. assosiasi
4. dependency
5. Generalization/inheritance
6. Link/extended Dependencya
7. package/kumpulan
8. File



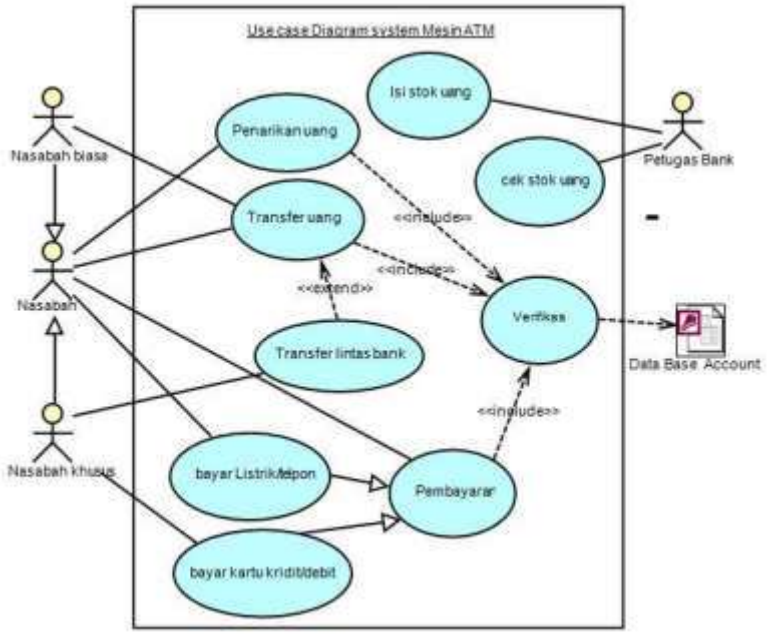
Gambar 7.19 Simbol pada use case Diagram

f. Contoh Use Case Diagram

Salah satu contoh sistem yang akan dijelaskan disini adalah ”Sistem mesin ATM”. Gambaran tentang proses business atau diskripsi persyaratan fungsionalitas dari sistem tersebut adalah sebagai berikut:

1) Nasabah dapat malakukan penarikan uang secara tunai
2) Nasabah dapat melakukan transaksi pembayaran seperti tagihan listrik, telepon, dll
3) Nasabah khusus dapat melakukan pembayaran kartu debit atau kredit
4) Nasabah biasa hanya dapat melakukan transfer uang ke rekening pada bank yang sama.
5) Nasabah khusus dapat melakukan transfer ke rekening pada bank yang berbeda

6) Untuk setiap transaksi, Sistem akan melakukan verifikasi ke database account
7) Petugas bank dapat mengecek persediaan uang di setiap ATM
8) Petugas bank mengisi ulang persediaan uang di mesin ATM



Gambar 7.20 Use case diagram sistem Mesin ATM

5. Activity Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Activity diagram merupakan state diagram khusus, di mana sebagian besar state adalah action dan sebagian besar transisi di-trigger oleh selesainya

state sebelumnya (internal processing). Oleh karena itu activity diagram tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Activity diagram digunakan untuk menggambarkan langkah-langkah atau aktivitas pada suatu sistem. Pada setiap use case yang ada, maka terdapat paling sedikit satu activity diagram. Diagram ini menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Activity diagram dipakai pada business modeling untuk memperlihatkan urutan aktivitas proses bisnis. Struktur diagram ini mirip *flowchart* atau *Data Flow Diagram* (DFD) pada perancangan terstruktur. Sangat bermanfaat apabila kita membuat diagram ini terlebih dahulu dalam memodelkan sebuah proses untuk membantu memahami proses secara keseluruhan.

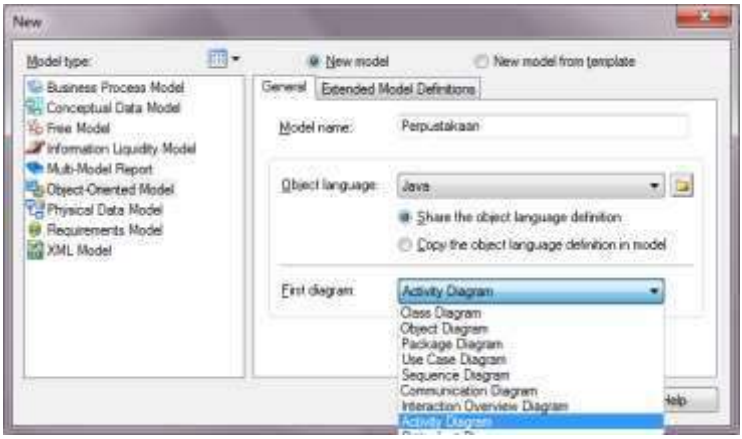
Activity diagram dibuat berdasarkan sebuah atau beberapa use case pada use case diagram. *Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, mulai dari mana berawal ,kemungkinan yang bisa terjadi dan bagaimana berakhirnya serta menggambarkan proses yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* tidak menggambarkan perilaku internal sebuah sistem tetapi lebih menggambarkan proses-proses dan jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

a. Langkah-langkah membuat Activity Diagram

Langkah-langkah yang dilakukan untuk membuat diagram activity adalah:

1) Buka aplikasi power designer
2) Dari menu pilih file → new atau menekan ctrl + N
3) Pilih *Object-Oriented Model* dan pada First diagram pilih *Activity diagram*

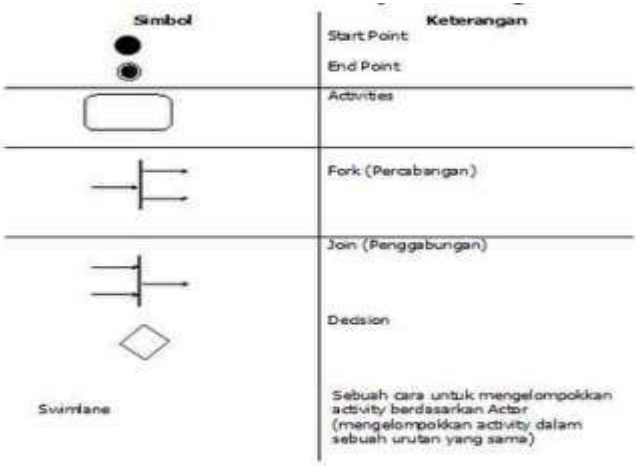


Gambar 7.21 *Langkah awal Activity diagram*

4) Berikut ini merupakan *Palette Power Designer* yang digunakan untuk membuat *activity diagram*.

Gambar 7.22 *Palette Power Designer*

Beberapa simbol yang digunakan untuk membuat *activity diagram*, yaitu:



Gambar 7.23 Simbol dalam activiy diagram

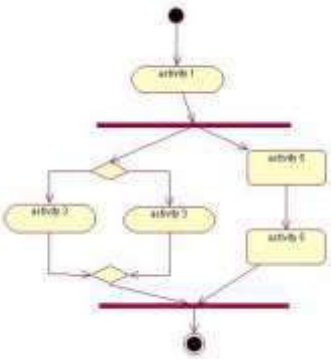
Terdapat beberapa hal penting yang harus diketahui, yaitu:

1) Activity mengambarkan sebuah pekerjaan atau tugas dalam workflow
2) Pada UML, activity digambarkan dengan simbol kotak
3) Start state dengan tegas menunjukan dimulainya suatu workflow pada sebuah activity diagram
4) Hanya ada satu start state dalam sebuah *workflow*
5) Pada UML, *start state* digambarkan dengan simbol lingkaran yang solid

6) *End state* menggambarkan akhir atau terminal dari pada sebuah *activity diagram*
7) Bisa terdapat lebih dari satu end state pada sebuah *activity diagram*
8) Pada UML, *end state* digambarkan dengan simbol sebuah *bull's eye*
9) *State transition* menunjukan kegiatan apa berikutnya setelah suatu kegiatan sebelumnya.
10) Pada UML, *state transition* digambarkan oleh sebuah *solid line* dengan panah

11) *Decision* adalah suatu titik atau *point* pada *activity diagram* yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi
12) Pada UML, *decision* digambarkan dengan sebuah simbol *diamond*/wajik.

Contoh *activity diagram* sebagai berikut :



Gambar 7.24 *Activity diagram*

Setelah membuat *use case diagram* (DFD & ERD) guna menggambarkan aliran data dan sistem database yang digunakan, maka langkah selanjutnya adalah menggambarkan langkah-langkah atau aktivitas dari sistem tersebut yaitu dengan membuat *activity diagram*.

Berikut ini adalah contoh langkah-langkah atau prosedur untuk membuat *activity diagram* dari sebuah studi kasus.

b. Studi kasus : Membuat *activity diagram* perpustakaan

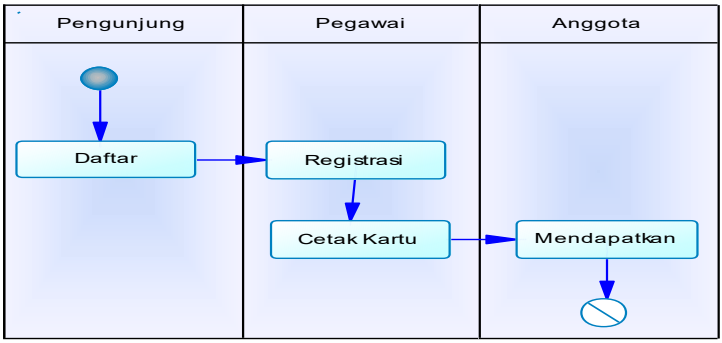
1) Proses pendaftaran anggota perpustakaan

Jika pengunjung perpustakaan ingin meminjam buku maka harus menjadi anggota, yang harus dilakukan pengunjung yaitu mendaftar, kemudian pustakawan meregristrasi lalu mencetak kartu anggota, setelah itu pustakawan memberikan kartu anggota, maka pengunjung sudah menjadi anggota dan dapat meminjam buku. Contoh scenario pendaftaran anggota perpustakaan dapat dilihat pada Tabel 7.1

Tabel 7.1 Skenario Pendaftaran Anggota

Name	**Pendaftaran**
Aktor	Pegawai, Pengunjung, Anggota
Purpose	Melakukan pendaftaran anggota perpustakaan
Typical Course of Events	
Actor Action	**System Response**
1. Pengunjung pengunjung mendaftar dengan mengisi blanko biodata 2. Pegawai membuka menu pendaftaran anggota perpustakaan 3. Pegawai meregristrasi 4. Pegawai mencetak kartu	4. sistem menampilkan form pendaftaran anggota perpustakaan 5. Sistem menerima input data anggota

Pada aktifitas diagram diatas yaitu seorang mahasiswa yang ingin meminjam buku di perpustakaan kampus, tetapi mahasiswa tersebut belum mempunyai member atau belum pernah meminjam buku sama sekali dari perpus, oleh karena itu perlu adanya pendaftaran identitas si peminjam. Diagram aktifitas pendaftaran anggota dapat dilihat pada Gambar 7.24



Gambar 7.25. Aktifitas Diagram Pendaftaran Anggota

2) Proses peminjaman buku

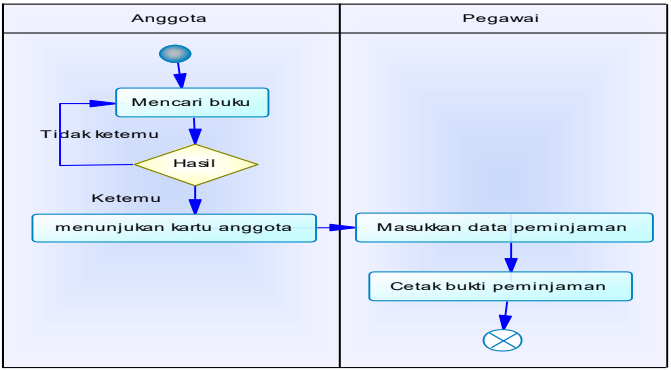
Sebelum anggota meminjam buku diperpustakaan, anggota harus membawa kartu dan menunjukkannya kepada pustakawan. Pustakawan akan mengecek kartu, mengecek buku yang akan dipinjam jika tidak cocok maka selesai, jika cocok atau sesuai maka pustakawan memberikan buku kepada anggota untuk dipinjam lalu prosespun selesai. Contoh scenario peminjaman buku perpustakaan dapat dilihat pada Tabel 7.2

Tabel 7.2 Skenario Peminjaman Buku

Name	**Peminjaman buku**
Aktor	Pegawai, Anggota
Purpose	Melakukan peminjaman buku
Typical Course of Events	
Actor Action	**System Response**

1. Anggota mencari data buku 2. Menunjukkan kartu anggkota 3. Pegawai meinput data anggota dan buku dalam menu peminjaman	4. sistem menampilkan form peminjaman 5. Sistem menerima input data peminjaman

Diagram aktifitas peminjaman buku dapat dilihat pada Gambar 7.26



Gambar 7.26. Diagram aktifitas peminjaman buku

3) Diagram pengembalian buku

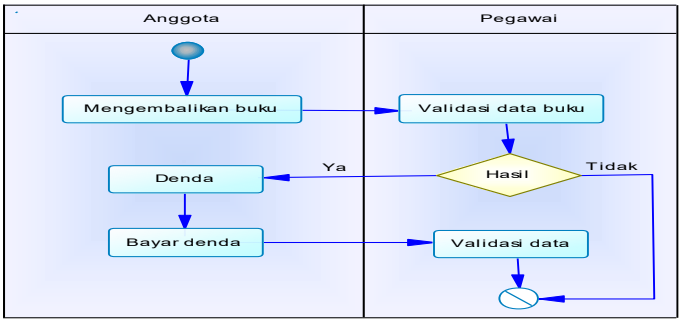
Contoh scenario pengembalian buku perpustakaan dapat dilihat pada Tabel 7.3

Tabel 7.3 Skenario Pengembalian Buku

Name	**Pengembalian**
Aktor	Pegawai, Anggota
Purpose	Melakukan pengembalian buku
Typical Course of Events	
Actor Action	**System Response**

1. Anggota membawa buku yang telah dipinjam kepada pegawai 2. Anggota menyerahkan kartu anggota 3. Pegawai memeriksa data-data anggota yang meminjam 4. Pegawai mengecek buku yang dipinjam 5. Pegawai mengupdate data anggota	6. sistem menampilkan form pengembalian 7. Sistem menerima input data pengembalian 8. Sistem memvalidasi data meliputi data buku, masa waktu pengembalian.
Alternative Course 9. jika melewati masa pengembalian maka anggota wajib membayar denda	

Diagram aktifitas pemngembalian buku dapat dilihat pada Gambar 7.27



Gambar 7.27 Diagram aktifitas pemngembalian buku

4) Proses pembayaran denda

Jika anggota perpustakaan telat mengembalikan buku yang dipinjam maka anggota mendapat denda. prosesnya yaitu anggota menunjukkan kartu anggota kemudian pustakawan memvalidasi atau mengecek data, mengecek buku yang dipinjam anggota. Jika anggota meminjam buku sesuai waktu peminjaman maka proses selesai jika tidak atau telat maka anggota dikenai denda, pustakawan menentukan jumlah denda yang harus dibayar oleh anggota, pustakawan memvalidasi data setelah anggota membayar denda lalu proses selesai.

Kesimpulan

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah system, menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Use case diagram menekankan kepada “apa” yang diperbuat oleh sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram digunakan untuk menggambarkan langkah-langkah atau aktivitas pada suatu sistem. Pada setiap use case yang ada, maka terdapat paling sedikit satu activity diagram.

Evaluasi

Setelah menyimak materi pemodelan UML, silahkan jawab pertanyaan berikut:

1. Apa fungsi penggambaran diagram *use case* dalam analisis berorientasi objek?
2. Kapan sebaiknya diagram sekuen, diagram kolaborasi, diagram komponen, dan diagram *deployment* digunakan dalam analisis dan desain berorientasi objek? Sebutkan alasannya!
3. Bandingkan studi kasus perancangan menggunakan DFD dan UML, sebutkan kelebihan dan kekurangan masing-masing metode tersebut!
4. Buat perancangan antarmuka sistem informasi manajemen perpustakaan!
5. Implementasikan hasil analisis dan desain studi kasus sistem informasi manajemen perpustakaan!

Daftar Pustaka

Kendall, Kenneth E. and Kendall, Julie E., 2011, *System Analysis and Design 8th ed*, Prentice Hall, New Jersey.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Sprague, R.H. and McNurlin, B.C., ***Information Systems Management in Practice, 5th edition,*** Prentice-Hall, 2002.

Suendri. 2018. Implementasi Diagram UML pada Perancangan Sistem Informasi Remunerasi Dosen dengan Database Oracle. 3(1).

Sukamto, Rosa A., & Shalahuddin, M. 2016. Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object. Bandung: Informatika

Sulistyorini, Prastuti. 2009. Pemodelan Visual dengan Menggunakan UML dan Rational Rose. Jurnal Teknologi Informasi Dinamik. 14(1).

Bab VIII
Manajemen Proyek Perangkat Lunak

Setelah mempelajari bab **Manajemen Proyek Perangkat Lunak**, maka diharapkan :

3. Mahasiswa mampu menerapkan perencanaan proyek
4. Mahasiswa mampu menerapkan pengujian perangkat lunak

1. Perencanaan Proyek RPL

Proyek adalah urutan kegiatan yang unik, kompleks, dan saling terkait, memiliki satu tujuan, dan tujuan harus terselesaikan dalam waktu tertentu, sesuai anggaran, dan memenuhi spesifikasi. Manajemen/ pengelolaan proyek perangkat lunak bertujuan agar perangkat lunak dibuat sampai ke tangan pelanggan tepat waktu dan sesuai dengan harapan pelanggan. Proyek dilaksanakan dengan tujuan untuk optimasi penggunaan sumber daya guna mencapai tujuan yang telah ditetapkan. Untuk mencapai tujuan tersebut manajemen proyek harus dilaksanakan dengan cara sebagai berikut:

a. Adanya koordinasi horisontal antar pelaksana yang tidak terlalu birokratis, sehingga pelaksanaan kegiatan dapat secara luwes dan cepat dilakukan antipasi bila terjadi penyimpangan.
b. Adanya penanggung jawab tunggal, biasanya oleh pimpinan proyek yang berfungsi sebagai pusat informasi, integrator antar komponen yang terlibat dan sekaligus pelaksanaan koordinasi dengan pihak diluar proyek.

Proyek dapat diuraikan menjadi rincian kegiatan yang terstruktur, dimana setiap kegiatan dapat diuraikan menjadi elemen-elemen kegiatan yang mandiri dengan sifat-sifat, yaitu sebagai berikut: (1) Dapat dikelola sebagai suatu paket kerja, (2) Beban biaya dan waktu dapat diukur, (3) Prestasi, biaya dan kualitas dapat diukur,

(4) Dapat diintegrasikan menjadi suatu satuan kegiatan, dan (5) Dapat disusun secara hirarki berjenjang. Macam-macam proyek perangkat lunak yang dapat diperoleh adalah (a). pengadaan perangkat keras, (b) pengembangan perangkat lunak, (c) pemeliharaan perangkat lunak, dan (d) konsultasi IT.

Manajemen proyek perangkat lunak sangat dibutuhkan karena permasalahan yang sering timbul pada proyek perangkat lunak tanpa pengelolaan yang baik adalah pembengkakan biaya proyek dan waktu pengerjaan tidak sesuai rencana. Padahal waktu dan biaya biasanya sudah dianggarkan oleh pelanggan dan developer perangkat lunak harus mampu mengelola agar target perangkat lunak yang akan dibuat dapat tercapai.

Aktivitas manajemen perangkat lunak dijabarkan sebagai berikut:

a. Aktifitas Manajemen

Aktivitas manajemen perangkat lunak meliputi beberapa langkah yang terstruktur, langkah-langkah tersebut adalah sebagai berikut.

1) Proposal Writing (Pembuatan Proposal). Pimpinan proyek harus membuat rencana pekerjaan proyek yang akan dilakukan dari persiapan awal hingga selesainya proyek tersebut. Persiapannya meliputi, tujuan dan maanfaat dijalankannya proyek,apa saja bentuk kegiatan yang dikerjakan,dan tahapan-tahapan pekerjaan.
2) Project Costing (Anggaran Proyek). Budged pengeluaran dan pemasukan proyek yang akan dikerjakan perlu dibuat yang serinci mungkin.
3) Project Planning and Scheduling (Penjadwalan dan Perencanaan Proyek).Perencanaan pelaksanaan proyek yang baik harusnya menggunakan jadwal yang tersusun rapi, dan penjadwalan tersebut dikonversi dengan seluruh kegiatan yang

akan dikerjakan dari study kelayakan, perencanaan, sampai implementasi dan maintenance proyek.

4) Project Monitoring and Review (Pemonitoran Proyek). Memonitor pelaksanaan proyek perlu dilakukan disetiap tahapan, sehingga kesalahan dan keterlambatan penyelesaian proyek dapat diketahui sedini mungkin.
5) Personal selection and evaluation (Evaluasi dan penyeleksi Personal). Sebelum dilaksanakannya sebuah proyek, maka personal yang terlibat dalam proyek, harus diseleksi sesuai dengan keterampilan dan pengalaman yang dimilikinya.
6) Report Writing and Presentation (Presentasi dan Laporan). Presentasi proposal proyek perlu dilakukan dengan menunjukkan prototype yang ada, sehinnga pihak manajemen yakin akan proyek tersebut

b. Perencanaan Struktur Organisasi

Pembentukan tim diperlukan untuk melakukan pelaksanaan sebuah proyek yang akan melaksanakan pekerjaan proyek pada setiap tahapan, anggota tim ini dapat juga dirotasi bila ada anggota tim yang merasakan kesulitan pada tahapan pengerjaannya. Tugas-tugas yang terbagi dari penyusunan struktur organisasi, dapat dijabarkan sebagai berikut:

a. *Project Manager* mempunyai tanggung jawab dan tugas yang bermacam-macam, tidak hanya terfokus pada hal-hal yg teknis sifatnya. Bagaimana layaknya seorang project manager harus mempunyai kemampuan membuat tim proyek agar tetap solid, mampu memonitor dan mengontrol budget serta mempunyai kemampuan analisis resiko yang baik.
b. *System Analys* bertugas untuk mengembangkan definisi sistem dan planning project.

c. *Designer* bertugas merancang produk yang sesuai dengan project yang sedang dikerjakan.
d. *Programmer* bertugas untuk membuat program-program yang sesuai dengan project timnya, yang nantinya program ini akan berpengaruh terhadap pengerjaan project tersebut.
e. *Tester* atau Penguji bertugas untuk melakukan uji per unit sistem apakah program yang dijalankan sesuai dengan apa yang diharapkan.

Tujuan perencanaan yaitu untuk menyediakan kerangka kerja yang memungkinkan manajer membuat estimasi yang dapat dipertanggungjawabkan mengenai sumber daya, biaya dan jadwal. Estimasi dibuat dengan sebuah kerangka waktu terbatas pada awal sebuah proyek perangkat lunak dan secara teratur diperbaharui secara teratur selagi proyek sedang berjalan.

c. Penjadwalan Proyek

Hal-hal yang perlu dilakukan pada penjadwalan proyek adalah : (a). Membagi kerja proyek menjadi lebih kecil dan memperkirakan waktu, sumber daya, dan personel untuk melakukan bagian kerja, (b) mengatur urutan pembagian kerja, (c) meminimalisir kebergantungan setiap bagian kerja agar tidak terjadi banyak waktu kosong (delay) karena saling menunggu bagian kerja lain selesai terlebih dahulu, (d) penjadwalan yang baik bergantung pada intuisi dan pengalaman pengelola proyek.

Permasalahan yang sering timbul pada saat penjadwalan proyek perangkat lunak, diantaranya : (a) memperkirakan waktu, sumber daya, dan biaya bukanlah hal yang mudah, (b) pembagian kerja tidak proporsional pada personel, (c) penambahan orang ke dalam proyek di tengah atau di belakang jalannya proyek dapat membuat proyek tidak tepat waktu karena memerlukan banyak komunikasi untuk

membuat orang yang baru masuk mengerti permasalahan yang terjadi, dan (d) masalah tidak terduga yang terjadi.

d. Manajemen Resiko

Manajemen resiko focus pada menidentifikasi resiko dan membuat perencanaan yang dapat meminimalisir resiko yang mungkin terjadi pada proyek perangkat lunak. Resiko yang biasanya muncul pada proyek perangkat lunak adalah:

a. Resiko proyek, yang mungkin terjadi adalah pergantian orang di dalam proyek, perubahan pengelolaan, dan tidak tersedianya perangkat keras.
b. Resiko produk, yang mungkin terjadi adalah perubahan spesifikasi/kebutuhan produk, keterlambatan waktu pengumpulan kebutuhan, besarnya sistem tidak dapat diperkirakan, dan perangkat pendukung proyek tidak ada.
c. Resiko bisnis, yang mungkin terjadi adalah perubahan teknologi dan persaingan produk.

Aktifitas yang dilakukan pada manajemen resiko adalah sebagai berikut:

a. Identifikasi resiko, mengidentifikasi resiko proyek, resiko produk, dan resiko bisnis
b. Analisis resiko, memperkirakan konsekuensi dari resiko yang mungkin terjadi
c. Perencanaan resiko, membuat perencanaan untuk meminimalisir resiko
d. Pengawasan resiko, membuat mekanisme pengawasan resiko sepanjang proyek terjadi.

e. Spektrum Manajemen

Manajemen perangkat lunak berfokus dengan tiga unsur yaitu manusia, masalah, dan proses.

1) Manusia

Faktor manusia sangat penting sehinga Sofware Engineering Institute telah mengembangkan sebuah model kematangan kemampuan manajemen manusia untuk mempertinggi kesiapan kesiapan orgnisasi perangkat lunak untuk mengerjakan aplikasi yang semakin kompleks dengan membantu menarik, menumbuhkan memotivasi, menyebarkan dan memelihara dan memelihara bakat yang dibutuhkan mengembangkan kemampuan perkembangn perangkat lunak mereka. Model kematangan manajemen manusian membatasi area praktis berikut kunci bagi masyarakat perangkat lunak: rekruitmen, seleksi, manajemen untuk kerja, pelatihan, konpensasi, perkembangan karir, disain, kerja dan organisasi, dan perkembangan tim atau kultur.

2) Masalah

Seorang manajer proyek perangkat lunak dihadapkan pada sebuah dilema pada awal proyek rekayasa perangkat lunak. Diperlukan perkiraan kuatitatif dan rencana organisasi, tetapi informasi yang solid tidak dapat diperoleh diperoleh. Analisis yang mendetail tentang kebutuhan perangkat akn memberikan informasi memadai untuk suatu perhitungan, tetapi analisis sering memerlukan waktu berminggu-minggu atau bahkan berbulan-bulan. Lebih buruk lagi kebutuhan terkadang berubah-ubah, berubah secara reguler pada saat proyek berjalan. Seorang manajer harus bisa mengamati masalah pada awal dimulainya sebuah proyek. Pada skala minimum, ruang lingkup masalah harus dibangun dan ditentukan.

3) Proses

Proses perangkat lunak memberikan suatu kerangka kerja dimana rencana komprihensif bagi pengembangan perangkat lunak dapat dibangun. Didalam suatu industri dikenal berbagai macam proses,demikian juga halnya dengan industri perangkat lunak.Perbedaan proses yang digunakan akan menguraikan aktifitas-aktifitas proses dalam cara-cara yang berlainan.

2. Pengujian Perangkat Lunak

Pengujian sistem adalah pengujian program perangkat lunak yang lengkap dan terintegrasi. Perangkat lunak atau yang sering dikenal dengan sebutan software hanyalah satuan elemen dari sistem berbasis komputer yang lebih besar. Biasanya, perangkat lunak dihubungkan dengan perangkat lunak dan perangkat keras lainnya.

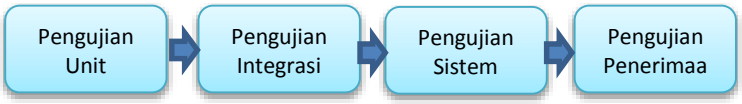
Pengujian perangkat lunak adalah proses atau rangkaian proses yang dirancang untuk memastikan bahwa program computer menjalankan apa yang seharusnya dilakukan dan sebaliknya, memastikan program agar tidak melakukan hal yang tidak diharapkan. Sebuah perangkat lunak seharusnya dapat diprediksi dan konsisten. Tugas utama dari seorang penguji adalah untuk menemukan *bug atau error* sebanyak mungkin serta mengetahui bagaimana error atau bug itu dihasilkan. Sehingga dapat dikatakan bahwa pengujian adalah proses mengeksekusi program dengan tujuan untuk menemukan *error.* Pengujian merupakan proses penting dalam siklus pengembangan software untuk memastikan kualitas dari sebuah *software.*

Proses pengujian terdiri dari beberapa aktifitas yang biasa disebut sebagai siklus hidup pengujian. Pada beberapa praktek, masing-masing aktifitas dapat dilakukan secara formal atau informal. Gambar di bawah ini menunjukkan aktifitas pengujian berdasarkan Fewster. Sebuah perangkat lunak perlu dijaga kualitasnya bahwa

kualitas bergantung kepada kepuasan pelanggan (*customer*). Kualitas perangkat lunak perlu dijaga keperluan sebagai berikut:

a. Agar dapat "*survive*" bertahan hidup di dunia bisnis perangkat lunak
b. Dapat bersaing dengan perangkat lunak yang lain
c. Penting untuk pemasaran global (*global marketing*)
d. Mengefektifkan biaya agar tidak banyak membuang perangkat lunak karena kegagalan pemasaran atau kegagalan produksi
e. Mempertahankan pelanggan (*customer*) dan mengingatkan keuntungan

Pengujian perangkat lunak adalah sebuah elemen topic yang sering dikaitkan dengan verifikasi dan validasi. Verifikasi mengacu pada sekumpulan aktifitas yang menjamin bahwa perangkat lunak mengimplementasikan dengan benar sebuah fungsi yang spesifik. Validasi mengacu pada sekumpulan aktifitas yang berbeda dan menjamin bahwa perangkat lunak yang dibangun dapat ditelusuri sesuai dengan kebutuhan pelanggan. Tahapan pengujian secara keseluruhan adalah sebagai berikut:



Pengujian perangkat lunak dapat dibedakan menjadi dua yaitu Black Box Testing dan White Box Testing**.**

a. Black Box Testing

Black Box Testing atau yang sering dikenal dengan sebutan pengujian spesifikasi fungsional merupakan metode pengujian untuk mengetahui apakah fungsi-fungsi masukan dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Dalam pengujian ini, tester menyadari apa yang harus dilakukan oleh program tetapi tidak memiliki pengetahuan tentang bagaimana

melakukannya. Misalnya menguji proses login maka kasus uji yang dibuat adalah :

a. Jika user memasukkan nama pemakai (username) dan kata sandi (password) yang benar.
b. Jika user memasukkan nama pemakai dan kata sandi yang salah, misalnya nama pemakai benar tetapi kata sandi salah atau sebaliknya, atau keduanya salah.

Gambar 8.1 Ilustrasi Black box testing

Kelebihan *Black Box Testing* yaitu:

a. Efisien untuk segmen kode besar
b. Akses kode tidak diperlukan
c. Pemisahan antara perspektif pengguna dan pengembang

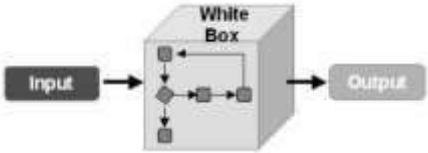
Kelemahan *Black Box Testing* yaitu:

a. Cakupan terbatas karena hanya sebagian kecil dari skenario pengujian yang dilakukan
b. Pengujian tidak efisien karena keberuntungan tester dari pengetahuan tentang perangkat lunak internal

b. White Box Testing

White Box Testing merupakan metode pengujian perangkat lunak di mana struktur internal diketahui untuk menguji siapa yang akan menguji perangkat lunak. Pengujian ini membutuhkan pengetahuan internal tentang kemampuan sistem dan pemrograman. Contoh dari pengujian white box adalah ketika

menguji alur dengan menelusuri pengulangan pada logika pemrograman.



Gambar 8.2 Ilustrasi *White Box testing*

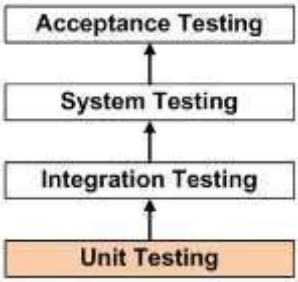
Kelebihan *White Box Testing* yaitu:

a. Efisien dalam menemukan kesalahan dan masalah
b. Diperlukan pengetahuan tentang internal perangkat lunak yang sedang diuji bermanfaat untuk pengujian menyeluruh
c. Memungkinkan menemukan kesalahan tersembunyi
d. Membantu mengoptimalkan kode

Kelemahan *White Box Testing* yaitu:

a. Membutuhkan pengetahuan tingkat tinggi dari perangkat lunak internal yang sedang diuji
b. Membutuhkan akses kode

Pengujian perangkat lunak memiliki urutan-urutan mengenai beberapa hal yang perlu dilakukan. Berikut adalah kategori pengujian perangkat lunak yang disusun secara kronologis:



a. *Unit Testing*: Pengujian dilakukan pada setiap modul atau blok kode selama pengembangan. Pengujian ini biasanya dilakukan oleh developer yang menulis kode.
b. *Integration Testing*: Pengujian yang dilakukan Sebelum, selama, dan setelah integrasi modul baru ke dalam paket perangkat lunak utama. Pengujian ini melibatkan pengujian setiap modul kode dari masing-masing individu. Satu perangkat lunak dapat berisi beberapa modul yang sering dibuat oleh beberapa developer yang berbeda.
c. *System Testing*: Pengujian yang dilakukan oleh agen pengujian profesional pada produk perangkat lunak yang telah selesai sebelum perangkat lunak tersebut diperkenalkan secara umum.
d. *Acceptance Testing*: Pengujian beta dari produk yang dilakukan oleh pengguna akhir yang sebenarnya.

Kesimpulan

Manajemen/pengelolaan proyek perangkat lunak bertujuan agar perangkat lunak dibuat sampai ke tangan pelanggan tepat waktu dan sesuai dengan harapan pelanggan. Aktivitas manajemen perangkat lunak meliputi beberapa langkah yang terstruktur, diantaranya pembuatan proposal, anggaran proyek, penjadwalan dan perencanaan proyek, monitor proyek, evaluasi dan penyeleksi personal, presentasi dan laporan. Untuk melakukan pelaksanaan sebuah proyek dibentuk tim yang akan melaksanakan pekerjaan proyek pada setiap tahapannya, anggota tim ini dapat juga dirotasi bila ada anggota tim yang merasakan kesulitan pada tahapan pengerjaannya.

Sebuah perangkat lunak perlu dijaga kualitasnya bahwa kualitas bergantung kepada kepuasan pelanggan (customer). Penting dilakukan pengujian perangkat lunak dalam siklus pengembangan software untuk memastikan kualitas dari sebuah software. Pengujian

perangkat lunak dapat dibedakan menjadi dua yaitu Black Box Testing dan White Box Testing.

Evaluasi

Setelah menyimak materi manajemen perangkat lunak, silahkan jawab pertanyaan berikut:

1. Apa pengertian manajemen proyek perangkat lunak?
2. Hal-hal apa saja yang harus dikelola dalam manajamen perangkat lunak? Jelaskan!
3. Mengapa harus dilakukan pengujian pada perangkat lunak?
4. Hal-hal apa saja yang harus diuji dalam perangkat lunak?
5. Apa perbedaan pengujian *black box* dengan pengujian *white box*? Jelaskan!

Daftar Pustaka

Browman, Kevin. 2004. *System Analysis: A Beginner's Guide*. Palgrave Macmillan

Dennis, Alan., Wixom, Barbara H. and Roth, Roberta M., 2012, *System Analysis & Design 5th ed*, John Wiley & Sons, New Jersey.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Mc.,Leod, R. Jr. 2002. System Development: A Project Management Approach. New York: Leigh Publishing LLC.

Naik, K. and Tripathy, P., 2008, *Software Testing and Quality Assurance Theory and Practice*, John Wiley & Sons, New Jersey

Pressman, R.S. 2012. Rekayasa Perangkat Lunak: Pendekatan Praktisi. Yogyakarta: Penerbit Andi.

Pressman, Roger S., 2001, *Software Engineering A Practitioner's Approach 7th ed*, McGraw-Hill, New York.

Sommerville, Ian., 2011, *Software Engineering 9th ed*, Pearson Education, Boston.

Sukamto, Rosa A., & Shalahuddin, M. 2016. Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object. Bandung: Informatika

Whitten, J.L. & Bentley, L.D. 2004. System Analysis & Design Methods: Sixth Edition. New York: Mc.Graw-Hill.

Bab IX
Pemeliharaan Perangkat Lunak

Setelah mempelajari bab **Pemeliharaan Perangkat Lunak**, maka diharapkan :

3. Mahasiswa mampu menjelaskan tenik pemeliharaan perangkat lunak
4. Mahasiswa mampu menerapkan Siklus Hidup Pemeliharaan Sistem (SMLC)

Pemeliharaan perangkat lunak adalah proses umum pengubahan/pengembangan perangkat lunak setelah diserahkan ke konsumen. Perubahan dapat berupa perubahan sederhana untuk membetulkan error koding atau perubahan yang lebih ekstensif untuk membetulkan error perancangan/perbaikan signifikan untuk membetulkan error spesifikasi/akomodasi persyaratan baru. Atau dapat didefinisikan sebagai suatu kombinasi dari berbagai tindakan yang dilakukan untuk menjaga suatu sistem dalam, atau memperbaikinya sampai, suatu kondisi yang bisa diterima.

Pada bulan April 1970 didefinisikan sebuah istilah untuk Teknologi Pemeliharaan yang mencakup pengertian yang lebih luas dari pada pengertian Pemeliharaan diatas. Istilah ini adalah Teroteknologi merupakan siklus terakhir dari SDLC yaitu dengan pemeriksaan periodik, audit dan permintaan pengguna akan menjadi source untuk melakukan perawatan system diseluruh masa hidup system. Istilah pemeliharaan perangkat lunak digunakan untuk menjabarkan aktivitas dari analis sistem (*software engineering*) yang terjadi pada saat hasil produk perangkat lunak sudah dipergunakan oleh pemakai (user).

Biasanya pengembangan produk perangkat lunak memerlukan waktu antara 1 sampai dengan 2 tahun, tetapi pada fase

pemeliharaan perangkat lunak menghabiskan 5 sampai dengan 10 tahun. Aktivitas yang terjadi pada fase pemeliharaan antara lain: (1) Penambahan atau peningkatan atau juga perbaikan untuk produk perangkat lunak; (2) adaptasi produk dengan lingkungan mesin yang baru; (3) Pembetulan permasalahan yang timbul.

" Pemeliharaan sistem berawal begitu sistem baru menjadi operasional dan berakhir masa hidupnya " .

Tujuan dari pemeliharaan system adalah :

a. Untuk memperpanjang usia kegunaan asset dari system tersebut. Hal ini penting dilakukan terutama dinegara berkembang karena kurangnya sumber daya modal untuk penggantian. Dinegara-negara maju kadang-kadang lebih menguntungkan untuk 'mengganti' daripada 'memelihara'.
b. Untuk menjamin ketersediaan optimum peralatan
c. Untuk menjamin kesiapan operasional dari seluruh peralatan yang diperlukan dalam keadaan darurat setiap waktu.
d. Untuk menjamin keselamatan orang yang menggunakan sarana tersebut.

1. Teknik Pemeliharaan Perangkat lunak

Pemeliharaan perangkat lunak dibedakan menjadi *corrective maintenance, adaptive maintenance, perfective maintenance,* dan *perfentive maintenance.*

a. Corrective Maintenance

Pemeliharaan ini untuk merespon terjadinya kesalahan-kesalahan saat produk dioperasikan baik berupa bug atupun berupa output yang tidak sesuai dengan kebutuhan pengguna.

b. Adaptive Maintenance

Pemeliharaan ini untuk merespon perubahan yang terjadi pada lingkungan yang mempengaruhi perangkat lunak tersebut

(seperti perangkat keras, sistem operasi, prosedur bisnis, kebijakan, dll).

c. *Perfective maintenance*
   Pemeliharaan ini untuk merespon permintaan tambahan berupa fungsi-fungsi baru yang muncul setelah pengguna melakukan uji coba perangkat lunak tersebut.
d. *Preventif maintenance*
   Pemeliharaan ini dilakuan untuk melakukan reengineering terhadap perangkat lunak agar lebih mudah diperbaiki, memiliki tingkat adaptasi yang tinggi dan mudah mengakomodasi munculnya kebutuhan baru.

Karakteristik perangkat lunak yang mudah dalam pemeliharaan adalah Perangkat lunak dikerjakan per modul, perangkat lunak mempunyai kejelasan, dokumentasi internal yang baik dan jelas, dan dilengkapi dokumen-dokumen pendukung lainnya.

Pemeliharaan juga mempengaruhi dokumen pendukung seperti :

a. Dokumen spesifikasi kebutuhan perangkat lunak
b. Dokumen rancangan
c. Dokumen rencana pengujian
d. Prinsip pengoperasian
e. Petunjuk pemakaian

Manfaat pemeliharaan perangkat lunak yaitu : (1) memastikan kesesuaian dengan kebutuhan fungsionalitas teknis software, (2) memastikan kesesuaian kebutuhan pihak manajerial mengenai jadwal dan budget, (3) dapat meningkatkan efisiensi software berikut juga aktifitas pemeliharaannya.

2. Siklus Hidup Pemeliharaan Sistem (SMLC)

Tahapan-tahapan yang dilakukan pada SMLC terdiri dari :

a. Memahami Permintaan Pemeliharaan

1) Mentransformasi permintaan pemeliharaan menjadi pengubahan
2) Menspesifikasi perubahan
3) Mengembangkan perubahan
4) Menguji perubahan
5) Melatih pengguna dan melakukan test penerimaan
6) Pengkonversian dan meluncurkan operasi
7) Mengupdate Dokumen
8) Melakukan pemeriksaan Pasca implementasi

b. *Maintainability* (Kemampuan pemeliharaan sistem)

Prosedur untuk peningkatan maintainability :

1) Menerapkan SDLC dan SWDLC
2) Menspesifikasi definisi data standar
3) Menggunakan bahasa pemrograman standart
4) Merancang modul-modul yang terstruktur dengan baik
5) Mempekerjakan modul yang dapat digunakan kembali
6) Mempersiapkan dokumentasi yang jelas, terbaru dan komprehensif
7) Menginstall perangkat lunak, dokumentasi dan soal-soal test di dalam sentral repositor sistem CASE atau CMS (change management system)

Untuk teknik pemeliharaan yang biasa dilakukan pada perangkat lunak, pada dasarnya tidak ada teknik atau metode yang 100 persen pasti. Hal itu karena pemeliharaan biasanya dilakukan sebagai solusi dari masalah yang muncul. Dengan kata lain, pemeliharaan mungkin tidak diperlukan andaikata tidak ada masalah yang muncul. Akan

tetapi, ada beberapa teknik yang biasa dilakukan dalam pemeliharaan perangkat lunak. Berikut beberapa diantaranya.

Kesimpulan

Pemeliharaan perangkat lunak adalah proses umum pengubahan/pengembangan perangkat lunak setelah diserahkan ke konsumen. Aktivitas yang terjadi pada fase pemeliharaan antara lain: (1) Penambahan atau peningkatan atau juga perbaikan untuk produk perangkat lunak; (2) adaptasi produk dengan lingkungan mesin yang baru; (3) Pembetulan permasalahan yang timbul. Jenis pemeliharaan perangkat lunak dibedakan menjadi corrective maintenance, adaptive maintenance, perfective maintenance, dan perfentive maintenance.

Evaluasi

Setelah menyimak materi pemeliharaan perangkat lunak, silahkan jawab pertanyaan berikut:

1. Bagaimana menurut pendapat anda tentang pemeliharaan perangkat lunak?
2. Jelaskan perbedaan spesifik dari jenis-jenis pemeliharaan perangkat lunak!
3. Apakah yang dimaksud dnegan *Maintainability* (Kemampuan pemeliharaan sistem)?
4. Menurut anda apakah pengaruh terbesar dari pemeliharaan perangkat lunak?
5. Apakah fungsi dari *Preventif maintenance*?

Daftar Pustaka

Kendall, J.E. & Kendall, K.E. 2010. Analisis dan Perancangan Sistem. Jakarta: Indeks.

Kendall, Kenneth E. and Kendall, Julie E., 2011, *System Analysis and Design 8th ed*, Prentice Hall, New Jersey.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Mc.,Leod, R. Jr. 2002. System Development: A Project Management Approach. New York: Leigh Publishing LLC.

Naik, K. and Tripathy, P., 2008, *Software Testing and Quality Assurance Theory and Practice*, John Wiley & Sons, New Jersey

DAFTAR PUSTAKA

Browman, Kevin. 2004. *System Analysis: A Beginner's Guide*. Palgrave Macmillan

Dennis, Alan., Wixom, Barbara H. and Roth, Roberta M., 2012, *System Analysis & Design 5th ed*, John Wiley & Sons, New Jersey.

Elmasri dan Navathe. 2007. *Fundamentals of Database Systems, Fifth Edition. Boston: Pearson Education, Inc. Addison Wesley*

Fatansyah. 2012. Basis Data. Bandung:Informatika

Indrajani. 2015. *Database Design.* Jakarta: Elex Media Komputindo

Kadir, Abdul. 2008. *Belajar Database menggunakan MySQL*. Yogyakarta : Andi Offset

Kendall, J.E. & Kendall, K.E. 2010. Analisis dan Perancangan Sistem. Jakarta: Indeks.

Kendall, Kenneth E. and Kendall, Julie E., 2011, *System Analysis and Design 8th ed*, Prentice Hall, New Jersey.

Marakas, G.M. 2006. System Analysis Design: an Active Approach. New York: Mc.Graw-Hill.

Mc.,Leod, R. Jr. 2002. System Development: A Project Management Approach. New York: Leigh Publishing LLC.

Naik, K. and Tripathy, P., 2008, *Software Testing and Quality Assurance Theory and Practice*, John Wiley & Sons, New Jersey

Pressman, R.S. 2012. Rekayasa Perangkat Lunak: Pendekatan Praktisi. Yogyakarta: Penerbit Andi.

Pressman, Roger S., 2001, *Software Engineering A Practitioner's Approach 7th ed*, McGraw-Hill, New York.

Simarmata, Janner. 2010. *Rekayasa Perangkat Lunak*. Yogyakarta: Andi

Sommerville, Ian., 2011, *Software Engineering 9th ed*, Pearson Education, Boston.

Sprague, R.H. and McNurlin, B.C., ***Information Systems Management***

in Practice, 5th edition, Prentice-Hall, 2002.

Suendri. 2018. Implementasi Diagram UML pada Perancangan Sistem Informasi Remunerasi Dosen dengan Database Oracle. 3(1).

Sukamto, Rosa A., & Shalahuddin, M. 2016. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object*. Bandung: Informatika

Sulistyorini, Prastuti. 2009. Pemodelan Visual dengan Menggunakan UML dan Rational Rose. Jurnal Teknologi Informasi Dinamik. 14(1).

Whitten, J.L. & Bentley, L.D. 2004. System Analysis & Design Methods: Sixth Edition. New York: Mc.Graw-Hill.

er S., 2001, *Software Engineering A Practitioner's Approach 7th ed*, McGraw-Hill, New York.

Simarmata, Janner. 2010. *Rekayasa Perangkat Lunak*. Yogyakarta: Andi

Sommerville, Ian., 2011, *Software Engineering 9th ed*, Pearson Education, Boston.

Suendri. 2018. Implementasi Diagram UML pada Perancangan Sistem Informasi Remunerasi Dosen dengan Database Oracle. 3(1).

Sukamto, Rosa A., & Shalahuddin, M. 2016. *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Object*. Bandung: Informatika

Sulistyorini, Prastuti. 2009. Pemodelan Visual dengan Menggunakan UML dan Rational Rose. Jurnal Teknologi Informasi Dinamik. 14(1).

BIODATA PENULIS

Fitria Nur Hasanah, M.Pd. dilahirkan di Lamongan, 23 September 1987. Pada tahun 2008, penulis mendapatkan gelar Diploma Manajemen Informatika di Universitas Brawijaya, lulus Sarjana Pendidikan Teknik Informatika di Universitas Negeri Malang tahun 2011, kemudian melanjutkan gelar magister Pendidikan Kejuruan dengan konsentrasi Teknik Informatika di Universitas Negeri Malang lulus tahun 2015. Tahun 2011 penulis mengawali karirnya sebagai Guru di SMK Nasional Malang bidang Rekayasa Perangkat Lunak dan Teknik Komputer Jaringan. Selanjutnya tahun 2016 menjadi Dosen tetap di Prodi Pendidikan Teknologi Informasi Universitas Muhammadiyah Sidoarjo. Tahun 2018 menjabat sebagai Ketua Program Studi Pendidikan Teknologi Informasi sampai dengan sekarang, sekaligus sebagai Sekretaris Asosiasi Pendidikan Tinggi Informatika dan Komputer (APTIKOM) Provinsi Jawa Timur periode tahun 2020-2024. Beberapa penelitian yang pernah dilakukan oleh penulis adalah tentang model pembelajaran dan pengembangan media pembelajaran.

Rahmania Sri Untari, M.Pd lahir di Malang, 19 April 1989. Pendidikan Sarjana diselesaikan di Program Studi Pendidikan Teknik Informatika, Universitas Negeri Malang (UM) pada tahun 2011. Pendidikan S2 di Program Pascasarjana Pendidikan Kejuruan UM selesai pada tahun 2015. Pada tahun 2016, penulis melanjutkan S3 Pendidikan Kejuruan UM sampai sekarang. Pada tahun 2011 penulis memulai karirnya di SMAN 6 Surabaya sebagai guru Teknik Informatika. Selanjutnya, pada tahun 2015 penulis melanjutkan karirnya untuk menjadi Dosen Tetap di Program Studi Pendidikan Teknologi Informasi (PTI) di Universitas Muhammadiyah Sidoarjo (UMSIDA) sampai sekarang. Minat penelitian penulis adalah pada bidang pembelajaran berbasis proyek, pengembangan media pembelajaran, pendidikan kejuruan, dan bidang pendidikan lainnya.

UNIVERSITAS MUHAMMADIYAH SIDOARJO
FAKULTAS PSIKOLOGI DAN ILMU PENDIDIKAN (FPIP)
PRODI PENDIDIKAN GURU ANAK USIA DINI (PG-PAUD) TERAKREDITASI B NOMOR : 2231 SK/BAN-PT/Akred/S/VII/2017
PRODI PENDIDIKAN GURU SEKOLAH DASAR TERAKREDITASI B NOMOR : 743/SK/BAN-PT/Akred/S/III/2018
PRODI PENDIDIKAN BAHASA INGGRIS TERAKREDITASI B NOMOR : 3057/SK/BAN-PT/Akred/S/XI/2018
PRODI PENDIDIKAN ILMU PENGETAHUAN ALAM (IPA) TERAKREDITASI B NOMOR : 432/SK/BAN-PT/Akred/S/III/2019
PRODI PENDIDIKAN TEKNOLOGI INFORMASI (TI) SK NOMOR : 0207/SK/BAN-PT/Akred/S/I/2017
PRODI PSIKOLOGI TERAKREDITASI B NOMOR : 0124/SK/BAN-PT/Akred/S/III/2016
TERAKREDITASI INSTITUSI (UNIVERSITAS) B
BAN-PT No. 225/SK/BAN-PT/Akred/PT/2019
Kampus I : Jl. Mojopahit 666 B Sidoarjo, Telp: 031 – 8945444, Fax: 031-8949333
Website: www.fpip.umsida.ac.id email:fpip@umsida.ac.id

SURAT TUGAS

Nomor: 205/II.3.AU/08.00/B/KEP/X/2020

Yang bertanda tangan di bawah ini;

Nama	: Dr. Akhtim Wahyuni, M.Ag
NIK	: 202200
Pangkat/ Golongan	: Lektor/ IIId
Jabatan	: Dekan Fakultas Psikologi dan Ilmu Pendidikan
Unit Kerja	: Fakultas Psikologi dan Ilmu Pendidikan

Menugaskan:

N a m a	: **Fitria Nur Hasanah, M.Pd**
NIK	: 216613
Pangkat Golongan	: Asisten Ahli/ IIIb
Jabatan	: Dosen Tetap
Unit Kerja	: Fakultas Psikologi dan Ilmu Pendidikan

Untuk membuat Modul Pengembangan Soft Skill dan Konselor Sebaya di Fakultas Psikologi dan Ilmu Pendidikan Universitas Muhammadiyah Sidoarjo Semester Ganjil Tahun Akademik 2020/2021. Demikian surat tugas ini dibuat, untuk dapat dipergunakan sebagaimana mestinya.

Sidoarjo, 22 Oktober 2020
DEKAN FPIP,

Dr. Akhtim Wahyuni, M.Ag

MODUL PENGEMBANGAN LIFE SKILL DAN KONSELOR SEBAYA

Ghozali Rusyid Affandi, S.Psi., M.A
Nurfi Laili, M.Psi., Psikolog
Fitria Nur Hasanah, M.Pd
Amaliyah Syabana
Rakhmat Auliya' Hidayat

MODUL PENGEMBANGAN LIFE SKILL DAN KONSELOR SEBAYA

Penulis:

Ghozali Rusyid Affandi, S.Psi., M.A.
Nurfi Laili M.Psi., Psikolog.
Fitria Nur Hasanah M.Pd.
Amaliyah Syabana
Rakhmat Auliya` Hidayat

Diterbitkan oleh
UMSIDA PRESS
Jl. Mojopahit 666 B Sidoarjo
ISBN 978-623-6081-12-9

MODUL PENGEMBANGAN LIFE SKILL DAN
KONSELOR SEBAYA

Penulis :
Ghozali Rusyid Affandi, S.Psi., M.A.
Nurfi Laili M.Psi., Psikolog.
Fitria Nur Hasanah M.Pd.
Amaliyah Syabana
Rakhmat Auliya` Hidayat

ISBN 978-623-6081-12-9

Editor :
Mahardika DK Wardana
M. Tanzil Multazam

Copy Editor :
Nibras Ali Gunanjar

Design Sampul dan Tata Letak :
Muhammad Ilham dzulfikri

Penerbit :
UMSIDA Press

Redaksi :
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit No 666B
Sidoarjo, Jawa TImur

Cetakan pertama, Desember 2020

PENGANTAR

Alhamdulillahirobbil'aalamiin.
Segala Puja dan Puji hanya untuk Alloh SWT Yang Maha Esa penggenggam seluruh Ilmu Pengetahuan. Hanya dengan karunianyalah Modul Pengembangan Life Skill dan Konselor Sebaya ini hadir. Shalawat serta salam kita haturkan kepada junjungan Nabi Besar Muhammad SAW.
Modul ini disusun guna menjawab salah satu persolan yang dihadapi oleh Pengelola Panti Asuhan, terutama berkenaan dengan kemampuan *life skill* Anak Panti Asuhan serta penanganan yang tepat dalam menangani setiap persolan yang dihapai oleh pengurus panti dalam mengelola anak-anak yang tinggal disana. Sehingga buku ini hadir sebagai pedoman dalam pengabdian kepada masyarakat pada anak-anak Panti Asuhan Yatim 'Aisyiyah Balongbendo. Modul ini tersusun tahap demi tahap dalam setiap kegiatannya disertai dengan praktek agar materi yang disampaikan dapat diaplikasikan secara nyata oleh anak-anak panti asuhan.
Kami ucapkan terima kasih sedalam-dalamnya kepada semua pihak yang turut memberikan dukungan sehingga penyusunan modul sebagai pedoman pengabdian kepada masyarakat dapat terwujud, khususnya kepada:

1. Jajaran Rektorat Universitas Muhammadiyah Sidoarjo
2. Direktur, Kasie serta staff Direktorat Penelitian dan Pengabdian Masyarakat UMSIDA.
3. Dekan, Kapordi serta staff Prodi Psikologi Fakultas Psikologi dan Ilmu Pendidikan UMSIDA.
4. Majelis Kesejahteraan Sosial PWA Jatim dan PDA Sidoarjo.
5. Kepala Panti Asuhan Yatim 'Aisyiyah Balongendo beserta jajarannya dan juga kepada Anak-anak Panti.

6. Rekan dosen seprofesi yang turut berperan dalam kegiatan pengabdian kepada masyarakat serta mahasiswa yang membatu terselenggaranya PKM dan penulisan Modul ini.

Modul ini adalah langkah awal kami untuk memberikan sumbangsih keilmuan kepada masyarakat, tentunya masih banyak kekuarangan dan kelamahan. Oleh sebab itu, saran dan kritik yang membangun sangat kami harapkan guna menjadikan modul ini menjadi lebih baik kedepannya.

Sidoarjo, Desember 2020

Penyusun

DAFTAR ISI

Sesi 2

Sesi 3

Sesi 4

Susunan Kegiatan

Peningkatan Kapasitas *Life Skill* Anak Panti Asuhan

Yatim 'Asyiyah Balongbendo

Pelatihan tahap 1: Peningkatan Kapasitas *Life Skill*
Waktu: 09.00 – 15.00
Peserta: Anak Panti Asuhan Usia kelas 4 SD – SMA
Tempat: Panti Asuhan Yatim 'Aisyiyah Balongbendo

No.	Waktu	Kegiatan
1.	09.00-09.30	Kontrak Belajar, Ice Breaking dan Pre Test
2.	09.30-10.30	My Diary: Memahami Potensi Diri Untuk Membangun Konsep Diri Positif
3.	10.30-11.30	Yes I Can: Mengubah Keyakinan Negatif Menjadi Positif
4.	11.30-12.30	Goals Setting: Praktek Merancang dan Memilih Peta Kehidupan Yang Mencerahkan
5.	12.30-13.00	Ishoma
6.	13.00-14.00	Mengelola Emosi: Mengenal dan Mengendalikan Emosi
7.	14.00-15.00	Evaluasi dan Post Test

Pelatihan tahap 2: Konseling Sebaya Waktu: 09.00 – 15.00 Peserta: Anak Panti Asuhan Usia SMP – SMA **Tempat: Panti Asuhan Yatim 'Aisyiyah Balongbendo**		
1	09.00 - 09.30	Kontrak Belajar, Ice Breaking dan Pre Test
2	09.30 - 10.30	Active Listening: Belajar menjadi pendengar yang baik sebagai modal konseling
3	10.30 - 11.30	Empati: Mengenal dan memahami perasaan orang lain
4	11.30 - 12.00	Memahami Permasalahan Anak dan Remaja Dalam Kehidupan Sehari-hari
5	12.00 - 12.30	Ishoma
6	12.30 - 13.30	Teknik Konseling Dasar
7	13.30 - 14.30	Praktek Konseling dan Pembentukan Pusat Konseling Remaja Panti Asuhan (Pojok Konseling) serta Komunitas Konseling Sebaya
8	14.30 - 15.00	Evaluasi dan Post Test

MODUL 1

PELATIHAN PENGEMBANGAN LIFE SKILL

PRE TEST: ISI DENGAN JUJUR YA

Jawablah pertanyaan berikut dengan memilih satu dari empat pilihan jawaban yang sesuai dengan keadaan Anda yang sebenarnya dengan memberikan tanda SILANG (X). Tidak ada jawaban benar atau salah dalam semua pertanyaan berikut, melainkan yang paling sesuai dengan diri Anda.

1. Saya kesulitan membuat rencana untuk membantu saya mencapai tujuan.

	1	2	3	4	
Tidak setuju					**Setuju**

2. Saya mengalami kesulitan menetapkan tujuan untuk diri saya sendiri.

	1	2	3	4	
Tidak setuju					**Setuju**

3. Begitu saya memiliki tujuan, saya biasanya dapat merencanakan bagaimana mencapainya.

	1	2	3	4	
Tidak setuju					**Setuju**

4. Saya cepat menyerah

	1	2	3	4	
Tidak setuju					**Setuju**

5. Saya menetapkan tujuan untuk diri saya sendiri dan memantau kemajuan saya.

	1	2	3	4	
Tidak setuju					**Setuju**

6. Ketika saya mencoba untuk mengubah sesuatu, saya memperhatikan bagaimana saya melakukannya.

	1	2	3	4	
Tidak setuju					**Setuju**

7. Saya tidak memperhatikan efek dari tindakan saya sampai semuanya terlambat.

	1	2	3	4	
Tidak setuju					**Setuju**

8. Saya memiliki standar pribadi, dan mencoba untuk menjalaninya.

	1	2	3	4	
Tidak setuju					**Setuju**

9. Saya mudah teralihkan dari rencana saya.

	1	2	3	4	
Tidak setuju					**Setuju**

10. Saya mengalami kesulitan untuk menindaklanjuti hal-hal setelah saya memutuskan untuk melakukan sesuatu.

	1	2	3	4	
Tidak setuju					**Setuju**

11.Saya mampu mencapai tujuan yang saya tetapkan untuk diri saya sendiri.

Tidak setuju	1	2	3	4	**Setuju**

12. Seringkali saya tidak memperhatikan apa yang saya lakukan.

Tidak setuju	1	2	3	4	**Setuju**

13. Saya menunda membuat keputusan.

Tidak setuju	1	2	3	4	**Setuju**

14.Jika saya membuat resolusi untuk mengubah sesuatu, saya memberi banyak perhatian pada apa yang saya lakukan.

Tidak setuju	1	2	3	4	**Setuju**

15. Saya sepertinya tidak belajar dari kesalahan saya.

Tidak setuju	1	2	3	4	**Setuju**

16. Saya biasanya melacak kemajuan saya menuju tujuan saya.

	1	2	3	4	
Tidak setuju					**Setuju**

17. Begitu saya melihat masalah atau tantangan, saya mulai mencari solusi yang bisa saya lakukan.

	1	2	3	4	
Tidak setuju					**Setuju**

18. Saya belajar dari kesalahan saya.

	1	2	3	4	
Tidak setuju					**Setuju**

19. Saya kesulitan mengambil keputusan tentang berbagai hal.

	1	2	3	4	
Tidak setuju					**Setuju**

20. Saya biasanya hanya membuat kesalahan satu kali untuk belajar darinya.

	1	2	3	4	
Tidak setuju					**Setuju**

21. Saya bisa berpegang pada rencana yang sudah saya tentukan dan dapat melakukannya dengan dengan baik.

Tidak setuju	1	2	3	4	**Setuju**

22. Saya biasanya dapat menemukan beberapa alternative pemecahan masalah ketika saya ingin mengubah sesuatu.

Tidak setuju	1	2	3	4	**Setuju**

SESI 1

My Diary:

Memahami Potensi Diri Untuk Membangun Konsep Diri Positif

Waktu : 60 Menit

Metode : Psikoedukasi dan Games

Materi :

Memahami potensi diri adalah langkah pertama dalam mempelajari kecakapan hidup. Ini didasarkan pada premis bahwa jika kita tidak tahu siapa kita, kita tidak akan tahu apa yang ingin kita ubah dan apa yang ingin kita bangun.

Mengenal diri sendiri merupakan memahami pengetahuan tentang potensi diri, seperti menyadari kelebihan/keungulan maupun kekurangan/ kelemahan diri yang ada pada diri sendiri. Dengan mengenal diri sendiri secara tepat akan mengetahui konsep diri yang

tepat pula, berupaya untuk mengembangkan yang positif dan mengatasi/ menghilangkan yang negatif.

Menurut John Robert Powers (dalam Elihami, 2018), konsep diri merupakan kesadaran dan pemahaman mengenai dirinya sendiri yang meliputi ; siapa aku, apa kemampuan yang kumiliki, apa kekuranganku, apa kelebihan yang kumiliki, apa perananku, dan apa keinginanku. Kesadaran dan pemahaman akan dirinya semakin mencerminkan prinsip hidup dan kehidupannya serta menjadi dasar perilaku hidup sehari-hari.

Kemampuan dan keterampilan saya

* Tujuan :

  Anak-anak akan mampu untuk :

  - Identifikasi kekuatan mereka - apa yang mereka kuasai dan kualitas positif apa yang mereka miliki;
  - Dapatkan umpan balik dari teman-teman mereka dalam kelompok tentang kualitas positif mereka;
  - Katakan mengapa mereka senang menjadi laki-laki atau perempuan;
  - Belajar melindungi diri dari komentar negatif tentang diri mereka sendiri; dan
  - Putuskan kualitas atau keterampilan apa yang ingin mereka perkuat.

Games 1: I Love My Self

* Prosedur Pelaksanaan

1. Meminta anak-anak untuk mengatakan, "Saya mencintai diri saya sendiri." Saat mengatakan ini anak-anak harus menggunakan gerakan sederhana seperti tersenyum, memeluk diri sendiri, dan berdiri tegak dan bangga dan cara lain apa pun yang mengungkapkan arti dari apa yang mereka katakan.
2. Setiap anak diberikan kertas serta meminta mereka untuk menulis di atasnya, dua hal yang mereka sukai tentang diri mereka sendiri atau yang mereka kuasai
3. Meminta anak-anak membagikan apa yang telah mereka tulis dengan pasangan mereka di sebelah kanan atau kiri.
4. Meminta anak-anak untuk menempelkan kertas di punggung mereka dan bergerak di sekitar ruangan.

5. Dalam sebuah lingkaran, meminta anak untuk membacakan semua yang telah dia dan yang lain tulis.

Games 2: Pelindung Saya

* Prosedur Pelaksanaan

1. Membagikan perisai berupa kertas buffalo berwarna dan berbentuk bulat untuk setiap anak. Mendiskusikan dengan anak-anak untuk apa perisai digunakan. Meminta anak-anak untuk menuliskan nama mereka di tengah perisai, menuliskan hal-hal terbaik yang mereka sukai tentang diri mereka di perisai.
2. Perisai dan kertas 1 dapat dipasang di dinding sehingga anak-anak dapat melihatnya kapan pun mereka datang ke grup atau ditaruh dikamar masing-masing.

Games 3 : Saya Senang

Saya Senang Menjadi Perempuan / Saya Senang Menjadi laki-laki (Diskusi kelompok)

1. Membagi kelompok yang terdiri anak laki-laki dan perempuan.
2. Memberikan dua kartu (memberi warna yang berbeda antara perempuan dan laki-laki). Meminta setiap anak melengkapi kalimat berikut:

   “Saya perempuan / laki-laki. Saya suka menjadi perempuan / laki-laki karena…. ” Atau "Saya ahli dalam……"

   Serta meminta anak menulis tiga kualitas atau keterampilan yang mereka miliki.

AYO PRAKTEK MENGENAL DIRI

Adik-adik silahkan menuliskan tiga ketrampilan yang adik miliki.

1. ……

……

……

……..

2. ……

……

……

……..

3. ……

……

……

……..

SESI 2

Yes I Can:

Mengubah Keyakinan Negatif Menjadi Positif

Waktu : 60 Menit

Metode : Psikoedukasi dan Simulasi

Materi :

Semua hal dapat kita raih, mungkin ini adalah hal yang mustahil bagi sebagian orang, namun tidak bagi sebagain orang yang lain.

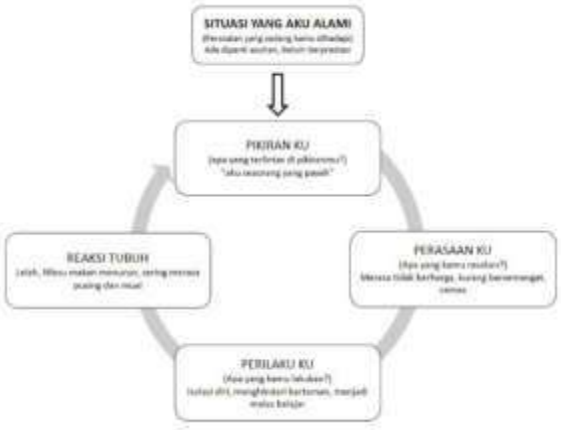
Apa yang menjadi pembeda antara keduanya?

Ya betul sekali, yang menjadi pembeda keduanya adalah keyakinan dalam diri. Keyakinan dalam diri merupakan modal utama untuk mencapai sebuah prestasi. Tanpa adanya keyakinan yang kuat, maka saat kalian dihadapkan pada tantangan, mungkin kalian akan berfikir “kok gagal lagi, gagal lagi” atau berfikir “gak mungkin hal itu bisa aku capai”.

Itulah yang sering dirasakan dan difikirkan oleh sesorang yang **pesimis.**

Berbeda dengan orang yang optimis. Mereka akan mengatakan “aku yakin aku mampu mencapai cita-citaku, halangan ini sifatnya hanya sementara jika aku terus berusaha”.

Pernahkah kalian mendengar tentang “hati-hati dengan pikiran dan perasaan Anda”? ungkapan itu adalah benar adanya. Menurut ilmu psikologi bahwa perilaku manusia itu didasari oleh pikiran dan perasaan sebagaimana gambar dibawah ini:



Persitiwa yang saat ini kita alami, seperti: berada di Panti asuhan, mendapat pekerjaan rumah, kurang berprestasi, kurang memiliki biaya untuk sekolah, tidak diterima oleh teman-teman dan persoalan lainnya akan masuk ke dalam pikiran kita, baik kita sadari maupun tidak kita sadari. Nah dari sinilah, sebuah persitiwa tersebut akan menjadi suatu yang membebani diri kita ataukah menjadi tantangan dan semangat bagi kita, itu semua tergantung dari pola pikir kita.

Jika pola pikir kita dalam menghadapi masalah tersebut dengan cara yang negatif, Seperti:

“aku orang yang payah karena tidak pernah berprestasi”

“aku anak yang tidak beruntung karena ada di Panti Asuhan”

“aku anak yang minder”

“kepada sih aku ada di Panti Asuhan, aku jadi muak berada disini”

Atau pikiran negatif lainnya.

Maka, pikiran negatif tersebut akan menjadikan perasaan atau emosi kita juga menjadi negatif, seperti:

sedih, perasaan tidak berdaya, kurang bersemangat, cemas berkepanjangan, bahkan bisa menjadikan diri kita depresi atau tertekan.

Perasaan Negatif yang mendominasi diri kita tersebut akan menggerakkan perilaku kita juga ke arah yang negatif, seperti: malas belajar, menarik diri dari teman-teman, menyendiri di kamar terus, suka marah-marah, malas beraktifitas, suka nyinyirin orang tapi tidak mau berusaha memperbaiki diri, inginnya tidur melulu dll.

Apabila hal ini terus kita lakukan, akan berbahaya bagi masa depan kita.

Lho kok berbahaya, kenapa?

Iya, karena jika perilaku negatif it uterus menerus kita lakukan akan menjadi kebiasaan bahkan akan menjadikan pribadi kita juga bersifat negatif. Bahkan akan mempengaruhi kesehatan kita, coba bayangkan jika perilaku kita setiap hari adalah pemarah atau berfikir bahwa diri kita adalah anak yang tidak berharga, maka kita akan sering mengalami sakit kepala bahkan

sering sakit maag yang disebut dengan psikosomatis (Sakit fisik karena ganggunan psikologis).

BERBEDA JIKA POLA PIKIR KITA BERSIFAT LEBIH POSITIF DALAM MENGHADAPI PERSOALAN.

Apabila, pola pikir kita lebih positif dalam mengahadapi kehidupan, seperti:

“Meskipun aku belum berprestasi, aku yakin hal ini bisa diperbaiki”

“Aku yakin meskipun aku tinggal di Panti Asuhan, aku tetap bisa berpertasi”

“Tinggal di Panti itu menyenangkan, karena banyak teman dan bisa belajar dari teman-teman”

“Aku anak yang percaya diri”

“Alhamdulillah aku masih bisa belajar dan sekolah saat tinggal dipanti”

Atau hal positif lainnya…

Maka, Persaan kita akan menjadi lebih bersemangat, menjadi lebih bangga dengan diri kita, merasa berharga, tidak malas, tidak menjadi pencemas atau tidak risau dengan kondisi yang disarakan serta lebih percaya diri.

Perasaan Positif yang mendominasi tersebut akan berdampak pada **perilaku** kita yang juga lebih **Positif dan Lebih Baik**, seperti: semangat belajar, suka bergaul, percaya diri, memiliki prestasi, tidak minder, anak yang energik, tidak menjadi pemalas, rajin beribadah dll.

Apabila perilaku yang kita lakukan setiap harinya adalah perilaku yang positif, maka hal ini akan menjadi **kebiasaan (habit) positif** bagi kita dan ini akan berdampak pada kepribadian yang lebih positif dan tangguh.

INGAT, POLA PIKIR KITA AKAN MEMPENGARUHI PERASAAN KITA, PERASAAN KITA AKAN MEMPENGARUHI PERILAKU KITA DAN PERILAKU KITA AKAN MEMPERNGARUHI KESEHATAN KITA. JIKA ITU TERUS-MENERUS DIALKUKAN AKAN MENJADI KEBIASAAN KITA DAN MEJANDI BAGIAN DARI KEPROBADIAN KITA DALAM MENGHADAPI SETIAP PERSOALAN.

PERHATIKAN FIRMAN ALLOH SWT DAN HADIST NABI MUHAMMAD SAW

Artinya: "*Alloh swt telah menurunkan perkataan yang paling baik (yaitu) Al - Quran yang serupa (mutu ayat - ayatnya) lagi berulang - ulang , gemetar karenanya kulit orang - orang yang takut kepada tuhannya , kemudian menjadi tenang kulit dan hati mereka di waktu mengingat Allah. itulah petunjuk Allah , dengan kitab itu dia menunjuki siapa yang dikehendaki - nya . dan barangsiapa yang disesatkan Allah , niscaya tak ada baginya seorang pemimpinpun***" (QS. AZ ZUMAR: 23)**

Dari Shuhaib rd ia berkata, Rasulullah saw bersabda: "*Sungguh menakjubkan perkara orang mukmin, setiap perkara baik baginya dan hal ini tidak dimiliki keculi oleh orang mukmin, jika ia diberikan kesenangan, ia bersyukur dan hal ini baik baginyA, jika ia ditimpa musibah iapun bersabar dan hal ini baik baginya***." (HR. Muslim).**

AYO PRAKTEK MERUBAH DAN GANTI FILE NEGATIF ANDA MENJADI FILE POSITIF

FILE / POLA PIKIR NEGATIF	FILE / POLA PIKIR POSITIF
* **Saya Pemalas**	✓ **Saya Anak yang semangat**
* **Saya mudah menyerah**	✓ **Tidak ada kata menyerah sebelum berhasil**
*	✓
*	✓
*	✓

*	✓
*	✓
*	✓
*	✓

SESI 3

Mengelola Emosi: Mengenal dan Mengendalikan Emosi

Waktu : 60 menit

Metode : Games

Materi :

Setiap manusia memiliki persepsi yang berbeda-beda terhadap peristiwa yang terjadi dalam hidupnya. Persepsi ini lah yang dapat menentukan respon seseorang baik itu berupa negatif atau positif. salah satu repon tersebut disebut emosi. adapun dua jenis emosi yakni emosi negatif dan positif. **Emosi negatif** mendorong seseorang melakukan tindakan-tindakan yang negatif seperti marah, takut, sedih,dengki, malu dll. emosi negatif ini juga bisa oranglain maupun diri sendiri. Sebaliknya, **emosi positif** mendorong seseorang untuk bertindak positif seperti gembira, bersyuku, dll . Tanpa disadari bahwa emosi dapat mempengaruhi perilaku manusia.

Perilaku impulsif dan terburu-buru mereka, yang dipandu oleh emosi mereka yang intens, bahkan dapat membahayakan mereka. Untuk alasan ini, penting bagi remaja, anak-anak dan remaja, belajar mengungkapkan perasaan dan emosi mereka dengan cara yang aman dan sehat yang tidak merugikan diri sendiri maupun orang lain.

Memahami perasaan kita adalah langkah pertama dalam mempelajari bagaimana memiliki kendali lebih atas perasaan kita. Semua anak perlu tahu bahwa memiliki perasaan yang kuat itu normal dan tidak ada perasaan yang "buruk". Norma budaya, pola asuh keluarga dan jenis kelamin merupakan beberapa faktor yang mempengaruhi ekspresi perasaan. Perasaan, jika dipandang sebagai "buruk" atau "salah", dapat disimpan dan ditekan. Akibatnya anak tidak mampu menyalurkan emosi tersebut dengan tepat. Keterampilan hidup membantu anak-anak mempelajari cara yang sehat, positif dan aman untuk mengekspresikan perasaan ini. Kegiatan dalam modul

ini dibatasi untuk membantu anak-anak memahami perasaan mereka dan menyarankan cara aman untuk mengungkapkannya.

AKTIVITAS MEMAHAMI PERASAAN

Tujuan

Anak-anak akan mampu untuk :

- Mengidenifikasi dan mengungkapkan perasaan yang berbeda;
- Memahami bahwa perasaan dapat diungkapkan baik secara verbal maupun non-verbal;
- Memahami bahwa perasaan berubah itu normal dan intensitasnya bisa berubah; dan
- Memahami berbagi emosi

Sesi 1: Identitas yang salah

* Prosedur Pelaksanaan

1. Panitia menyiapkan kartu dengan nama-nama perasaan seperti suka, malu, kesepian, sedih, takut, lucu, bingung, heboh, marah, senang, dan penasaran dengan menambahkan perasaan lain dan memasukkannya ke dalam tas.
2. Anak-anak duduk dalam lingkaran dan "membagikan bingkisan perasaan" saat musik dimainkan. Ketika musik berhenti, anak yang memiliki parsel mengambil kartu dari tas dan

memerankan emosi yang dijelaskan. Tidak ada kata yang bisa digunakan. Anak-anak lainnya harus menebak perasaan yang diungkapkan.

Sesi 2: Identitas yang salah

* Prosedur Pelaksanaan

1. Meminta anak-anak untuk mengungkapkan semua perasaan mereka dengan mengambar, seperti contoh : awan untuk emosi yang menyakitkan dan matahari untuk perasaan bahagia. Mereka dapat menggunakan awan monsun untuk emosi yang menyakitkan atau warna pelangi yang berbeda untuk perasaan bahagia. Ingatkan mereka bahwa ini hanya saran. Mereka bisa memberikan bentuk apapun pada perasaan mereka. Tidak ada aturan dalam latihan ini. Anak-anak harus menggambarkan perasaan mereka saat mereka merasakannya. Jika perasaannya sangat kuat, mereka bisa membuat gambar besar dan jika perasaan itu datang kadang-kadang atau tidak terlalu kuat membuat gambar yang kecil.

AYO PRAKTEK MENGUNGKAP EMOSI

Adik-adik silahkan mengungkapkan perasaan adik-adik saat ini dengan mengambar orang atau apapun. Semakin besar gambarannya melambangkan kebahagian begitupun sebaliknya jika gambaran semakin kecil maka menandakan perasaan yang sedih

SESI 4

Goal Setting:

Merancang dan Memilih Peta Kehidupan Yang Mencerahkan

Waktu : 60 Menit

Metode : Psikoedukasi dan Simulasi

Materi :

Goal setting merupakan dua kata berbahasa Inggris yaitu, *goal* artinya tujuan dan *setting* artinya penentuan, sehingga *goal setting* diartikan sebagai penentuan tujuan. Menurut Locke dkk., (1981) *goal* diartikan sebagai "*what an individual is trying to accomplish*". Konsep ini hampir sama dengan konsep tujuan dan maksud. Konsep ini juga sering dimaknai dengan tujuan yang termasuk di dalamnya adalah standar performansi (ukuran untuk evaluasi hasil performansi) (Locke dkk., 1981).

Asumsi dasar penelitian mengenai penentuan tujuan adalah bahwa tujuan (*goal*) merupakan pengatur secara langsung akan perilaku atau tindakan seseorang (Locke dkk., 1981). Penetapan tujuan yang jelas juga akan menampakkan adanya peningkatan antusiame, dan dengan adanya tujuan yang penting bagi seseorang akan mengantarkannya pada produksi energi yang besar daripada tujuan yang tidak terlalu penting (Morisano dkk., 2010).

Tujuan tidak serta merta dapat meningkatkan persepsi diri, motivasi dan performansi akademik siswa, akan tetapi tujuan harus memenuhi sifat tujuan itu sendiri yaitu *specificity, proximity* dan *difficulty* (Locke & Latham, 2002; Schunk, 2008).

Ada empat mekanisme dasar bahwa tujuan dapat mengantarkan seseorang mencapai performansi yang telah ditetapkan secara maksimal (Locke & Latham, 2002). 1) Tujuan dapat mengarahkan perhatian dan tindakan seseorang kepada tugas yang relevan untuk pencapaian prestasi. 2) Tujuan berfungsi untuk

memberikan energi yang menggerakkan atau memobilisasi pada usaha yang tinggi untuk mencapai performansi. 3) Tujuan berpengaruh terhadap ketekunan seseorang. 4) Tujuan mempengaruhi tindakan untuk pencarian, dan penggunaan pengetahuan dan strategi yang relevan terhadap tugas.

Makna Penentuan Tujuan (*Goal Setting*)

1. Setiap orang harus memiliki tujuan atau cita-cita atau mimpi dalam hidupnya, sebab;
    a. Tujuan dapat mengarahkan perhatian dan tindakan seseorang kepada tugas yang relevan untuk pencapaian prestasi.
    b. Tujuan berfungsi untuk memberikan energi, energi ini dapat membantu seseorang menggerakkan pada usaha yang tinggi untuk mencapai performansi.
    c. Tujuan berpengaruh terhadap ketekunan seseorang.

d. Tujuan mengarahkan seseorang dalam pencarian, dan penggunaan pengetahuan serta strategi yang relevan terhadap tugas.

2. Setelah seseorang memiliki tujuan yang akan dicapai, maka ia harus yakin bahwa ia akan dapat meraih sukses (memiliki efikasi diri yang tinggi). Dengan kayakinan yang tinggi akan membuatnya tetap berkomitmen meskipun banyak halangan yang merintang.
3. Guna mencapai prestasi yang gemilang, seseorang harus mempunyai sikap *Going Extra Mile* (memberikan sedikit lebih daripada yang lain) untuk menciptakan magnet bagi dirinya. Magnet ini akan menjadikannya "lebih" daripada orang lain.
4. Jangan mudah berputus asa, sebab setiap waktu memberikan kesempatan kapada setiap orang untuk berkembang dan meraih sukses. Ikutilah perbuatan buruk dengan perbuatan baik dengan membuat suatu karya yang dapat bermanfaat bagi diri kita maupun orang lain, daripada hanya

menyesal terus-menerus yang hanya akan menjadikan diri kita semakin terpuruk tanpa adanya perbaikan dan hasil yang nyata.

5. Disiplin dan komitmen terhadap cita-cita yang telah kita tetapkan akan mengantarkan kita pada kesuksesan kita dalam memimpin dan membangun suatu perusahaan.

Setiap langkah yang besar berawal dari langkah-langkah yang kecil, begitu pula setiap perusahaan yang besar berawal dari usaha kecil yang terus-menerus kita lakukan.

POWERFULL GOAL

5 SYARAT SEBUAH TUJUAN AGAR MENJADI

POWERFULL:

1. ***SPIRITUAL***

   - Alloh SWT meridhoi
   - Ibadah
     - Berbuat terbaik (FASTABIQUL KHOIROT)
     - Berbakti kepada orang tua

2. MASIF

- Jelas dan terukur
- Kuat

3. SENSITIF

- Dapat dirasakan dengan alat indra
- Emosi

4. PROGRESIF

- Bertahap mencapainya
- Meningkat

5. PROAKTIF

- Aktif mencari jalan mencapai tujuan
- Jangan tinggalkan semua amal meskipun tidak bisa dikerjakan semuanya.

Karena tadi sudah membahas tentang Powerfull Goals, lalu kriteria dari Smart Goals itu apa aja?

Melalui huruf S.M.A.R.T. kita bisa melihat apakah target/rencana yang kita buat itu sudah dikatakan baik/belum. Adapun kriteria dari SMART Goals sendiri antara lain :


SPESIFIC
Target yang baik adalah target yang spesifik, jelas, dan dapat dipaparkan secara detail.
MEASUREABLE
Target yang baik adalah target yang dapat diukur progress pencapaiannya.
ACHIEVABLE
Target yang baik adalah target yang dapat dicapai/dijangkau.
RELEVANT
Target yang relevant adalah target yang apabila tercapai maka terget tersebut memiliki dampak yang baik bagi semua hal.

TIMEBOUND

Target yang baik selalu dibatasi pencapaiannya dengan batas waktu tertentu, tujuannya agar progress yang baik bisa diusahakan

RENUNGAN TENTANG MASA DEPAN

Setiap orang memiliki kesempatan,

Tetapi tidak setiap orang menggunakan kesempatan

yang diberikan

Setiap orang diberi kehidupan,

Tetapi tidak semua orang benar-benar memanfaatkan

hidupnya

Setiap orang berhak memilih,

Tetapi tidak semua orang memilih

Setiap orang bisa berkarya,

Tetapi tidak semua orang memiliki karya

Yang kemarin masih dipaksa untuk ke asrama

Yang kemarin masuk asrama karena terpaksa

Yang kemarin masih bermalas-malasan sekolah

Atau alasan lain yang belum jelas tujuanmu

Maka saat ini aku memiliki tujuan

AYO MERANCANG PETA MASA DEPAN YANG MENCERAHKAN

SKALA PRIORITAS

CONTOH

NO	WAKTU/ USIA	TARGET	CARA MENCAPAI
1	1 minggu ke depan (21-28 Oktober)	Menguasai Materi persamaan linier	Setiap hari mengerjakan 2 soal persamaan linier
2	2 Bulan ke depan (April 2021)	Lulus SMA dengan nilai rata-rata minimal 80	Setiap minggu mengerjakan soal-soal Ujian
3	Juni 2021	Masuk Perguruan Tinggi Negeri	Belajar soal-soal SMNPTN setiap hari 10 soal, Les Online
4	Usia 20 Tahun	Menjadi Enterprenuer di Bidang IT	Mencari informasi mengenai bisnis IT, Mengikuti kursus Bisnis IT

INI SKALA PRIORITASKU

NO	WAKTU/ USIA	TARGET	CARA MENCAPAI
1			
2			
3			
4			
5			
6			

7			
8			
9			
10			
11			
12			

POST TEST: SETALAH MENGIKUTI KEGIATAN APA YANG KALIAN RASAKAN?

Jawablah pertanyaan berikut dengan memilih satu dari empat pilihan jawaban yang sesuai dengan keadaan Anda yang sebenarnya dengan memberikan tanda SILANG (X). Tidak ada jawaban benar atau salah dalam semua pertanyaan berikut, melainkan yang paling sesuai dengan diri Anda.

1. Saya kesulitan membuat rencana untuk membantu saya mencapai tujuan.

	1	2	3	4	
Tidak setuju					**Setuju**

2. Saya mengalami kesulitan menetapkan tujuan untuk diri saya sendiri.

	1	2	3	4	
Tidak setuju					**Setuju**

3. Begitu saya memiliki tujuan, saya biasanya dapat merencanakan bagaimana mencapainya.

	1	2	3	4	
Tidak setuju					**Setuju**

4. Saya cepat menyerah

	1	2	3	4	
Tidak setuju					**Setuju**

5. Saya menetapkan tujuan untuk diri saya sendiri dan memantau kemajuan saya.

	1	2	3	4	
Tidak setuju					**Setuju**

6. Ketika saya mencoba untuk mengubah sesuatu, saya memperhatikan bagaimana saya melakukannya.

	1	2	3	4	
Tidak setuju					**Setuju**

7. Saya tidak memperhatikan efek dari tindakan saya sampai semuanya terlambat.

	1	2	3	4	
Tidak setuju					**Setuju**

8. Saya memiliki standar pribadi, dan mencoba untuk menjalaninya.

	1	2	3	4	
Tidak setuju					**Setuju**

9. Saya mudah teralihkan dari rencana saya.

	1	2	3	4	
Tidak setuju					**Setuju**

10. Saya mengalami kesulitan untuk menindaklanjuti hal-hal setelah saya memutuskan untuk melakukan sesuatu.

	1	2	3	4	
Tidak setuju					**Setuju**

11.Saya mampu mencapai tujuan yang saya tetapkan untuk diri saya sendiri.

Tidak setuju	1	2	3	4	**Setuju**

12. Seringkali saya tidak memperhatikan apa yang saya lakukan.

Tidak setuju	1	2	3	4	**Setuju**

13. Saya menunda membuat keputusan.

Tidak setuju	1	2	3	4	**Setuju**

14.Jika saya membuat resolusi untuk mengubah sesuatu, saya memberi banyak perhatian pada apa yang saya lakukan.

Tidak setuju	1	2	3	4	**Setuju**

15. Saya sepertinya tidak belajar dari kesalahan saya.

Tidak setuju	1	2	3	4	**Setuju**

16. Saya biasanya melacak kemajuan saya menuju tujuan saya.

Tidak setuju	1	2	3	4	**Setuju**

17. Begitu saya melihat masalah atau tantangan, saya mulai mencari solusi yang bisa saya lakukan.

Tidak setuju	1	2	3	4	**Setuju**

18. Saya belajar dari kesalahan saya.

Tidak setuju	1	2	3	4	**Setuju**

19. Saya kesulitan mengambil keputusan tentang berbagai hal.

Tidak setuju	1	2	3	4	**Setuju**

20. Saya biasanya hanya membuat kesalahan satu kali untuk belajar darinya.

Tidak setuju	1	2	3	4	**Setuju**

21. Saya bisa berpegang pada rencana yang sudah saya tentukan dan dapat melakukannya dengan dengan baik.

Tidak setuju	1	2	3	4	**Setuju**

22. Saya biasanya dapat menemukan beberapa alternative pemecahan masalah ketika saya ingin mengubah sesuatu.

Tidak setuju	1	2	3	4	**Setuju**

MODUL 2

PELATIHAN KONSELOR SEBAYA ANAK PANTI ASUHAN

PRE TEST: ISI DENGAN JUJUR YA

Jawablah pertanyaan berikut dengan memilih satu dari empat pilihan jawaban yang sesuai dengan keadaan Anda yang sebenarnya dengan memberikan tanda SILANG (X). Tidak ada jawaban benar atau salah dalam semua pertanyaan berikut, melainkan yang paling sesuai dengan diri Anda.

1. Saya melamun dan berfantasi, secara teratur, tentang hal-hal yang mungkin terjadi pada saya.

	1	2	3	4	
Tidak setuju					**Setuju**

2. Saya sering memiliki perasaan yang lembut dan prihatin terhadap orang yang kurang beruntung daripada saya

	1	2	3	4	
Tidak setuju					**Setuju**

3. Saya terkadang merasa sulit untuk melihat sesuatu dari sudut pandang "orang lain".

	1	2	3	4	
Tidak setuju					**Setuju**

4. Terkadang saya tidak merasa kasihan pada orang lain saat mereka mengalami masalah

	1	2	3	4	
Tidak setuju					**Setuju**

5. Saya benar-benar terlibat dengan perasaan karakter ketika membaca novel.

	1	2	3	4	
Tidak setuju					**Setuju**

6. Dalam situasi darurat, saya merasa gelisah dan tidak nyaman.

	1	2	3	4	
Tidak setuju					**Setuju**

7. Saya biasanya objektif ketika saya menonton film atau drama, dan saya tidak terlalu sering terjebak di dalamnya.

1	2	3	4

Tidak setuju					**Setuju**

8. Saya mencoba untuk melihat dari sisi ketidaksepakatan setiap orang sebelum saya membuat keputusan.

	1	2	3	4	
Tidak setuju					**Setuju**

9. Ketika saya melihat seseorang dimanfaatkan, saya merasa agak protektif terhadap mereka.

	1	2	3	4	
Tidak setuju					**Setuju**

10. Saya terkadang merasa tidak berdaya ketika saya berada di tengah situasi yang sangat emosional.

	1	2	3	4	
Tidak setuju					**Setuju**

11. Kadang-kadang saya mencoba memahami teman-teman saya dengan lebih baik dengan membayangkan bagaimana segala sesuatunya terlihat dari sudut pandang mereka.

	1	2	3	4	
Tidak setuju					**Setuju**

12. Menjadi sangat terlibat dalam buku atau film bagus agak jarang bagi saya.

	1	2	3	4	
Tidak setuju					**Setuju**

13. Ketika saya melihat seseorang terluka, saya cenderung tetap tenang.

	1	2	3	4	
Tidak setuju					**Setuju**

14. Kemalangan orang lain biasanya tidak terlalu mengganggu saya.

1	2	3	4

Tidak setuju					**Setuju**

15. Jika saya yakin saya benar tentang sesuatu, saya tidak membuang banyak waktu mendengarkan argumen orang lain.

	1	2	3	4	
Tidak setuju					**Setuju**

16. Setelah menonton drama atau film, saya merasa seolah-olah saya adalah salah satu karakternya.

	1	2	3	4	
Tidak setuju					**Setuju**

17. Berada dalam situasi emosional yang tegang membuatku takut.

	1	2	3	4	
Tidak setuju					**Setuju**

18. Ketika saya melihat seseorang diperlakukan tidak adil, saya terkadang tidak merasa kasihan pada mereka.

Tidak setuju	1	2	3	4	**Setuju**

19. Saya biasanya cukup efektif dalam menangani keadaan darurat.

Tidak setuju	1	2	3	4	**Setuju**

20. Saya sering tersentuh oleh hal-hal yang saya lihat terjadi.

Tidak setuju	1	2	3	4	**Setuju**

21. Saya percaya bahwa ada dua sisi untuk setiap pertanyaan dan mencoba untuk melihat keduanya.

Tidak setuju	1	2	3	4	**Setuju**

22.Saya akan menggambarkan diri saya sebagai orang yang cukup berhati lembut.

Tidak setuju	1	2	3	4	**Setuju**

23.Ketika saya menonton film yang bagus, saya dapat dengan mudah menempatkan diri saya di posisi pemeran utama.

Tidak setuju	1	2	3	4	**Setuju**

24.Saya cenderung kehilangan kendali selama keadaan darurat.

Tidak setuju	1	2	3	4	**Setuju**

25. Ketika saya marah pada seseorang, saya biasanya mencoba untuk "menempatkan diri saya pada posisinya" untuk sementara waktu.

	1	2	3	4	
Tidak setuju					**Setuju**

26. Ketika saya membaca sebuah cerita atau novel yang menarik, saya membayangkan bagaimana perasaan saya jika peristiwa-peristiwa dalam cerita tersebut yang terjadi pada saya.

	1	2	3	4	
Tidak setuju					**Setuju**

27. Ketika saya melihat seseorang yang sangat membutuhkan bantuan dalam keadaan darurat, hati saya hancur berkeping-keping.

	1	2	3	4	
Tidak setuju					**Setuju**

28. Sebelum mengkritik seseorang, saya mencoba membayangkan bagaimana perasaan saya jika saya berada di tempat mereka.

	1	2	3	4	
Tidak setuju					**Setuju**

SESI 1

Mendengar Aktif:

Sebuah Seni Berkomunikasi

Waktu : 60 Menit

Metode : Psikoedukasi dan Game

Materi :

Mendengar adalah elemen dasar dari komunikasi yang efektif. Apabila dibandingkan dengan berbagai elemen komunikasi lainnya, menghubungkan antara komunikasi dengan mendengar aktif adalah sebuah korelasi yang tepat agar membentuk komunikasi dua arah yang sukses (Mălureanu & Vasluianu, 2016).

Komunikasi adalah proses memberi dan menerima pesan makna diantara dua orang (Clevenger, 1959). Dalam komunikasi sendiri terdiri dari 2 jenis yaitu komunikasi verbal dan non verbal. Komunikasi verbal adalah proses penyampaian pesan dengan menggunakan kata-kata, baik secara lisan maupun tulisan. Sedangkan komunikasi non verbal adalah proses penyampaian pesan melalui berbagai isyarat yang bukan kata-kata. Simbol non verbal ini biasanya

ditampilkan dengan volume saat mengucap kata, sentuhan, gesture atau gerakan tubuh, gerakan mata, dan juga ekspresi wajah.

Apa saja yang perlu diperhatikan dalam proses mendengar aktif berkaitan dengan komunikasi verbal dan non verbal? Berikut adalah penjelasannya

- Komunikasi Non Verbal

  Bahasa tubuh :

  - Gerakan dan posisi tubuh : rileks, condong kea rah klien sebaya yang diajak berbicara
  - Hindari : menunduk, terus gelisah, tangan membuat Gerakan tertentu, melihat ke jalan atau melihat ke arah *handphone* terus menerus.

  Ekspresi :

  - Wajah tersenyum, senang menunjukkan minat
  - Hindari : muram, kesal, marah, takut, kecewa, bingung

  Suara: volume cukup terdengar, bicara tidak cepat, nada bicara tenang

  Kontak Mata: menatap mata klien sebaya tanpa ketegangan

Sentuhan : menepuk bahu, memegang tangan.

- Komunikasi Verbal
  - Menjaga alur pembicaraan dengan cara parafrasing (mengulang kembali pembicaraan klien sebaya) bertujuan untuk memperjelas atau mempertegas pernyataan klien sebaya dan membantunya untuk dapat memfokuskan pembicaraan dan merefleksikan pembicaraan
  - Menafsirkan dengan benar
  - Tidak memotong pembicaraan
  - Klarifikasi dengan cara mengajukan pertanyaan terbuka dan tertutup sesuai kebutuhan
  - Menyimpulkan

Hal yang perlu dihindari:

- Menghakimi :
  - Mengkritik “apa kamu belum mengerti juga”
  - Memberi julukan “kamu tolol”
  - Mengasumsi “kamu tidak suka sikap saya ya..”
  - Menyindir “kamu hebat ya, suka bolos tapi ga dikeluarkan juga dari sekolah”

- Memberikan solusi
  - Memerintahkan “aku akan lebih suka jika kamu..”
  - Mengancam “kalau kamu tidak lakukan, maka..”
  - Moralisasi “kamu seharusnya..”
  - Pertanyaan berlebihan “kamu pergi kemana, sama siapa, apa yang kamu lakukan?”
  - Menasehati “kamu akan lebih baik jika..”
- Menghindar
  - Membelokkan “olahraga apa yang kamu lakukan sekarang?”
  - Argument logis “satu-satunya cara supaya kamu tidak tertular HIV adalah tidak “pakai” lagi”
  - Menentramkan “ya sudah, badai pasti berlalu”

Proses mendengar aktif bukan hanya sekedar mendengarkan saja apa yang disampaikan oleh penyampai pesan, namun juga harus menyimak dan mampu menyampaikan kembali apa yang sudah didengar sebelumnya. Pada konteks konseling teman sebaya, terdapat banyak manfaat dari melakukan proses mendengar aktif, yaitu 1). Dapat memeriksa

kembali apakah pemahaman kita terhadap pesan yang disampaikan oleh klien sudah benar atau belum, 2). Klien akan merasa dihargai apabila didengarkan, 3). Mencegah timbulnya rasa marah dari klien, 4). Membantu untuk mengingat apa yang dikatakan oleh klien, 5). Membantu untuk menjaga hubungan baik dengan klien karena klien merasa tidak digurui.

Ada beberapa teknik yang perlu dipelajari agar dapat mendengar aktif dan dapat menjadi bekal menjadi seorang konselor sebaya.

1. Parafrasing (Refleksi Isi)
   Menyampaikan hasil pemahaman diri sendiri mengenai esensi dari pesan klien dengan menggunakan kata-kata sendiri (kata-kata yang berbeda dari apa yang diucapkan klien)
   Tips parafrasing:
   - Ringkas
   - Merefleksikan inti pesan yang disampaikan klien namun menggunakan kata-kata sendiri, bukan mengulang ucapan klien)
   - Fokus pada isi pesan
2. Refleksi Perasaan
   Mampu menangkap dan mengungkapkan kembali isi perasaan yang dirasakan oleh klien dalam bentuk

kata-kata, dan kemudian diucapkan oleh konselor sebaya

Tips refleksi perasaan:

- Fokus pada kata-kata yang menggambarkan perasaan
- Amati bahasa tubuh saat klien sedang menyampaikan pesan
- Tanyakan pada dirimu sendiri, “bila saya mengalami hal tersebut, apa yang saya rasakan?”

3. Refleksi Makna

   Merupakan gabungan dari refleksi isi dan refleksi perasaan.

   Tips refleksi makna:

   - Menangkap fakta dan perasaan klien dengan mengungkapkan kembali dalam kata-kata yang dibuat sendiri oleh konselor
   - Menangkap isi pesan, baik yang berupa fakta maupun perasaan yang diungkapkan oleh klien

Komunikasi verbal dan non verbal juga menjadi

BE ACTIVE LISTENER!

Tujuan:

Peserta akan mampu untuk:

- Mengidentifikasi setiap kalimat yang disampaikan oleh klien dengan menggunakan tiga teknik mendengar aktif tersebut diatas
- Memberikan pemahaman yang tepat dari pesan yang disampaikan oleh klien

Game Simulasi : Be Active Listener!

Prosedur Pelaksanaan

1. Setiap peserta akan melakukan aktivitas ini secara berpasangan
2. Setiap pasangan akan mendapatkan satu (1) lembar kerja yang berisikan beberapa penggalan pernyataan dari klien
3. Setiap pasangan harus melakukan proses identifikasi dari pernyataan klien di dalam lembar kerja menggunakan tiga (3) teknik mendengar aktif yang telah disampaikan pada bagian penjelasan diatas
4. Proses diskusi akan dilakukan selama 15 menit
5. Setelah 15 menit, masing-masing pasangan secara bergiliran akan mempresentasikan hasil diskusi mereka di depan kelas

6. Trainer akan mengadakan forum diskusi terbuka untuk masing-masing pasangan dan dapat saling memberikan koreksi maupun masukan

Lembar Kerja 1: Be Active Listener!

Buatlah respon berupa kalimat tertulis terhadap pernyataan-pernyataan dibawah ini, yang dapat dikategorikan ke dalam mendengar aktif!

1. Aku ini orangnya gak enakan.. Kalo ada teman yang mengajak bolos, ya aku bilang ayo aja. Padahal aku tau kalo nilai-nilai pelajaranku pas-pas an. Aku juga gak terlalu pandai. Jadinya sering dimarahin guru karena banyak bolos. Pernah orangtuaku dipanggil ke sekolah, tapi kebiasanaku mau diajak bolos ya tetap kujalani.

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……

Refleksi perasaan = ……
……
……
……

Refleksi makna = ……
……
……

2. Sudah 3 bulan ini aku sering banget nongkrong sama geng ku. Aku tahu sebenarnya aktivitasku itu gak ada gunanya, tapi aku merasa jenuh berada di panti terus..

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……

Refleksi makna = ……
……
……
……
……

3. Baru minggu lalu, aku kenal cowok lewat facebook, eh.. tapi dia sudah ngajak aku jalan ke pacet.

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……

……
……
……
……
……

Refleksi makna = ……

……
……
……
……
……
……

4. Aku tahu pelajaran kelas 3 SMP itu sulit, dan aku mesti banyak belajar. Tapi aku juga tidak mau belajar terus di panti. Aku masih butuh teman-teman.

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……
……

5. Orangtuaku minta aku melanjutkan ke SMA, sedangkan aku kepengen melanjutkan sekolah di SMK Perhotelan. Eh ternyata orangtuaku ga setuju karena katanya nanti aku jadi ikutan pergaulan bebas

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……

6. Pusing…pusing deh… sudah dua bulan ini aku batuk-batuk, tapi masih belum sembuh. Mau berobat, tapi takut sama orangtua karena pasti disalahkan karena aku merokok

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……
……

7. Ibuku selalu bilang agar rangkingku di kelas masuk 10 besar. Tapi ibuku gak pernah pahamiku. Ibu sering pergi dan tidak mempedulikanku

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……
……

8. Kalau aku menstruasi, kupikir kok makin sakit ya. Bahkan seminggu sebelum mens, kepala pusing, pengen marah-marah, bahkan mual. Kalau sudah begitu, aku sulit konsentrasi nih.

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……

9. Aku gak nyangka, ternyata salah satu temanku kecanduan narkoba lho. Padahal keliatannya dia baik dan alim. Aku juga suka curhat sama dia.

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……
……
……

10. Mens bulan ini kok belum datang juga ya… biasanya tengah bulan udah mens. Perutku ga nyaman banget nih, kayak kembung gitu. Aduh, aku takut hamil..

Kalimat mendengar aktif:

Parafrasing = ……

……
……
……
……
……

Refleksi perasaan = ……
……
……
……
……
……

Refleksi makna = ……
……
……
……
……
……

MARI BERLATIH MENDENGAR AKTIF DALAM KELOMPOK

Tujuan Latihan:

Memberikan kesempatan praktik **mendengar aktif** dengan menggunakan masalah keseharian masing-masing peserta

Durasi : 30 menit

Prosedur pelaksanaan :

1. Panitia membagi peserta ke dalam kelompok yang terdiri dari 3 orang
2. Masing-masing orang dalam kelompok bergiliran berperan sebagai konselor sebaya, klien sebaya, dan pengamat
3. Tugas konselor sebaya: melakukan **mendengar aktif** terhadap apa yang diungkapkan oleh klien sebaya.
4. Tugas klien sebaya: mengungkapkan masalahnya sendiri sehari-hari dalam beberapa kalimat kepada konselor sebaya
5. Tugas pengamat: mengamati dan mengevaluasi apakah konselor sebaya sudah mendengar aktif menggunakan lembar pencatatan observasi.

Tabel 1. Lembar Pencatatan Observasi

Komponen Keterampilan Konselor Sebaya		Ya	Tidak	Komentar
Komunikasi Non Verbal				
Posisi duduk	Rileks			
	Condong ke arah klien			
Ekspresi	Senyum			
	Antusias			
Suara	Volume cukup terdengar			
	Bicara tidak cepat			
	Nada bicara tenang			
Kontak mata	Menatap klien tanpa ketegangan			
Sentuhan	Menepuk bahu/ memegang tangan			
Komunikasi Verbal				
Tidak menghakimi	Tidak mengkritik			
	Tidak memberi julukan			
	Tidak menyindir			
	Tidak berasumsi			

Tidak memberikan solusi	Tidak memerintah			
	Tidak mengancam			
	Tidak moralisasi			
	Tidak bertanya berlebihan			
	Tidak menasehati			
Tidak menghindari masalah	Tidak membelokkan masalah			
	Tidak menghibur			
	Tidak berargumen			
Mendengar Aktif				
	Refleksi Isi			
	Refleksi Perasaan			
	Refleksi Makna			

SESI 2

EMPATI: EKSPRESIKAN PERASAANMU

Waktu : 45 Menit

Metode : Psikoedukasi

Materi :

Empati adalah kondisi emosi dimana seseorang merasakan apa yang dirasakan orang lain seperti dia mengalaminya sendiri, dan apa yang dirasakannya tersebut sesuai dengan perasaan dan kondisi orang yang bersangkutan. Meskipun empati merupakan respon yang bersifat emosi namun juga melibatkan keterampilan kognitif seperti kemampuan untuk mengenali kondisi emosi orang lain dan kemampuan mengambil peran (Feshbach dalam Eisenberg, 1989).

Ciri-ciri empati adalah melibatkan aspek afektif dan kognitif. Aspek afektif yang dimaksud adalah kecenderungan seseorang untuk mengalami perasaan emosional orang lain yaitu ikut merasakan ketika orang lain merasa sedih, bahagia, terluka, menderita, bahkan disakiti. Sedangkan aspek kognitif dalam empati difokuskan pada proses intelektual untuk memahami perspektif orang lain dengan tepat dan menerima

pandangan mereka. Misalnya membayangkan perasaan orang lain ketika marah, kecewa, senang, memahami keadaan orang lain dari cara pandang, cara berbicara, dan raut wajah saat berbicara (Eisenberg, 2002).

Sebagai seorang konselor sebaya, peserta perlu memahami apa yang dirasakan oleh klien tetapi masih dapat memisahkan perasaan itu sendiri. Empati dapat ditunjukkan melalui sikap memahami perasaan klien dan bisa membayangkan seandainya diri kita yang memiliki masalah serupa dengan masalah klien tersebut.

Ada beberapa cara untuk menumbuhkan rasa empati di dalam diri yaitu

1. Belajar mendengarkan
   Melalui proses mendengarkan aktif, peserta akan dapat memahami permasalahan yang dialami oleh klien dengan lebih obyektif
2. Bersikap membuka diri
   bersikap menerima dan terbuka terhadap apapun permasalahan yang disampaikan oleh klien, akan membuat klien merasa lebih nyaman bercerita.
3. Berikan afeksi secara fisik
   Sentuhan fisik dapat mencerminkan rasa empati atas apa yang disampaikan klien. Namun pada

proses pelaksanaan konseling, perlu juga mempertimbangkan factor budaya apabila akan melakukan sentuhan fisik. Akan lebih baik sentuhan dilakukan pada konselor dan klien yang memiliki jenis kelamin yang sama, agar tidak menimbulkan perbedaan persepsi.

4. Fokuskan perhatian di lingkungan sekitar
   Lebih peka terhadap apa yang terjadi di lingkungan sekitar juga dapat mengasah rasa empati.
5. Jangan menjustifikasi
   Saat klien menceritakan mengenai masalahnya, dengarkan saja. Jangan memberikan komentar terlebih dahulu sebelum klien menyelesaikan ceritanya. Terlebih lagi jangan memberikan simpulan yang tidak disetujui oleh klien.
6. Berikan bantuan
   Sesudah memahami permasalahan secara keseluruhan, berilah bantuan sesuai dengan apa yang peserta. Apabila permasalahan dirasa terlalu sulit untuk ditangani, maka peserta harus merujuk klien kepada konselor professional.

EKSPRESIKAN PERASAANMU

Gambar 1 : Anak Broken Home

Apa yang kamu rasakan apabila menjadi anak yang berada dalam posisi di dalam gambar ini?

Gambar 2 : Ibu yang sedang membawa foto anaknya yang meninggal

Apa yang kamu rasakan apabila menjadi ibu yang berada dalam posisi di dalam gambar ini?

Gambar 3 : Anak yang menjadi korban bullying

Apa yang kamu rasakan apabila menjadi anak yang berada dalam posisi di dalam gambar ini?

Gambar 4: Anak-anak yang tertangkap karena minuma keras

Apa yang kamu rasakan apabila menjadi anak yang berada dalam posisi di dalam gambar ini?

gambar 5: Persahabatan remaja

Apa yang kamu rasakan apabila menjadi anak yang berada dalam posisi di dalam gambar ini?

SESI 3

Remaja dan Permasalahan Sosial di sekelilingnya

Durasi : 45 menit

Metode : Psikoedukasi

Materi :

Remaja adalah salah satu tahapan perkembangan yang penting dalam rentang kehidupan seseorang. Masa ini adalah periode transisi dari masa kanak-kanak menuju masa dewasa. Oleh karena itu pada masa ini adalah tonggak awal menuju pada kematangan emosi adalah pada masa remaja. Hurlock (2000) menjelaskan masa remaja merupakan masa yang sarat akan konflik, karena pada masa ini setiap individu akan mengalami perubahan yang sangat kompleks, yaitu perubahan fisik, pola perilaku, peran sosial, serta merupakan masa pencarian identitas untuk menjadi diri sendiri sebagai individu.

Perkembangan remaja dapat berkembang ke arah positif ataupun negatif. Menurut Hurlock (2000) salah satu permasalahan yang dialami remaja adalah masalah sosial. Untuk mencapai suatu pola sosialisasi dewasa, remaja harus membuat berbagai penyesuaian baru. Hal terpenting dan tersulit adalah penyesuaian diri

dengan meningkatnya pengaruh kelompok sebaya, perubahan dalam perilaku sosial, dalam seleksi persahabatan, nilai-nilai baru dalam dukungan dan penolakan sosial, dan nilai-nilai baru dalam seleksi pemimpin.

Banyaknya perubahan yang terjadi di dalam diri remaja, merupakan potensi terjadinya permasalahan baik di dalam diri remaja ataupun dengan lingkungan sekitar. Dalam sesi ini akan dibahas beberapa permasalahan yang kerap kali terjadi dan dialami oleh para remaja.

1. Body Image

   Topik ini berkaitan dengan perkembangan fisik remaja yang banyak terjadi perubahan sangat drastis. Dari mulai pola menstruasi, tumbuhnya payudara, perubahan bentuk tubuh, perubahan suara yang banyak terjadi pada remaja perempuan. Sedangkan pada remaja laki-laki biasanya ditandai dengan pecahnya suara, tumbuhnya kumis dan jambang, tumbuh jakun, muncul banyaknya jerawat di area wajah, serta tubuh yang semakin tinggi.

   Perubahan drastis ini menyebabkan banyak remaja yang merasa stres bahkan ada pula yang kehilangan kepercayaan diri. Sehingga akhirnya merasa sedih dan cemas yang berlebihan. Hal ini

akan semakin menjadi apabila lingkungan di sekitarnya merespon perubahan fisik ini secara negatif, misalnya dengan mengolok atau menghina. Maka ini akan menimbulkan potensi untuk melakukan kenakalan.

2. Stabilitas Emosi

Perubahan fisik yang drastis disertai pula dengan proses berkembangnya tingkat kematangan emosi, secara nyata berpengaruh pada kondisi stabilitas emosi remaja. Kebutuhan akan penerimaan di dalam lingkungan teman sebaya juga dapat menjadi salah satu pemicu kestabilan emosi pada remaja. Apabila remaja mengalami penolakan dari lingkungan sebayanya, maka bisa dipastikan remaja akan mudah mengalami stres dan bahkan dapat mengembangkan kepribadian yang rendah diri.

Tuntutan dan harapan dari orang dewasa yang ada di sekitarnya mengenai konsep kemandirian, menurut Elkind & Postman (dalam Fuhrmann, 1990) juga ikut mengambil peran menimbulkan banjir stres pada remaja.

3. Depresi Akademik

Harapan setiap orangtua adalah memiliki anak yang berprestasi dalam bidang akademiknya.

Harapan ini menuntun orangtua untuk memberikan tuntutan yang tinggi pada anak-anaknya. Namun pada kenyataannya, tidak semua anak dapat menampilkan prestasi akademik yang baik.

Kesenjangan antara harapan dan kenyataan ini juga bisa menjadi salah satu penyebab dari munculnya stres akademik yang apabila terjadi secara terus menerus dan dalam waktu yang lama akan menjadi depresi akademik. Sehingga akhirnya menimbulkan rasa benci sekolah pada sang anak.

4. Penyalahgunaan obat-obatan

Ketidakharmonisan di dalam keluarga, kesibukan orangtua yang sangat tinggi sehingga anak merasa tidak diperhatikan, atau bahkan orangtua yang bercerai juga bisa menjadi salah satu penyebab remaja melakukan perilaku penyalahgunaan obat-obatan. Hal ini dilakukan sebagai bentuk pelarian dari kondisi lingkungannya yang tidak dapat mendukung proses tumbuh kembang remaja.

Bahkan tekanan sosial yang terjadi di sekolah juga bisa menjadi sebab remaja melakukan hal ini. Apabila remaja sudah mengalami kecanduan, maka proses penyembuhannya haruslah langsung ditangani oleh pihak professional yang biasanya dilakukan di panti rehabilitasi.

5. Kenakalan Remaja

   Kenakalan remaja ini banyak sekali bentuknya. Masuk dalam geng motor, ataupun geng yang bersifat destruktif adalah salah satu jenisnya. Tawuran atau perkelahian secara massal ataupun secara individu adalah bentuk lainnya. Pencurian, perkosaan, penodongan, ataupun perampokan bahkan vandalism juga merupakan bentuk dari kenakalan remaja.

   Alasan remaja melakukan aktivitas kenalakan ini biasanya karena faktor solidaritas antar kelompok. Bahkan hal ini juga menjadi syarat agar seorang remaja dapat diterima menjadi salah satu kelompok remaja.

6. Kehamilan di Luar Nikah

   Hal ini banyak sekali faktor yang dapat menyebabkannya terjadi. Bisa berasal dari faktor hubungan antar anggota keluarga yang kurang harmonis, ataupun juga karena faktor pondasi spiritualnya masih belum kuat. Bahkan juga ada karena faktor ketakutan apabila diputuskan oleh pasangan dan kemudian menjadi kurang populer di kalangan teman sebaya, karena dianggap “jomblo”.

SESI 4

DASAR-DASAR TEKNIK KONSELING SEBAYA

Durasi : 120 menit

Metode : 60 menit Penjelasan, 60 menit Roleplay

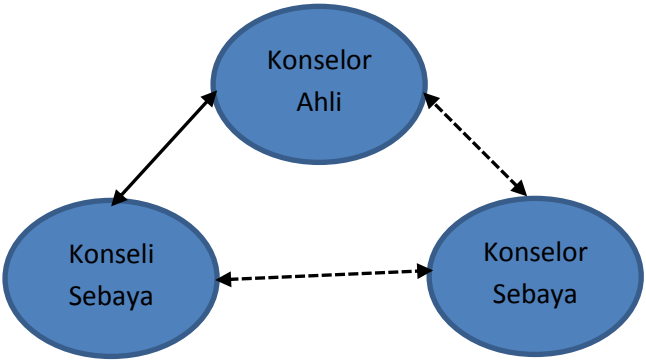
Materi :

Konseling teman sebaya awalnya muncul dengan konsep *peer support* yang dimulai tahun 1939 untuk membantu para penderita alkoholik (Carter, 2005 dalam Suwarjo, 2008). Dalam konsep tersebut diyakini bahwa individu yang pernah kecanduan alkohol dan memiliki pengalaman berhasil mengatasi kecanduan tersebut akan lebih efektif dalam membantu individu lain yang sedang mencoba mengatasi kecanduan alohol. Dan dalam beberapa tahun, konselor sebaya mulai merambah ke berbagai setting kehidupan.

Konseling sebaya merupakan suatu bentuk pendidikan psikologis yang disengaja dan sistematik. Konseling sebaya memungkinkan siswa untuk memiliki keterampilan-keterampilan guna mengimplementasikan pengalaman kemandirian dan kemampuan mengontrol diri yang sangat bermakna bagi remaja. Secara khusus konseling teman sebaya tidak memfokuskan pada evaluasi isi, namun lebih memfokuskan pada proses

berfikir, proses-proses perasaan dan proses pengambilan keputusan. Dengan cara yang demikian, konseling sebaya memberikan kontribusi pada dimilikinya pengalaman yang kuat yang dibutuhkan oleh para remaja yaitu *respect*. (Carr, 1981 dalam Suwarjo, 2008).

Konseling teman sebaya secara kuat menempatkan keterampilan-keterampilan komunikasi untuk memfasilitasi eksplorasi diri dan pembuatan keputusan. Orang yang melakukan proses konseling sebaya disebut sebagai konselor sebaya. Konselor sebaya bukan lah seorang konselor professional. Konselor sebaya adalah para siswa (remaja) yang memberikan bantuan kepada siswa lainnya di bawah konselor ahli, sehingga terdapat hubungan triardik seperti di bawah ini.



= interaksi antara konselor ahli dengan konseli melalui konselor sebaya

= interaksi langsung antara konselor ahli dengan konselor sebaya

Gambar 1: Interaksi triardik antara konselor ahli, konselor sebaya dan konseli sebaya (Suwarjo, 2008)

Tidak semua masalah yang dapat diselesaikan oleh konselor sebaya. Hanya masalah yang dimasukkan dalam kategori ringan dan hanya berdampak pada diri klien sendiri yang masih bisa dibantu oleh konselor sebaya. Proses pemberian bantuan ini juga perlu melalui supervisi dari konselor ahli. Berikut adalah bagan proses identifikasi masalah dan pemberian bantuan kepada klien sebaya.

Kemampuan dasar yang harus dimiliki oleh konselor sebaya menurut pengembangan teori dari Erford (2015) adalah

1. Kemampuan memperhatikan (*attending response)*
   Kemampuan untuk memperhatikan klien sehingga konselor sebaya dapat menunjukkan sikap, tingkah laku, serta ekspresi wajah yang menerima secara tulus dan ikhlas dalam mengamalkan prinsip konseling.
2. Kemampuan berempati (*emphatizing)*
   Berempati terhadap permasalahan yang dihadapi oleh klien
3. Kemampuan menangkap serta merangkum (*summarizing)*
   Menggunakan kemampuan mendengar aktif dapat membantu konselor sebaya untuk bisa menangkap makna pesan klien secara tepat. Kemudian bisa merangkum keseluruhan isi pesan dari mulai awal sampai akhir menggunakan teknik parafrasing.
4. Kemampuan mengajukan pertanyaan (*questioning)*
   Merupakan kemampuan konselor untuk bisa memberikan pertanyaan yang tepat dalam proses mengungkap permasalahan utama yang sedang dialami klien
5. Keaslian (*genuineness*).

Ketulusan serta keikhlasan yang ditampilkan dalam perilaku jujur dan sesuai dengan pikiran

6. Keterampilan asertif (*assertiveness*)
   Yaitu kemampuan untuk untuk bisa mengungkapkan perasaan, pendapat, serta keyakinan terhadap kesulitan yang dialami dalam menjalani kehidupan sehari-hari
7. Pemecahan masalah (*problem solving*)
   Tahap ini hanya dilakukan untuk permasalahan yang ringan saja. Bahkan untuk masalah yang ringan dan berat, proses pemecahan masalah yang harus dituliskan konselor sebaya di dalam lembar pencatatan konseling adalah merujuk klien kepada konselor ahli.

Proses konseling haruslah merupakan sebuah proses yang sistematis dan terstruktur. Sehingga sejak awal sampai dengan akhir konseling, konselor haruslah menuliskan hasil pencatatan proses konseling di dalam lembar pencatatan konseling yang telah disiapkan. Hal ini bertujuan agar mempermudah semua pihak yang berkepentingan dengan klien untuk melihat kembali riwayat proses konseling yang telah dialami oleh klien. Sehingga apabila terjadi permasalahan di kemudian hari terhadap klien, semua pihak yang berkaitan dengan klien dapat memikirkan alternatif pemecahan masalah agar klien dapat menyelesaikan permasalahannya

LEMBAR PENCATATAN HASIL KONSELING

Tanggal	Nama	L/P	Umur	Masalah	Tindak Lanjut	
					Konseling	Rujuk

LEMBAR RUJUKAN KONSELING

Tanggal Konseling	……/……/20..	Konseling ke-	1/2/3/4/5/6
Nama Lengkap			
Tempat/ Tanggal Lahir			
Alamat			
Jenis Kelamin: L/P	No HP:	Suku:	Agama:
Klasifikasi kasus			
Keluhan			
Masalah utama			
Alasan Merujuk			

Pengirim:

Nama Konselor Sebaya : ……

Nomor HP : ……

MARI KITA BERMAIN PERAN

Waktu : 60 menit

Metode : Simulasi Kasus

Tujuan

Peserta mampu untuk:

- Mengembangkan tujuh kemampuan yang harus dimiliki oleh seorang konselor sebaya
- Melakukan proses konseling sebaya menggunakan kasus simulasi dan mempraktikkannya dengan teman sebayanya pula
- Menuliskan catatan konseling pada lembar pencatatan konseling berdasarkan hasil konseling dengan klien

Prosedur pelaksanaan:

1. Panitia akan menyiapkan 6 kartu skenario kasus yang akan digunakan peserta untuk roleplay
2. Panitia akan membagi peserta ke dalam kelompok yang berisi 2 orang (berpasangan)
3. Setiap pasangan akan diberikan 1 kartu scenario kasus, namun sebelumnya kelompok harus memutuskan siapa yang akan menjadi konselor sebaya dan siapa yang akan menjadi klien sebaya

4. Kartu skenario akan diberikan kepada peserta yang memilih memerankan klien sebaya
5. Dan pada peserta yang memilih memerankan sebagai konselor sebaya, akan diberikan lembar pencatatan konseling yang digunakan untuk proses mencatat seluruh proses simulasi konseling saat itu
6. Satu putaran simulasi akan dilakukan dalam durasi 20 menit. Dan terdapat 2 kali putaran dengan saling bertukar peran dan bertukar kartu skenario.

KARTU 1

Kamu adalah seorang siswa SMA di salah satu sekolah di Sidoarjo, dan berusia 17 tahun. Kamu punya satu gank yang beranggotakan 5 siswa yang cukup terkenal di sekolah. Gank tersebut terkenal dengan kebiasaan membolos dan merokok. Sebenarnya kamu sudah lama punya keinginan untuk keluar dari kelompok tersebut. Namun kamu khawatir kalau keluar dari kelompok maka akan dikucilkan dari kelompok. Akhirnya kamu pun curhat ke konselor sebaya.

KARTU 2

Kamu saat ini berusia 16 tahun dan curhat kepada konselor sebaya tentang masalah ibumu. “Saya tahu ibu saya selalu sakit, tetapi sebenarnya dia tidak sakit beneran. Sakit ini sakit itu. Selalu begitu dari dulu. Kalau saya mau pergi dengan teman-teman saya untuk waktu yang cukup lama, misalnya karena sedang libur, pasti ada keluhannya. Kalau saya punya teman yang dia tidak suka, pasti nanti dia bilang pusing. Saya tahu dia begitu karena usianya memang sudah 65 tahun. Bagaimana kalau sekarang dia sakit beneran…??

KARTU 3

Kamu saat ini berusia 17 tahun. Bercerita kepada temanmu yang seorang konselor sebaya. “Suatu hari, saya dan dua orang teman pakai putaw bareng-bareng. Saat itu, salah satu teman saya *overdosis* (OD). Saya dan teman saya yang satu lagi tidak bisa berbuat apa-apa hingga teman saya yang OD itu meninggal di hadapan saya. Melihat proses yang mengerikan itu langsung di depan mata saya sendiri membuat saya ketakutan dan ingin berhenti, tapi saya ragu apa saya bisa melakukannya…?

KARTU 4

Kamu sedang bercerita kepada konselor sebaya, bahwa kamu kalau di rumah dianggap sebagai anak yang “bandel” oleh orangtua dan saudara-saudaramu. Di rumah kamu juga merasa terisolasi dan tidak ada orang yang bisa diajak berbicara mengenai masalahmu. Hal ini terjadi pula saat di sekolah. Orangtuamu sangat sibuk dan seolah-olah tidak memiliki waktu untuk berbincang denganmu. Di sekolah sebenarnya kamu punya cukup banyak teman, namun banyak juga dari mereka yang kerjaannya hanya belajar saja, dan tidak mau diajak menjadi “anak gaul”. Kamu sangat kesal dengan keadaan anak-anak yang bersikap “banci” seperti itu sehingga akhirnya kamu sering bolos. Dan akhir-akhir ini kamu sering terlibat perkelahian antar pelajar. Hal ini sudah diketahui oleh wali kelas, dan kamu diancam dikeluarkan dari sekolah jika tidak mengubah kebiasaan.

KARTU 5

Kamu adalah seorang perempuan berusia 15 tahun. Saat berhadapan dengan konselor sebaya, kamu awalnya diam saja dan tidak mau berbicara. Kamu memiliki badan yang gemuk dengan tinggi 157 cm dan berat 65 kg. Kamu sangat ingin langsing seperti teman-temanmu yang lainnya. Sudah 1 minggu ini kamu tidak makan nasi dan sering memuntahkan kembali makanan yang masuk dengan cara memasukkan dua jarinya ke dalam tenggorokannya melalui mulut. Akhirnya kamu sering sakit. Kemudian konselor sebaya membujukmu untuk menceritakan masalahmu.

POST TEST: ISI DENGAN JUJUR YA

Jawablah pertanyaan berikut dengan memilih satu dari empat pilihan jawaban yang sesuai dengan keadaan Anda yang sebenarnya dengan memberikan tanda SILANG (X). Tidak ada jawaban benar atau salah dalam semua pertanyaan berikut, melainkan yang paling sesuai dengan diri Anda.

1. Saya melamun dan berfantasi, secara teratur, tentang hal-hal yang mungkin terjadi pada saya.

	1	2	3	4	
Tidak setuju					**Setuju**

2. Saya sering memiliki perasaan yang lembut dan prihatin terhadap orang yang kurang beruntung daripada saya

	1	2	3	4	
Tidak setuju					**Setuju**

3. Saya terkadang merasa sulit untuk melihat sesuatu dari sudut pandang "orang lain".

	1	2	3	4	
Tidak setuju					**Setuju**

4. Terkadang saya tidak merasa kasihan pada orang lain saat mereka mengalami masalah

	1	2	3	4	
Tidak setuju					**Setuju**

5. Saya benar-benar terlibat dengan perasaan karakter ketika membaca novel.

	1	2	3	4	
Tidak setuju					**Setuju**

6. Dalam situasi darurat, saya merasa gelisah dan tidak nyaman.

	1	2	3	4	
Tidak setuju					**Setuju**

7. Saya biasanya objektif ketika saya menonton film atau drama, dan saya tidak terlalu sering terjebak di dalamnya.

Tidak setuju	1	2	3	4	**Setuju**

8. Saya mencoba untuk melihat dari sisi ketidaksepakatan setiap orang sebelum saya membuat keputusan.

Tidak setuju	1	2	3	4	**Setuju**

9. Ketika saya melihat seseorang dimanfaatkan, saya merasa agak protektif terhadap mereka.

Tidak setuju	1	2	3	4	**Setuju**

10.Saya terkadang merasa tidak berdaya ketika saya berada di tengah situasi yang sangat emosional.

	1	2	3	4	
Tidak setuju					**Setuju**

11. Kadang-kadang saya mencoba memahami teman-teman saya dengan lebih baik dengan membayangkan bagaimana segala sesuatunya terlihat dari sudut pandang mereka.

	1	2	3	4	
Tidak setuju					**Setuju**

12. Menjadi sangat terlibat dalam buku atau film bagus agak jarang bagi saya.

	1	2	3	4	
Tidak setuju					**Setuju**

13. Ketika saya melihat seseorang terluka, saya cenderung tetap tenang.

	1	2	3	4	
Tidak setuju					**Setuju**

14. Kemalangan orang lain biasanya tidak terlalu mengganggu saya.

	1	2	3	4	
Tidak setuju					**Setuju**

15. Jika saya yakin saya benar tentang sesuatu, saya tidak membuang banyak waktu mendengarkan argumen orang lain.

	1	2	3	4	
Tidak setuju					**Setuju**

16. Setelah menonton drama atau film, saya merasa seolah-olah saya adalah salah satu karakternya.

Tidak setuju	1	2	3	4	**Setuju**

17. Berada dalam situasi emosional yang tegang membuatku takut.

Tidak setuju	1	2	3	4	**Setuju**

18. Ketika saya melihat seseorang diperlakukan tidak adil, saya terkadang tidak merasa kasihan pada mereka.

Tidak setuju	1	2	3	4	**Setuju**

19. Saya biasanya cukup efektif dalam menangani keadaan darurat.

Tidak setuju	1	2	3	4	**Setuju**

20. Saya sering tersentuh oleh hal-hal yang saya lihat terjadi.

	1	2	3	4	
Tidak setuju					**Setuju**

21. Saya percaya bahwa ada dua sisi untuk setiap pertanyaan dan mencoba untuk melihat keduanya.

	1	2	3	4	
Tidak setuju					**Setuju**

22. Saya akan menggambarkan diri saya sebagai orang yang cukup berhati lembut.

	1	2	3	4	
Tidak setuju					**Setuju**

23. Ketika saya menonton film yang bagus, saya dapat dengan mudah menempatkan diri saya di posisi pemeran utama.

	1	2	3	4	
Tidak setuju					**Setuju**

24. Saya cenderung kehilangan kendali selama keadaan darurat.

	1	2	3	4	
Tidak setuju					**Setuju**

25. Ketika saya marah pada seseorang, saya biasanya mencoba untuk "menempatkan diri saya pada posisinya" untuk sementara waktu.

	1	2	3	4	
Tidak setuju					**Setuju**

26. Ketika saya membaca sebuah cerita atau novel yang menarik, saya membayangkan bagaimana perasaan saya jika peristiwa-peristiwa dalam cerita tersebut yang terjadi pada saya.

	1	2	3	4	
Tidak setuju					**Setuju**

27. Ketika saya melihat seseorang yang sangat membutuhkan bantuan dalam keadaan darurat, hati saya hancur berkeping-keping.

	1	2	3	4	
Tidak setuju					**Setuju**

28. Sebelum mengkritik seseorang, saya mencoba membayangkan bagaimana perasaan saya jika saya berada di tempat mereka.

	1	2	3	4	
Tidak setuju					**Setuju**

Referensi

Chen, Y. H., & Lin, Y. J. (2018). Validation of the Short Self-Regulation Questionnaire for Taiwanese College Students (TSSRQ). *Frontiers in psychology, 9*, 259.

Damajanti, M., Juwitasari, M., Unik Pratiwi, A., Fatimah, E., & Sudaryanti. (2010). *Pedoman teknik konseling kesehatan remaja bagi konselor sebaya*. 1–57. http://kesga.kemkes.go.id/images/pedoman/TEKNIK KONSELING BAGI KONSELOR SEBAYA.pdf

De Corte, K., Buysse, A., Verhofstadt, L. L., Roeyers, H., Ponnet, K., & Davis, M. H. (2007). Measuring empathic tendencies: reliability and Validity of the dutch version of the interpersonal reactivity. *Psychologica Belgica*, *47*(4), 235–260. http://dx.doi.org/10.5334/pb-47-4-235

Elihami, E. (2018). *Konsep pengenalan diri*. *March*, 1–9. https://www.researchgate.net/publication/323656839_Konsep_Pengenalan_Diri/link/5aa2558ea6fdcc22e2d2ec31/download

Life Skills Education Toolkit for Orphans & Vulnerable Children in India, India – (October 2007)

Muslikah, Hariyadi, S., & Nurul Amin, Z. (2015). Pengembangan model peer counseling sebagai media pengala- man praktik konseling. *Indonesian*

Journal of Guidance and Counseling : Theory and Application, *5*(1), 39–44. journal.unnes.ac.id/sju/index.php/jbk

Prasetiawan, H. (2016). Konseling teman sebaya (peer counseling) untuk mereduksi kecanduan game online. *Counsellia: Jurnal Bimbingan Dan Konseling*, *4*(1),6475.http://ejournal.unipma.ac.id/index.php/JBK/article/download/453/421

Putu Yuli, D. (2016). *Modul Komunikasi Verbal Dan Non Verbal*.13.https://simdos.unud.ac.id/uploads/file_pendidikan_dir/a3a4fc3bf4ad19b0079f4a31c593398b.pdf

Biografi Penulis

Ghozali Rusyid Affandi, S.Psi., M.A. adalah dosen Fakultas Psikologi Universitas Muhammadiyah Sidoarjo (UMSIDA) sejak tahun 2014. Penulis menyelesaikan pendidikan S1 Psikologi di Universitas Merdeka Malang, dan melanjutkan pendidikan S2 di bidang Psikologi Pendidikan di Fakultas Psikologi UGM. Penulis merintis karya ilmiah sejak S1 dan diteruskan di S2 dengan menjadi bagian dari Center of Indigenous and Cultural Psychology (CICP) Fakultas Psikologi UGM. Penelitian yang pernah dilakukan penulis berkenaan dengan beberapa bidang, antara lain: Psikologi Pendidikan (Goal Setting, efikasi diri, School Well Being, dinamika psikologis siswa membolos serta kesiapan sekolah), bidang Psikologi Islam (Religiusitas dan karakter positif perpektif Al Qur'an), serta psikometri (analisis kualitas Nijmeegse Schoolbekwaamheids Test dan bender-gestalt test). Buku pertama yang penulis terbitkan pada tahun 2019 dengan judul "Sudah Siapkah Anak Kita untuk Sekolah: Panduan untuk Orang Tua dan Sekolah. Buku Kedua pada tahun 2020 dengan judul "Prestasi Akedemik Siswa Sekolah Dasar Tingkat Awal: Tinjauan Kesiapan Sekolah & Motivasi Berprestasi. Pada tahun 2012-2013 mengelola Jurnal Ilmiah Tabularasa di Fakultas Psikologi Universitas Merdeka Malang dan pada tahun 2014 menjadi pengelola Jurnal Ilmiah Psikologia yang dimiliki oleh Universitas Muhammadiyah Sidoarjo. Pada tahun 2014-2019 penulis menjabat sebagai Kepala Laboratorium Psikologi di Fakultas Psikologi Universitas Muhammadiyah Sidoarjo. Dan tahun 2019-saat ini penulis sebagai sekertaris Prodi Psikologi Fakultas Psikologi Universitas Muhammadiyah Sidoarjo. Email: ghozali@unsida.ac.id

Fitria Nur Hasanah, M.Pd. dilahirkan di Lamongan, 23 September 1987. Pada tahun 2008, penulis mendapatkan gelar Diploma Manajemen Informatika di Universitas Brawijaya, lulus Sarjana Pendidikan Teknik Informatika di Universitas Negeri Malang tahun 2011, kemudian melanjutkan gelar magister Pendidikan Kejuruan dengan konsentrasi Teknik Informatika di Universitas Negeri Malang lulus tahun 2015. Tahun 2011 penulis mengawali karirnya sebagai Guru di SMK Nasional Malang bidang Rekayasa Perangkat Lunak dan Teknik Komputer Jaringan. Selanjutnya tahun 2016 menjadi Dosen tetap di Prodi Pendidikan Teknologi Informasi Universitas Muhammadiyah Sidoarjo. Tahun 2018 menjabat sebagai Ketua Program Studi Pendidikan Teknologi Informasi sampai dengan sekarang, sekaligus sebagai Sekretaris Asosiasi Pendidikan Tinggi Informatika dan Komputer (APTIKOM) Provinsi Jawa Timur periode tahun 2020-2024. Beberapa penelitian yang pernah dilakukan oleh penulis adalah tentang model pembelajaran dan pengembangan media pembelajaran.

Nurfi Laili, M.Psi., Psikolog lahir di Surabaya, 28 April 1989. Penulis menamatkan studi S1 Psikologi (2012) dan Magister Psikologi Profesinya (2015) di Fakultas Psikologi Universitas Airlangga Surabaya. Kajian penelitian dan studi yang digelutinya sejak jenjang sarjana adalah tentang dunia psikologi pendidikan dan perkembangan pada anak cerdas istimewa dan remaja gifted dan juga mengenai self regulated learning dalam proses belajar mengajar. Penulis saat ini merupakan dosen Fakultas Psikologi Universitas Muhammadiyah Sidoarjo sejak tahun 2017. Sejak tahun 2020, penulis aktif melaksanakan layanan asesmen psikologi serta bertugas sebagai ketua di Pusat Pelayanan Psikologi Terapan Umsida (P3TU) Berkorespondensi dengan penulis dapat melalui nurfilaili@umsida.ac.id.

Amaliyah Syabana adalah salah satu mahasiswa Universitas Muhammadiyah Sidoarjo (UMSIDA) Prodi Psikologi semester 7. Penulis ini lahir di Sidoarjo, 07 Mei 1999 merupakan anak dari Bapak Jayadi dan Ibu Lailatul Munawaroh . Mengenai pendidikannya, Amaliyah Syabana diketahui menghabiskan masa kecilnya dengan bersekolah di SD Muhammadiyah 5 Porong. Kemudian SMP di SMPN 3 Porong. Setelah itu, dia melanjutkan pendidikannya di SMA Negeri 1 Porong. Sampai saat ini, Amaliyah Syabana menempuh pendidikan tinggi ke Fakultas Psikologi dan Ilmu Pendidikan di UNIVERSITAS MUHAMMADIYAH SIDOARJO (UMSIDA) dengan program studi yang diambil yaitu Psikologi. Selama berada di bangku perkuliahan, penulis juga aktif dalam organisasi IMM komisariat AR-RAZI sebagai ketua bidang sosial. Pada tahun 2020 penulis menjadi asisten praktikum di laboturium psikologi UMSIDA.

Rakhmat Auliya' Hidayat merupakan salah satu mahasiswa Universitas Muhammadiyah Sidoarjo (UMSIDA) Prodi Pendidikan Teknologi Informasi semester 6. Penulis ini lahir di Surabaya, 02 Juni 1999 merupakan anak dari Bapak Karyono Pribadi dan Ibu Panirah. Mengenai pendidikannya, Rakhmat Auliya' Hidayat diketahui menghabiskan masa kecilnya dengan bersekolah di SD Muhammadiyah 2 Sidoarjo. Kemudian SMP di SMP Muhammadiyah 1 Sidoarjo. Setelah itu, dia melanjutkan pendidikannya di SMA Muhammadiyah 1 Sidoarjo. Sampai saat ini, Rakhmat Auliya' Hidayat menempuh pendidikan tinggi ke Fakultas Psikologi dan Ilmu Pendidikan di Universitas Muhammadiyah Sidoarjo (UMSIDA) dengan program studi yang diambil yaitu Pendidikan Teknologi Informasi. Pada tahun 2019 penulis menjadi Ketua Bidang Kewirausahaan di organisasi Pimpinan Daerah Ikatan Pelajar Muhammadiyah (PD. IPM). Selama berada di bangku perkuliahan, penulis juga aktif dalam organisasi Media Community Visualization sebagai koordinator bidang desain.

ISBN 978-623-6081-12-9

UNIVERSITAS MUHAMMADIYAH SIDOARJO
FAKULTAS PSIKOLOGI DAN ILMU PENDIDIKAN (FPIP)
PRODI PENDIDIKAN GURU ANAK USIA DINI (PG-PAUD) TERAKREDITASI B NOMOR : 2231 SK/BAN-PT/Akred/S/VII/2017
PRODI PENDIDIKAN GURU SEKOLAH DASAR TERAKREDITASI B NOMOR : 743/SK/BAN-PT/Akred/S/III/2018
PRODI PENDIDIKAN BAHASA INGGRIS TERAKREDITASI B NOMOR : 3057/SK/BAN-PT/Akred/S/XI/2018
PRODI PENDIDIKAN ILMU PENGETAHUAN ALAM (IPA) TERAKREDITASI B NOMOR : 432/SK/BAN-PT/Akred/S/III/2019
PRODI PENDIDIKAN TEKNOLOGI INFORMASI (TI) SK NOMOR : 0207/SK/BAN-PT/Akred/S/I/2017
PRODI PSIKOLOGI TERAKREDITASI B NOMOR : 0124/SK/BAN-PT/Akred/S/III/2016
TERAKREDITASI INSTITUSI (UNIVERSITAS) B
BAN-PT No. 225/SK/BAN-PT/Akred/PT/2019
Kampus I : Jl. Mojopahit 666 B Sidoarjo, Telp: 031 – 8945444, Fax: 031-8949333
Website: www.fpip.umsida.ac.id email:fpip@umsida.ac.id

SURAT TUGAS

Nomor: 265/II.3.AU/08.00/B/KEP/II/2022

Yang bertanda tangan di bawah ini;

Nama	: Dr. Akhtim Wahyuni, M.Ag
NIK	: 202200
Pangkat/ Golongan	: Lektor/ IIId
Jabatan	: Dekan Fakultas Psikologi dan Ilmu Pendidikan
Unit Kerja	: Fakultas Psikologi dan Ilmu Pendidikan

Menugaskan:

N a m a	: **Fitria Nur Hasanah, M.Pd**
NIK	: 216613
Pangkat Golongan	: Asisten Ahli/ IIIb
Jabatan	: Dosen Tetap
Unit Kerja	: Fakultas Psikologi dan Ilmu Pendidikan

Untuk membuat Modul Pemrograman Visual Berbasisi Project di Fakultas Psikologi dan Ilmu Pendidikan Universitas Muhammadiyah Sidoarjo Semester Genap Tahun Akademik 2021/2022. Demikian surat tugas ini dibuat, untuk dapat dipergunakan sebagaimana mestinya.

Sidoarjo, 22 Februari 2022
DEKAN FPIP,

Dr. Akhtim Wahyuni, M.Ag

Modul Berbasis Proyek

PEMROGRAMAN VISUAL

Fitria Nur Hasanah, M.Pd
Dr. Rahmania Sri Untari, M.Pd

PROGRAM STUDI PENDIDIKAN TEKNOLOGI INFORMASI

Modul
Pemrograman Visual Berbasis Proyek

Penulis:

Fitria Nur Hasanah, M.Pd

Dr. Rahmania Sri Untari, M.Pd

Diterbitkan oleh
UMSIDA PRESS
Jl. Mojopahit 666 B Sidoarjo

ISBN: 978-623-464-016-8

Modul
Pemrograman Visual Berbasis Proyek

Penulis :
Fitria Nur Hasanah, M.Pd

Dr. Rahmania Sri Untari, M.Pd

ISBN : 978-623-464-016-8

Editor :
Septi Budi Sartika

M. Tanzil Multazam

Copy Editor :
Fika Megawati

Design Sampul dan Tata Letak :
Mochamad Nashrullah

Penerbit :
UMSIDA Press

Redaksi :
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit No 666B
Sidoarjo, Jawa TImur

Cetakan pertama, Mei 2022

KATA PENGANTAR

Alhamdulillah Puji Syukur Penulis sampaikan kehadirat Allah SWT, yang telah melimpahkan rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan Modul Pemrograman Visual Berbasis Proyek dengan baik. Modul ini merupakan hasil penelitian yang didanai oleh Umsida dan ditujukan kepada mahasiswa yang mengampuh mata kuliah Pemrograman Visual. Dengan dibuatnya modul ini penulis berharap agar dapat bermanfaat dan membantu dalam memahami materi Pemrograman Visual yang semakin berkembang.

Ucapkan terima kasih penulis ucapkan kepada :

1. Dr. Hidayatulloh, M.Si., Rektor UMSIDA yang memberikan kesempatan luas kepada tim penulis untuk berkarya dan menyumbangkan pikiran sehingga buku ajar ini terselesaikan.
2. Dr. Akhtim Wahyuni, M.Ag Dekan Fakultas Psikologi dan Ilmu Pendidikan yang memberikan arahan dan motivasi kepada penulis dalam menyelesaikan buku ajar ini.
3. Tim peneliti yang telah membantu dan memberikan support dalam menyelesaikan modul hasil penelitian ini.
4. Rekan-rekan dosen Pendidikan Teknologi Informasi UMSIDA yang telah berbagi pengalaman dalam penulisan modul.

Penulis menyadari bahwa modul ini masih jauh dari kesempurnaan, maka dari itu penulis mengharapkan kritik dan saran pembaca demi kesempurnaan modul ini kedepannya. Akhir kata penulis mengucapkan terima kasih, mudah-mudahan bermanfaat bagi para pembaca.

Sidoarjo, Mei 2022

Penulis

DAFTAR ISI

PRAKTIKUM 1. FORM DAN OBJEK KONTROL

Tujuan Praktikum

1. Memahami dan mengenal Form dan Objek Kontrol;
2. Dapat membuat aplikasi sederhana dengan menggunakan form dan objek control: Label, Text Box, Checkbox, Button, Radio Button, Combo Box, dan List Box

Dasar Teoritis

Form merupakan media interaksi antara pengguna dengan aplikasi yang dibuat. Form dapat dikatakan sebagai wadah atau penampung objek control yang akan digunakan. Form juga dapat dikatakan objek karena dapat memberikan reaksi saat menemui suatu kejadian. Form dapat dikategorikan menjadi 2 yaitu :

1. Form Dinamis

   Yaitu form yang dapat dimanipulasi atau diubah bentuk serta disisipi objek control yang berisi perintah-perintah yang diperlukan oleh aplikasi yang akan dibuat, contoh :

   - Window (Windows Form, Console, Class Library, WPF dsb.)
   - Web (ASP.Net Web Application, ASP.NET Web Servis dsb)
   - Smart Device (Smart Device Project)
   - Database(Access dan SQL Server)
   - Report (Report Aplication dan Crystal Report)

2. Form Statis

   Yaitu form yang tidak dapat dimanipulasi atau diubah bentuk serta disisipi objek control. Form ini hanya dapat dipanggil melalui kode perintah, Contoh :

   a. Form Pesan (Messagebox)

      Form yang bertugas untuk menampilkan pesan keterangan terhadap suatu kejadian yang diterima oleh aplikasi.

   b. InputBox (Kotak Input Pesan)

      Form ini digunakan untuk interaksi antara pengguna dengan aplikasi yang dibuat, dimana pengguna tersebut memasukkan suatu nilai lalu mengklik suatu tombol dan menunggu efek yang ditimbulkan oleh aplikasi yang dibuat.

Tabel OBJEK KONTROL

Objek	Keterangan
Label	Objek control yang dapat menampilkan output tetapi tidak dapat memberikan input pada saat dijalankan.
TextBox	Objek control yang dapat diberikan input pada saat program dijalankan.
Button	Objek control yang dapat mengeksekusi perintah-perintah yang telah diberikan.
Checkbox	objek control yang berfungsi untuk memilih beberapa item data. Dengan objek ini anda dapat lebih dari satu pilihan dan bahkan memilih semua pilihan tersedia.
RadioButton	fungsi hampir sama dengan checkbox, tetapi hanya dapat memilih satu pilihan yang tersedia.
Combo Box	objek control yang dapat digunakan untuk menampilkan daftar item dengan pilihan dropdown.
List Box	objek control yang jauh berbeda dengan ComboBox hanya saja pada listbox menu daftar pilihan ditampilkan secara kesuluruhan

Langkah Praktikum

1. Buat **project baru**, dengan nama **Prak2**.
2. Desain Tampilan form seperti gambar berikut ini :

3. Tambahkan table ke form seperti pada gambar diatas dan atur propertinya seperti table berikut:

Object	**Properties**	**Nilai**
Form1	Name	Form1
	Text	Form Sederhana
Label1	Name	Label1
	Text	Nama
Label2	Name	Label2
	Text	NIM
Label3	Text	Haai
	Name	Label3
Label4	Text	Mahasiswa dengan NIM
	Name	Label4
Label5	Text	…….
	Name	Labelhasilnim
Label6	Text	Prodi
	Name	Label6
Label7	Text	……
	Name	Labelhasilprodi
Label8	Text	….
	Name	Labelhasilnama
Button1	Name	Button1
	Text	Cek
GroupBox1	Text	Program Studi
	Name	GroupBox1
GroubBox2	Text	Hasil
	Name	GroupBox2
Radiobutton1	Text	Pendidikan TI
	Name	Rbti
Radiobutton1	Text	PG-PAUD
	Name	Rbpaud
Radiobutton1	Text	Pendidikan IPA
	Name	rbipa

TextBox1	Name	Textnama
TextBox2	Name	Textnim
Object	Properties	Nilai
Form	Name	Form1
	Text	Praktikum 4
Label	Text	Nama
Label	Text	Jenis Kelamin
Label	Text	Prodi
TextBox	Name	txtnama
	Text	(dikosongkan)
ComboBox	Name	cbojeniskelamin
	Items	-Laki-Laki -Perempuan
	DropDownStyle	DropDownList
ComboBox	Name	Cboprodi
	Item	- PTI - PBI - PGSD - P IPA - PG PAUD - Psikologi
	Dropdownlist	Dropwownlist
Button	Name	Btntampilkan
	Text	Tampilkan

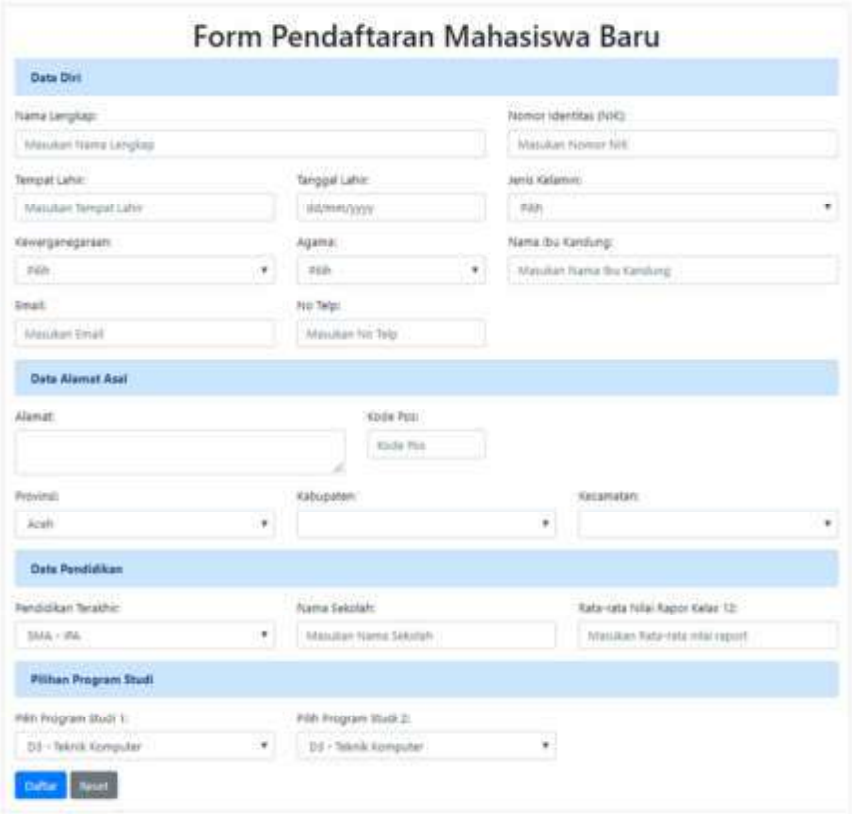
```
Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        Labelhasilnama.Text = Textnama.Text
        Labelhasilnim.Text = Textnim.Text

    End Sub

    Private Sub rbti_CheckedChanged(sender As Object, e As EventArgs) Handles rbti.CheckedChanged
        Labelhasilprodi.Text = "Pendidikan Teknologi Informasi"

    End Sub

    Private Sub rbpaud_CheckedChanged(sender As Object, e As EventArgs) Handles rbpaud.CheckedChanged
        Labelhasilprodi.Text = "PG Paud"
    End Sub

    Private Sub rbipa_CheckedChanged(sender As Object, e As EventArgs) Handles rbipa.CheckedChanged
        Labelhasilprodi.Text = "Pendidikan IPA"
    End Sub
End Class
```

Doble klik pada Button, lalu masukkan list code berikut:

```
Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        MessageBox.Show(txtnama.Text & vbCrLf & cbojeniskelamin.Text & vbCrLf & cboprodi.Text, "Hasil Pengisian Form", MessageBoxButtons.OK, MessageBoxIcon.Information
                        )

    End Sub
End Class
```

Proyek 1. Pembuatan Formulir

1. Proyek pertama yaitu membuat formulir dengan menerapkan object control pada pemrograman visual dengan mengikuti langkah praktikum
2. Lakukan modifikasi pada praktikum pertama, sehingga menjadi form pendaftaran mahasiswa baru



PRAKTIKUM 2. VARIABEL, TYPE DATA, DAN OPERATOR

Tujuan Praktikum

1. Memahami dan menerapkan penggunaan variable dalam bahasa pemrograman Visual Basic
2. Memahami dan menerapkan penggunaan type data dalam bahasa pemrograman Visual Basic
3. Memahami dan menerapkan penggunaan operator dalam bahasa pemrograman Visual Basic

Dasar Teoritis

Variabel adalah nama atau simbol yang digunakan untuk mewakili suatu nilai. Suatu variable mempunyai nama dan menyimpan tipe data yang merupakan jenis data variabel.

Aturan penamaan variabel adalah sebagai berikut:

a. Harus dimulai dengan sebuah huruf
b. Tidak lebih dari 255 karakter
c. Tidak boleh sama dengan nama statement, fungsi, metode, objek, dan sebagainya yang merupakan bahasa dari Visual BASIC.
d. Tidak boleh ada spasi, tanda titik(.), tanda seru(!), atau karakter @, &, $, dan #.

Deklarasi variabel dapat dituliskan dengan urutan sebagai berikut:

Public <nama_variabel> As <Tipe_Data>

Atau

Dim <nama_variabel> As <Tipe_Data>

```
Contoh :
Public Angka1 As Integer
Dim Nama As String
```

2. Tipe Data

Tipe data adalah jenis data yang disimpan dalam variabel. Tipe data untuk Visual BASIC adalah sebagai berikut:

Tipe Data Numerik: digunakan untuk menyimpan data numerik, terdiri dari:

Tipe	Ukuran	Range
Byte	1 byte	0 sampai 255
Integer	2 byte	-32.768 sampai 32.767
Long	4 byte	-2.147.483.648 sampai 2.147.483.647
Single	4 byte	-3,402823E38 sampai -1,401298E-45; 1,401298E-45 sampai 3,402823E38
Double	8 byte	-1.79769313486232E308 sampai -4,94065645841247E-324; 4,94065645841247E-324 sampai 1.79769313486232E308
Currency	8 byte	-922.337.203.685.477,5808 sampai 922.337.203.685.477,5807

Tipe Data String : digunakan untuk menyimpan data berbentuk karakter.Panjang maksimal karakter yang dapat disimpan adalah 65.400 karakter. Penulisan data dengan tipe ini diawali dan diakhiri dengan tanda petik dua (“).

Contoh:

```
Dim Nama As String
Nama = “Dewi”
```

Tipe Data Logika (Boolean) : melakukan pengetesan logika. Data dengan tipe data ini hanya dapat bernilai benar(True) atau salah(False).

Contoh:

```
Dim Baru As Boolean
Baru = True
```

3. Konstanta

Konstanta adalah suatu nilai konstan yang tidak berubah. Seperti halnya variabel, konstanta dapat diberi nama dimana aturan penamaannya sama dengan variabel.

Contoh: Const A = 10

4. Operator

Operator Pemberi Nilai

Deklarasi pemberian nilai pada Visual BASIC = Bahasa BASIC yaitu menggunakan operator sama dengan (=).

Contoh : a = 24

nama = “Fery Updi”

Operator Arimatika

Operator	Operasi
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
\	Pembagian dengan hasil bilangan bulat
Mod	Sisa pembagian (Modulus)

Operator Boolean

Operator	Operasi
Not	Negasi
And	Logika and
Or	Logika or
Xor	Logika xor

Operator Pembanding

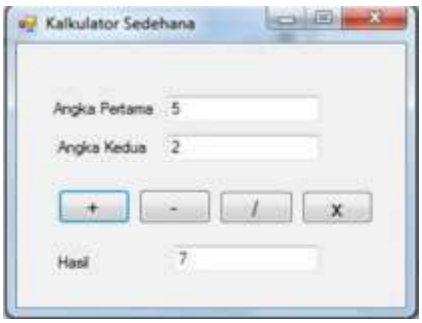
Operator	Operasi
=	Sama dengan
<>	Tidak sama dengan
<	Kurang dari
>	Lebih dari
<=	Kurang dari atau sama dengan
>=	Lebih dari atau sama dengan

Langkah Praktikum

a. Buatlah Form baru dengan nama frmkalkulator

b. Buatlah sebuah kalkulator sederhana yang bisa melakukan operasi penambahan, pengurangan, pembagian dan perkalian antara dua buah bilangan yang diinputkan oleh user.

c. Hasil program kalkulator sederhana ini kurang lebih sebagai berikut:



d. Properties name diatas adalah:

Object	Properties	Nilai
Form1	Name Text	frmkalkulator Latihan Visual Basic
Label	Name	lblpertama
	Text	Angka Pertama
Label	Name	Lbldua
	Text	Angkat Kedua
Textbox1	Name	txtAngka1
Textbox2	Name	txtAngka2
Textbox3	Name	txtHasil
Button1	Name	btnTambah
Button2	Name	btnKurang
Button3	Name	btnBagi
Button4	Name	btnKali

Listing Program untuk button tambah

```
Public Class frmkalkulator
    Private Sub btnTambah_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles btnTambah.Click
        Dim Angka1, Angka2, Hasil As Double
        Angka1 = Val(txtAngka1.Text)
        Angka2 = Val(txtAngka2.Text)
        Hasil = Angka1 + Angka2
        txthasil.Text = Hasil
    End Sub
```

Silahkan teman-teman mahasiswa melanjutkan untuk button pengurangan, pembagian, dan perkalian.

Proyek 2. Pembuatan Form Sederhana

1. Proyek 2 yaitu membuat kalkulator sederhana dengan menggunakan operator
2. Ikuti langkah pada praktikum 2 dan lakukan modivikasi sehingga menjadi sebuah aplikasi kalkulator sederhana

PRAKTIKUM 3. PERCABANGAN

Tujuan Praktikum

1. Memahami penggunaan statement if … then, else
2. Memahami penggunaan Select Case
3. Menerapkan penggunaan percabangan dalam pembuatan aplikasi

Dasar Teoritis

Pencabangan merupakan perintah yang dapat memberikan pilihan suatu kondisi, program akan menjalankan perintah apabila suatu kondisi memenuhi syarat tertentu. Pada beberapa kasus terkadang kita menginginkan komputer melakukan suatu pernyataan tertentu bila suatu kondisi terpenuhi. Dalam Visual Basic perintah percabangan/pemilihan keputusan dapat dilakukan dengan statemen **If...Then** dan **Select Case.** Pencabangan dapat dibedakan menjadi :

a. IF … THEN

If … then merupakan pencabangan yang mempunyai satu pencabangan atau satu blok perintah. Format penulisannya :

```
If kondisi then
[perintah]
End if
```

Bila <kondisi> bernilai true maka <kode program> akan dijalankan

b. IF … THEN … ELSE

Suatu perintah pencabangan bersarang (Nested If) yang merupakan perkembangan dari perintah pencabangan IF.. THEN, yang dapat menjalankan satu blok perintah, yang memiliki dua nilai atau syarat bahkan lebih yang akan diuji untuk menjalankan sautu kondisi tertentu. Bila kondisi pertama benar maka jalankan perintah blok pertama, jika kondisi pertama salah maka jalankan perintah blok kedua dan selanjutnya.

Format Penulisannya :

```
IF (kondisi) then
[perintah]
ELSE
[perintah]
```

```
End if
```

Bila <kondisi> bernilai true maka<blok kode program 1> akan dikerjakan, tetapi bila <kondisi> bernilai false maka<blok kode program 2> akan dikerjakan Dan

```
If (kondisi2) then
[perintah]
ELSEIF (kondisi2) Then
[perintah]
...
ELSE
[perintah]
End If
```

c. **Select … Case …**

Select Case adalah control pencabangan yang mempunyai fungsi hampir sama dengan pencabangan if … then… else. Select mempunyai penulisan dan pembacaan yang lebih mudah, efektif dan efesien. Namun mempunyai kelemahan yaitu tida dapat menguji lebih dari satu ekspresi atau unkapan. Format penulisannya adalah sebagai berikut :

```
Select Case kondisi
Case | Case is = ekspresi1
    [perintah1]
Case | Case is = ekspresi2
    [perintah2]
Case | Case is = ekspresi3
    [perintah3]
...
Case Else
    [perintah]
End Select
```

Langkah Praktikum

Praktikum If.. then

1. Buat **form baru.**
2. Desain Tampilan form seperti gambar berikut ini :

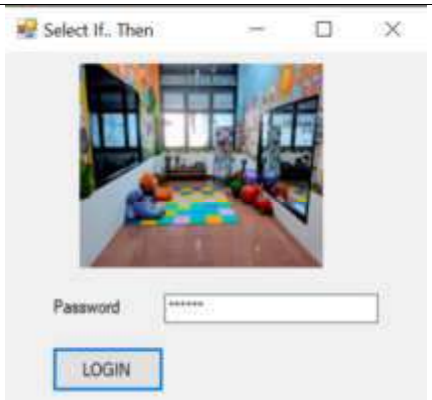
Ubah property setiap object sebagai berikut :

Object	Properties	Value
Form1	Text StartPosition	Struktur If….. Then CenterScreen
PictureBox1	Sizemode image Visible	Stretchimage Komputer.jpg (ambil dari PC gambar bebas) False
Label1	Caption	Password :
Textbox1	PasswordChar	*
Button	Text Default	LOGIN True

1. Buka jendela kode dengan **double clik** pada **Button,** dan ketikan script sebagai berikut:

```
Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        If TextBox1.Text = "fitria" Then
            PictureBox1.Visible = True
            TextBox1.Text = ""
        End If
    End Sub
End Class
```

2. Jalankan program

	
Tampilan setelah dijalankan	Jika password yag dimasukkan benar = “fitria” maka gambar akan muncul, jika tidak maka gambar tidak muncul

Praktikum 2. If … then …. Else

Modifikasi praktikum sebelumnya (if tunggal)

```
Public Class Form1

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
        If TextBox1.Text = "fitria" Then
            PictureBox1.Visible = True
            TextBox1.Text = ""
        Else
            MsgBox("Password salah")
            TextBox1.Text = ""

        End If
    End Sub
End Class
```

Proyek 3. Pembuatan aplikasi Form perhitungan dengan kriteria tertentu

Membuat aplikasi form perhitungan belanja dengan kriteria sebagai berikut:

Total Harga = Harga Satuan x Jumlah Barang

Diskon dan Bonus, didapat dengan ketentuan:

Total Harga	Diskon	Bonus
>=500 ribu	20% x Total Harga	Tas Pinggang
200 ribu – 500 ribu	15% x Total Harga	Payung
100 ribu – 200 ribu	10% x Total Harga	Kaos
50 ribu – 100 ribu	5% x Total Harga	Cangkir
< 50 ribu	0%	Tidak Ada

Total Bayar = Total Harga – Diskon

PRAKTIKUM 4. PERULANGAN

Tujuan Praktikum

Setelah mempelajari bab ini, diharapkan mahasiswa mampu:

1. Mempraktikkan penggunaan For…Next
2. Mempraktikkan penggunaan Do…Loop

Dasar Teoritis

1. For Next

Statemen ini akan mengulangi suatu blok pernyataan sebanyak jumlah yang ditentukan. Statemen ini digunakan jika banyaknya jumlah perulangan sudah diketahui.

Sintaks:

```
For <Variabel_Pengulang> = NilaiAwal To NilaiAkhir [Step Tingkat]
<Pernyataan_1>
...
<Pernyataan_n>
Next <Variabel_Pengulang>
```

Statemen ini digunakan untuk kondisi yang mempunyai nilai berurutan dan variable yang mempunyai nilai numerik. Default untuk Step adalah 1, jadi untuk perulangan dengan urutan menaik 1, nilai step tidak perlu ditulis. Sedangkan untuk perulangan menurun (Nilai awal > Nilai Akhir), nilai step diawali dengan tanda minus(-). Misalnya : For i = 10 To 1 Step -1.

Contoh:

Untuk mencetak angka 1 sampai 10 secara berurutan pada objek ListBox dapat dilakukan dengan memberi listing program sebagai berikut:

```
For i = 1 To 10
List1.Additem i
Next i
```

2. Do Loop

Statemen ini mengulang blok statemen bila kondisi benar atau sampai kondisi menjadi benar. Bila tidak ada perintah keluar, proses perulangan (*loop*) akan terus

berlangsung. Statemen ini digunakan untuk kondisi yang mempunyai nilai tidak pasti dan tidak berurutan. Statemen ini memiliki dua buah bentuk logika.

a. Statemen Do...Loop...While

Statemen ini akan mengerjakan pernyataan dalam blok statemen ketika kondisi bernilai benar, dan akan berhenti ketika kondisi sudah bernilai salah.

Sintaks:

```
Do While <Kondisi>
      <Pernyataan_1>
      ...
      <Pernyataan_n>
Loop
```

atau

```
Do
      <Pernyataan_1>
      ...
      <Pernyataan_n>
Loop While <Kondisi>
```

Contoh:

Untuk mencetak angka 1 sampai 10 secara berurutan pada objek ListBox dapat dilakukan dengan memberi listing program sebagai berikut:

```
i = 1
Do While i <= 10
      List1.Additem i i
      = i + 1
Loop
```

b. Statemen Do...Loop...Until

Statemen ini akan mengerjakan pernyataan dalam blok statemen ketika kondisi bernilai salah, dan akan berhenti ketika kondisi mencapai nilai benar.

Sintaks:

```
Do Until <Kondisi>
      <Pernyataan_1>
      ...
      <Pernyataan_n>
Loop
```

atau

```
Do
      <Pernyataan_1>
      ...
      <Pernyataan_n>
Loop Until <Kondisi>
```

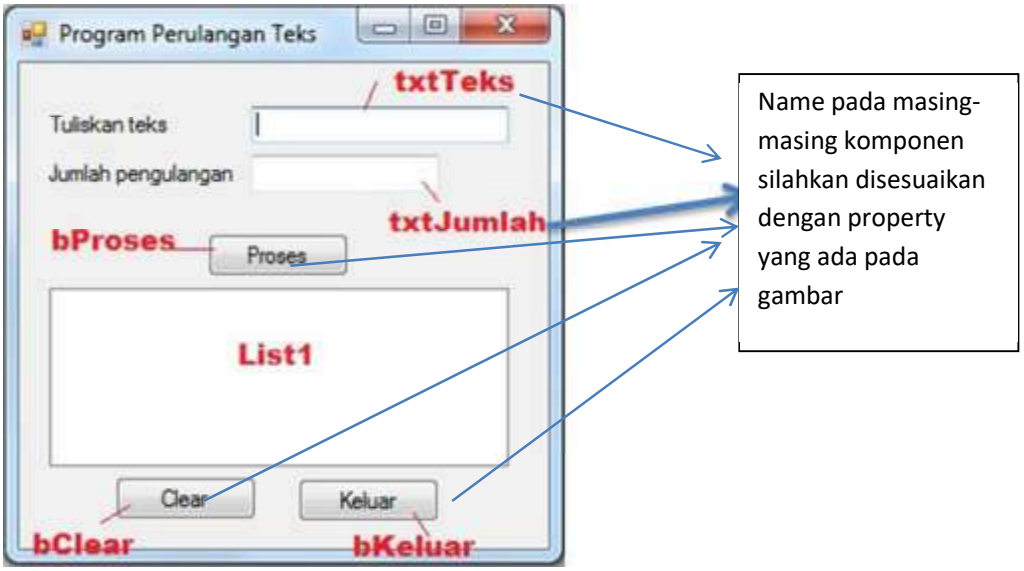
Contoh:

Untuk mencetak angka 1 sampai 10 secara berurutan pada objek ListBox dapat dilakukan dengan memberi listing program sebagai berikut:

```
i = 1
Do
     List1.Additem i
     i = i + 1
Loop Until i > 10
```

Langkah Praktikum

1. Buatlah sebuah form baru pada Visual BASIC .NET, desain tampilan form sehingga didapat tampilan seperti pada gambar



Kode Program

```
Public Class Perulangan
    Private Sub bKeluar_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles bKeluar.Click
        End
    End Sub

    Private Sub bProses_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles bProses.Click
        Dim teks As String
        Dim jumlah As Integer
        teks = txtTeks.Text
        jumlah = txtJumlah.Text
        For i = 1 To jumlah
            List1.Items.Add(teks)
        Next i
    End Sub

    Private Sub bClear_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles bClear.Click
        txtTeks.Text = ""
        txtJumlah.Text = ""
        List1.Items.Clear()
    End Sub
End Class
```

2. Buatlah sebuah form baru pada Visual BASIC .NET, desain tampilan form sehingga didapat tampilan seperti pada gambar

Kode Program :

```
Public Class Perulangan2
    Private Sub Button1_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button1.Click
        ListBox1.Items.Clear()
        For i = 1 To 10
            ListBox1.Items.Add("For Next : " & i)
        Next
    End Sub

    Private Sub Button2_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button2.Click
        ListBox1.Items.Clear()
        Dim i As Integer = 0
        Do While i <= 10
            ListBox1.Items.Add("Do While : " & i)
            i = i + 1
        Loop
    End Sub

    Private Sub Button3_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button3.Click
        ListBox1.Items.Clear()
        Dim i As Integer = 0
        Do Until i > 10
            ListBox1.Items.Add("Do Until : " & i)
            i = i + 1
        Loop
    End Sub
End Class
```

3. Buatlah sebuah form baru pada Visual BASIC .NET, desain tampilan form sehingga didapat tampilan seperti pada gambar

Object	Properties	Nilai
Form1	Name Text	Form1 Perulangan FOR
Label1	Text	Tgl/Bln/Thn
ComboBox1	Name	CmbTgl
ComboBox2	Name	CmbBln

ComboBox3	Name	CmbThn

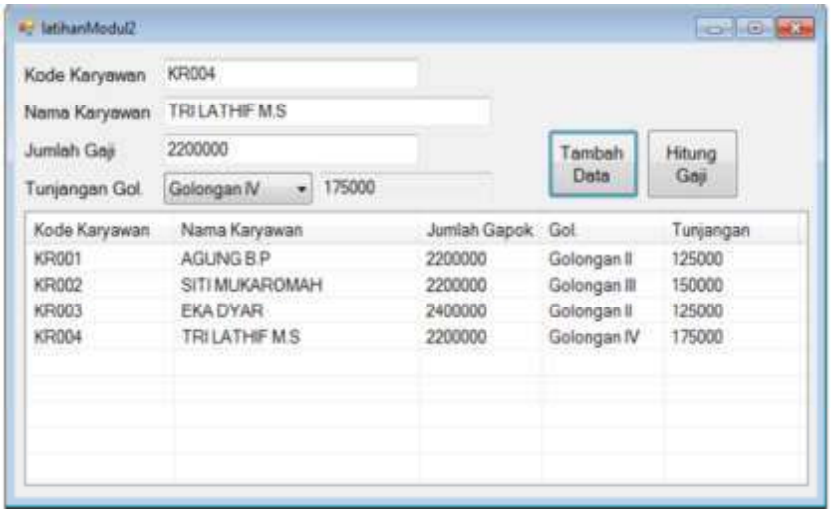
Klik ganda pada Form 1, kemudian ketik kode program berikut:

```
Dim tgl, bln, thn As Integer
    For tgl = 1 To 31
      cmbTgl.Items.Add(tgl)
    Next tgl

    For bln = 1 To 12
      cmbBln.Items.Add(bln)
    Next bln

    For thn = 1900 To 9999
      CmbThn.Items.Add(thn)
    Next thn
```

Jalankan aplikasi dengan menekan tombol **F5** (di keyboard), atau melalui ikon Start.

Proyek 4. Pembuatan Form Belanja

Pembuatan form penggajian karyawan dengan menerapkan perulangan di dalamnya.



PRAKTIKUM 5. APLIKASI MDI (MULTIPLE DOCUMENT INTERFACE)

Tujuan Praktikum

1. Memahami dan mengenal MDI (Multiple Document Interface)
2. Dapat membuat aplikasi sederhana dengan menggunakan MDI

Dasar Teoritis

MDI Form Biasanya digunakan untuk membuat suatu program aplikasi berbentuk Multiple Document Interface. MDI Form biasanya terdiri atas beberapa Form lain yang disebut dengan MDI Child. Berikut Contoh Aplikasi **MDI Form** :

Sebuah MDI Form dapat dibuat dari sebuah Form biasa (Windows Form), yaitu dengan cara mengatur properti **IsMdiContsiner** pada Form Induk tersebut.

Pada gambar diatas, sebuah Form bernama **Form_MDI** diatur menjadi suatu **MDI Form** dengan cara mengubah isi **IsMdiContainer**-nya menjadi **True**.

Sebuah Form dapat diatur menjadi **MDI Child** bagi sebuah **MDI Form** melalui kode program. Contoh Codingnya sebagai berikut :

Form1.Mdiparent = Form_MDI

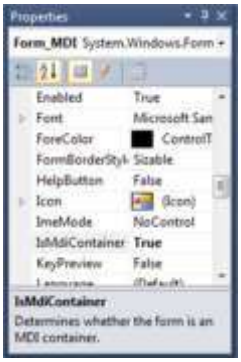
Pada contoh kode program di atas, sebuah Form bernama **Form2** diatur agar menjadi MDI Child bagi sebuah MDI Form bernama **Form MDI.**

Langkah Praktikum

1. Buatlah Sebuah Project Baru bernama **MDI Parent.**
2. Tambahkan dua buah Form baru kedalam Project (dengan cara klik kanan pada **MDI Parent** yang ada pada **Solution Explorer** kemudian > **add** > **new item** dan pilih **Windows Form)**. Sehingga sekarang ada 3 Form. Kemudian ganti nama Form 3 menjadi **MDI Parent**. Sehingga jadi seperti dibawah ini :

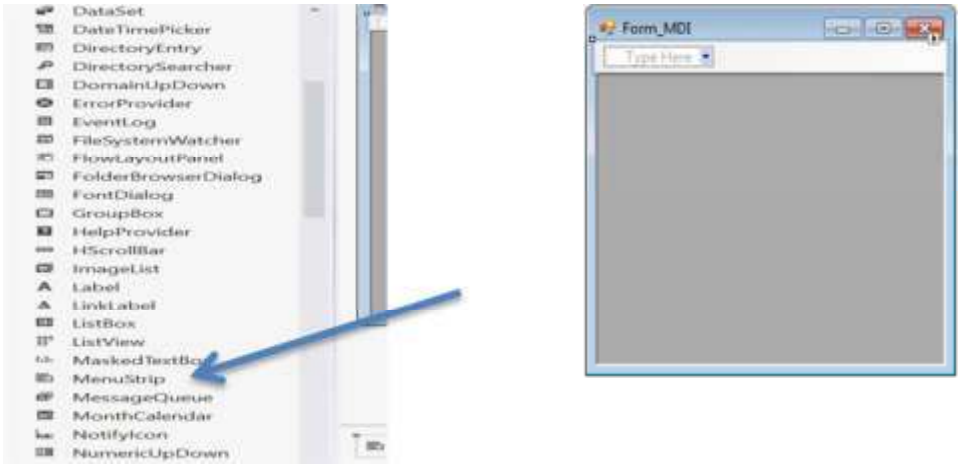
3. Kemudian Klik **MDI Form** dan pada **Solution Explorer**, klik **View Designer** untuk mengaktifkan **Form MDI.**

4. Ubah property **IsMdiContainer** pada **Form MDI** menjadi **True.**



5. Sekarang **Form MDI** telah menjadi sebuah ***MDI Form***.

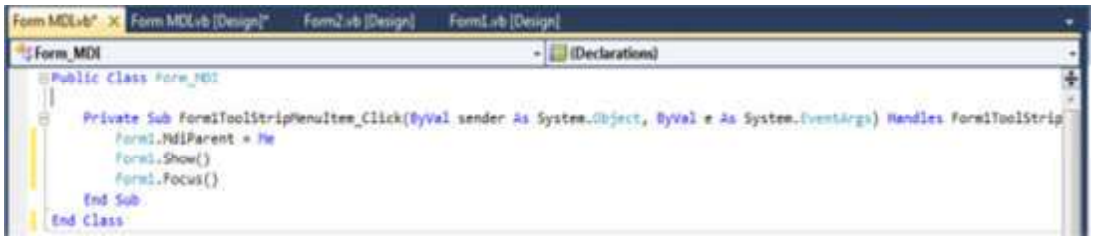
6. Tambahkan kontrol **Menu Strip** pada **Form MDI** sehingga **Form MDI** akan terlihat seperti berikut :



7. Lalu buatlah beberapa menu pada **Form MDI** seperti contoh dibawah ini.

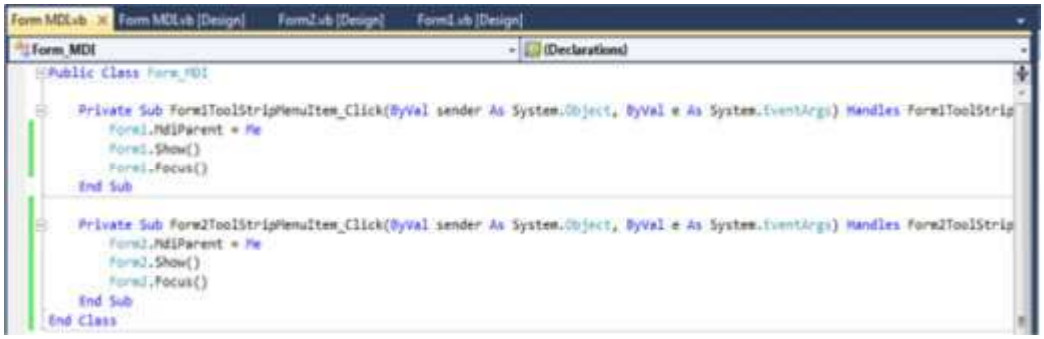
8. Sekarang double klik **Form1** pada **Menu Strip Form MDI** tadi.
9. Sehingga akan muncul kode program seperti dibawah ini.

10. Selanjutnya isi kode program pada menu**, jadikan Form1** sebagai **MDI Child** dari **Form MDI** tersebut. Gunakan coding **Me**, yang berarti diri sendiri (**MDI Form)** dan metode **Show** dan **Focus** untuk menampilkan **Form1** dan juga membuat **Form1** menjadi tampil yang paling depan nantinya, contoh :

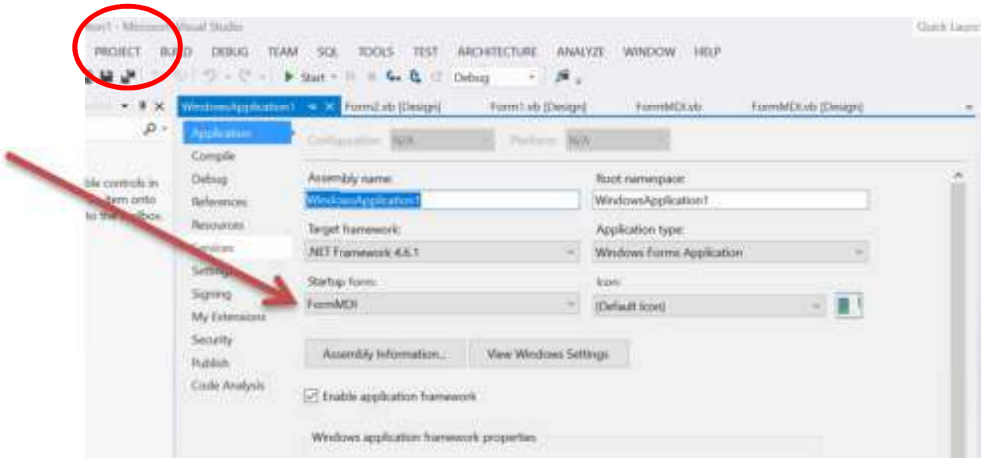


11. Tampilkan kembali **Form MDI**, Lalu klik double lagi pada menu **Form2.**

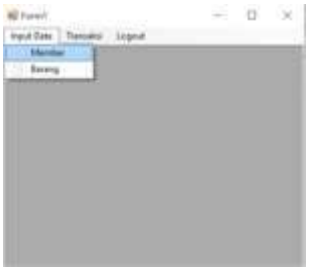
12. Lengkapi kode program pada **Form MDI** sehingga kode terisi sesuai dengan fungsinya.



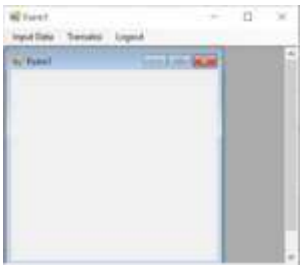
13. Selanjutnya kita harus mengtur agar ketika program dijalankan, Form yang pertama kali muncul adalah **FormMDI**.
14. Klik menu **Project > MDI Properties…** pada Menu Bar.
15. kemudian ketika muncul tampilan seperti dibawah ini, klik Tab **Application**.
16. Kemudian pada kotak pilihan **Startup form** ganti menjadi **FormMDI**.



17. Klik menu **Build > Build MDI Parent** pada menu bar untuk mengompilasi program aplikasi yang baru saja dibuat.
18. Jalankan program



19. selanjutnya coba klik menu **Member**, dan amati apa yang terjadi, **Form1** akan muncul di dalam **FormMDI** tersebut.

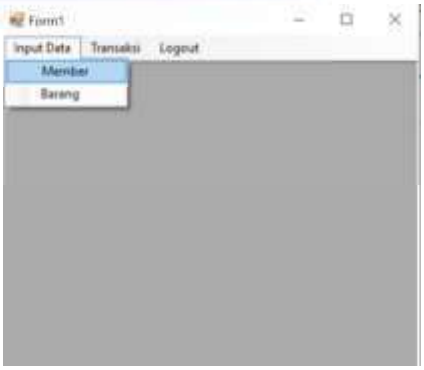


20. Begitu juga dengan **Form2**, bila diklik pada menu **Form2**, maka di dalam **FormMDI** akan tampil Form2, sehingga tampak jelas kegunaan dari MDI Form yang bisa menampung banyak Form dalam satu Form Induk.
21. Tutuplah aplikasi Anda dengan menklik menu **Debug > Stop Debugging.2.**Save Program **MDI Parent** ke folder yang anda inginkan.

Proyek 4. Pembuatan Aplikasi dengan memanfaatkan MDI

Membuat aplikasi dengan menggunakan Visual Studio tentang **tema jual beli barang** atau **perpustakaan** (silahkan dipilih salah satu) dengan kriteria :

1. Menggunakan form MDI (aplikasi terdiri dari beberapa form yang saling terhubung)
2. Diawali dengan Login,
3. Jika login berhasil maka akan menuju ke halaman utama
4. Halaman utama berisi menu, yang ketika di klik menu itu akan menuju ke halaman form, **misalnya** terdapat **menu pendaftaran member**, **input barang, menu transaksi pembayaran, exit**
5. Minimal aplikasi terdiri dari 4 form



PRAKTIKUM 6. APLIKASI DATABASE

Tujuan Praktikum

1. Memahami dan mengenal database
2. Dapat membuat membuat database Ms.Access dan Heidi SQL
3. Dapat membuat aplikasi database dengan akses data lewat komponen ADO.Net

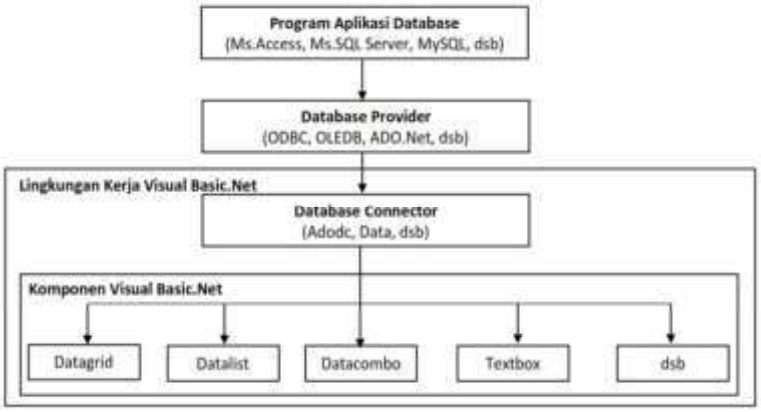
Dasar Teoritis

Database adalah informasi yang tersimpan dan tersusun rapi di dalam suatu tempat, dan dapat dengan mudah dimanipulasi seperti menambah data, menghapus, mencari, mengatur informasi yang kita butuhkan.

Database terdiri dari beberapa table, dalam satu table terdiri dari data dalam bentuk baris (record) dan kolom (field). sebuah **field** memaparkan sebuah informasi dalam tentang identitas data dalam tabel, sedangkan **record** memaparkan sebuah data dalam tabel secara lengkap.

Database Management System atau yang biasa disingkat dengan DBMS merupakan perangkat lunak atau program komputer yang dirancang secara khusus untuk memudahkan pengelolaan database. Salah satu macam DBMS yang populer dewasa ini berupa RDBMS (*Relational DataBase Management System*), yang menggunakan model basis data relasional atau dalam bentuk tabel-tabel yang saling terhubungkan. Microsoft Access, Microsoft SQL Server, Heidi SQL, Navicat, dan MySQL merupakan contoh produk RDBMS. Pemrograman Database (*Database Programming*) merupakan suatu bentuk pemrograman alternatif untuk pengolahan database.

Dengan pemrograman database kita dapat secara leluasa mengatur tampilan dan alur kerja sebuah database dengan lebih baik. Visual BASIC.Net merupakan salah satu bahasa pemrograman yang telah mendukung pemrograman database. Visual BASIC.Net dapat dihubungkan dengan program aplikasi pengolah data lain seperti Access, MySQL, SQL Server dan sebagainya. Alur kerja pemrograman database dalam Visual BASIC.Net dapat dijelaskan melalui Gambar 1.



Gambar 2. Alur kerja pemrograman database dalam Visual Basic

Koneksi Visual Basic dengan Database

Untuk dapat menghubungkan Visual Basic.Net dengan database, kita akan menggunakan komponen ADO Data Control (ADODC). Komponen ini dapat dihubungkan dengan beberapa komponen yang digunakan untuk mengakses data seperti textbox, datagrid, dsb.

Data Provider

Data provider bertanggung jawab untuk menyediakan dan menghubungkan koneksi ke database. NET Framework saat ini dilengkapi dengan dua DataProvider yaitu:

a. SQL Data Provider yang dirancang hanya untuk bekerja dengan SQL Server
b. Daya Provider OLEDB yang memungkinkan untuk terhubung ke database jenis lain seperti Access, MySQL dan Oracle.

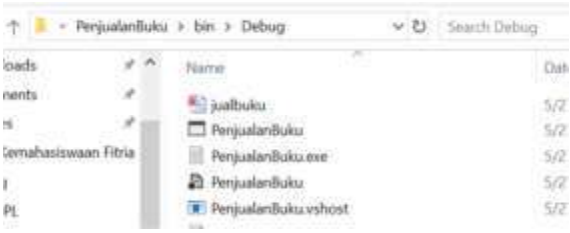
Setiap Data Provider terdiri dari kelas komponen berikut:

a. Object Connection yang menyediakan koneksi ke database
b. Object Command yang digunakan untuk mengeksekusi perintah
c. Object DataReader yang menyediakan fungsi forward-only, read-only, recordset

Langkah Praktikum

Langkah-langkah dalam membuat database adalah sebagai berikut :

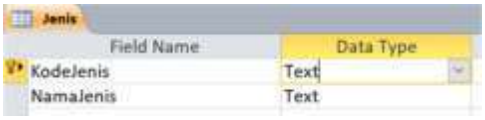
1. Buatlah database dengan Microsoft access dengan nama **JualBuku** lalu simpan kedalam folder tempat kalian menyimpan **project visual basic Anda > bin > debug**, misalnya : Project vb kalian simpan dengan nama folder **PenjualanBuku** Project VB\PenjualanBuku\PenjualanBuku\bin\Debug



Langkah pembuatan database dengan ms access, buka Ms Access lalu buat **nama database**

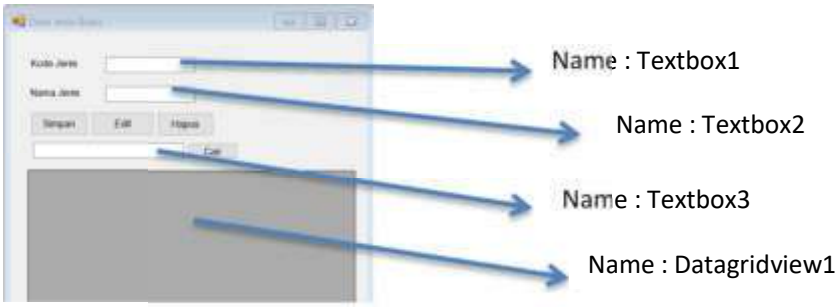
JualBuku lalu pilih lokasi penyimpanan pada **folder yang sama dengan form visual basic** , kemudian **klik create**

2. Buat table dengan nama **Jenis** yang terdiri dari 2 kolom yaitu **KodeJenis** dan **NamaJenis**
   Dengan cara pilih menu create – table, klik kanan pada table yang baru dibuat pilih desain view, lalu beri nama table dengan **Jenis** lalu klik ok, buat field dan type data seperti gambar di bawah ini:



Praktikum II. Menghubungkan form yang telah dibuat dengan database **jualbuku**

1. Buat sebuah form pada visual studio, dengan tampilan



2. Untuk membuat koneksi pada database dengan visual studi maka tambahkan sebuah module (klik kanan pada menu **Project** > **Add** > **Module**) kemudian tulislah script berikut:

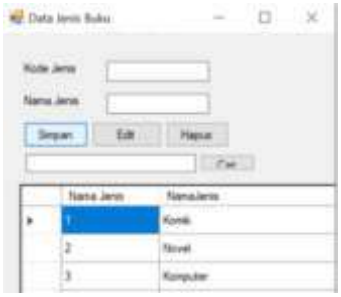
```
Imports System.Data.OleDb

Module Module1
    Public Conn As OleDbConnection
    Public da As OleDbDataAdapter
    Public ds As DataSet
    Public cmd As OleDbCommand
    Public rd As OleDbDataReader
    Public str As String
    Public Sub Koneksi()
        str = "Provider=Microsoft.ACE.OLEDB.12.0; Data Source=" & Application.StartupPath & "\jualbuku.accdb"
        Conn = New OleDbConnection(str)
        If Conn.State = ConnectionState.Closed Then
            Conn.Open()

        End If

    End Sub
End Module
```

Jualbuku.accdb adalah nama database yang telah dibuat sebelumnya dan harus diletakkan satu folder yang sama

3. Kemudian kembali pada form yang telah dibuat, double klik pada form, lalu ketik koding berikut

```
Form1.vb*   Module1.vb   Form1.vb [Design]*
Form1                                   aturgrid
Imports System.Data.OleDb
Public Class Form1
    Sub Kosong()
        TextBox1.Clear()
        TextBox2.Clear()
        TextBox3.Clear()
    End Sub

    Sub isi()
        TextBox2.Clear()
        TextBox2.Focus()
    End Sub

    Sub tampiljenis()
        da = New OleDbDataAdapter("Select * From Jenis", Conn)
        ds = New DataSet
        ds.Clear()
        da.Fill(ds, "Jenis")
        DataGridView1.DataSource = ds.Tables("Jenis")
        DataGridView1.Refresh()
    Private Sub Form1_Load(sender As Object, e As EventArgs) Handles MyBase.Load
        Call Koneksi()
        Call tampiljenis()
        Call Kosong()
        Call aturgrid()
    End Sub
    End Sub
```

Jika script ini berhasil, maka isi dari table Jenis yang ada pada database JualBuku akan muncul



Ingat pada mata kuliah Basis Data/ Database :

Perintah query untuk menampilkan isi keseluruhan **table Jenis** adalah :

Select * from Jenis

4. Menambahkan **fungsi simpan** pada form dengan **Double klik pada button simpan**, lalu tambahkan script dan query untuk melakukan penyimpanan pada database yang sudah dibuat.

```
Private Sub Button1_Click(sender As Object, e As EventArgs) Handles Button1.Click
    If TextBox1.Text = "" Or TextBox2.Text = "" Then
        MsgBox("Silahkan lengkapi isian data...!")
        TextBox1.Focus()
    Else
        cmd = New OleDbCommand("select * from Jenis where KodeJenis='" & TextBox1.Text & "'", Conn)
        rd = cmd.ExecuteReader
        rd.Read()
        If Not rd.HasRows Then
            Dim Simpan As String = "insert into Jenis(KodeJenis, NamaJenis) values" & _
              "('" & TextBox1.Text & "','" & TextBox2.Text & "')"
            cmd = New OleDbCommand(Simpan, Conn)
            cmd.ExecuteNonQuery()
            MsgBox("Simpan data sukses....!", MsgBoxStyle.Information, "Perhatian")
        End If
        Call tampiljenis()
        Call Kosong()
        TextBox1.Focus()
    End If
End Sub
```

5. Untuk melakukan editing data, dapat klik pada Datagridview1, maka data akan langsung terisi ke Textbox1 dan Textbox2. Double klik pada Datagridview1, Skript yang digunakan adalah

```
Private Sub DataGridView1_CellContentClick(ByVal sender As Object, ByVal e As System.Windows.Forms.DataGridView
    Dim i As Integer
    i = Me.DataGridView1.CurrentRow.Index
    With DataGridView1.Rows.Item(i)
        Me.TextBox1.Text = .Cells(0).Value
        Me.TextBox2.Text = .Cells(1).Value
    End With
End Sub
```

Untuk melakukan **edit data**, double klik pada **button edit**, Script untuk **edit / ubah** data adalah

```
Private Sub Button2_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button2.Click
    If TextBox1.Text = "" Then
        MsgBox("Kode Jenis belum diisi")
        TextBox1.Focus()
        Exit Sub
    Else
        Dim Ubah As String = "Update Jenis set " & _
        "Jenis='" & TextBox2.Text & "' " & _
        "where KodeJenis='" & TextBox1.Text & "'"
        cmd = New OleDbCommand(Ubah, Conn)
        cmd.ExecuteNonQuery()
        MsgBox("Ubah data sukses..!", MsgBoxStyle.Information, "Perhatian")
        Call TampilJenis()
        Call Kosong()
        TextBox1.Focus()
    End If
End Sub
```

6. Untuk melakukan **hapus data**, double klik pada **button hapus**, Script untuk **hapus** data adalah

```
Private Sub Button3_Click(ByVal sender As Object, ByVal e As System.EventArgs) Handles Button3.Click
    If TextBox1.Text = "" Then
        MsgBox("Kode Buku belum diisi")
        TextBox1.Focus()
        Exit Sub
    Else
        If MessageBox.Show("Yakin akan menghapus Data Jenis " & TextBox1.Text &
                           " ?", "", MessageBoxButtons.YesNo) = Windows.Forms.DialogResult.Yes Then
            cmd = New OleDbCommand("Delete * From Jenis where KodeJenis='" & TextBox1.Text & "'", Conn)
            cmd.ExecuteNonQuery()
            Call Kosong()
            Call TampilJenis()
        Else
            Call Kosong()
        End If
    End If
End Sub
```

7. Untuk melakukan **pencarian data**, double klik pada **button cari**, Script untuk **pencarian** data adalah

```
Private Sub btsearch_Click(sender As Object, e As EventArgs) Handles btsearch.Click
    cmd = New OleDbCommand("Select * from Jenis where NamaJenis like '%" & TextBox3.Text & "%'", Conn)
    rd = cmd.ExecuteReader
    rd.Read()
    If rd.HasRows Then
        da = New OleDbDataAdapter("Select * from Jenis where NamaJenis like '%" & TextBox3.Text & "%'", Conn)
        ds = New DataSet
        da.Fill(ds, "Dapat")
        DataGridView1.DataSource = ds.Tables("Dapat")
        DataGridView1.ReadOnly = True
    Else
        MsgBox("Data Tidak Ditemukan")
    End If
```

Proyek 5. Pembuatan Form Sederhana

Buatlah Form yang terhubung dengan database JualBuku (melanjutkan database praktikum), tambahkan satu table pada database tersebut dengan nama Buku yang dapat menyimpan data Buku, dengan isi kolom sebagai berikut:

1. Kode_Buku
2. Judul
3. ISBN
4. Tahun
5. Kode_Jenis
6. Pengarang
7. Penerbit
8. Harga
9. Stok

✓ Buat fungsi simpan, edit, hapus, dan pencarian pada form tersebut.

✓ Buat laporan praktikum dan tugas praktikum dengan memberikan capture coding dan capture hasil praktikum yang telah dijalankan

✓ Kirim Project anda dalam bentuk file WinRar/WinZip dengan nama folder PTI_Nama melalui elearning pemrograman visual

PRAKTIKUM 7. EXCEPTION HANDLING

Tujuan Praktikum

1. Memahami dan mengenal Exception Handling
2. Dapat membuat aplikasi sederhana dengan menggunakan Exception Handling

Dasar Teoritis

Perintah penanganan kesalahan atau yang lebih dikenal dengan sebutan Exception Handling Perintah ini digunakan untuk menangani kesalahan dalam menjalankan aplikasi, dengan adanya perintah ini setiap kesahan akan diatasi secara otamatis sesuai dengan perintah penanganan kesalahan yang telah dibuat, sehingga tidak terjadi kemacetan aplikasi yang sedang berjalan.

Struktur Exception Handling

Struktur Exception Handling atau penanganan kesalahan antara lain :

Try … Catch … Finally … End Try

Merupakan perintah penanganan kesalahan yang berfungsi untuk menangani kesalahan dalam menjalankan aplikasi. Bentuk penulisan :

```
Try
    [ Perintah ]
Catch
    [Tampilkan kesalahan]
Finally
    [ Kode program setelah perintah Try dan Catch ]
End Try
```

Contoh :

```
Try
Dim X%
X = TxtAngka.Text
Catch ex As Exception
MsgBox(ex.ToString)
MsgBox("Input angka saja")
Finally
TxtAngka.Text=""
TxtAngka.Focus()
End Try
```

Throw

Throw merupaka perintah penanganan kesalahan yang berfungsi untuk menangani kesalahan apabila perintah try…catch … Finally … End. Try tidak dapat menangani suatu kesalahan dari jawaban aplikasi.

Contoh penulisan :

Try

[Perintah]

Catch ex As Exception

[Tampilkan kesalahan]

Throw ex

// mengembalikan ke kode pemanggil

Finally

[Kode program setelah perintah Try dan Catch]

Perintah ini bersifat optional bias dibuat bias tidak

End Try

Contoh :

```
Try
Dim X%
X = TxtAngka.Text
Catch ex As Exception
MsgBox(ex.ToString)
MsgBox("Input angka saja")
Throw ex
Finally
TxtAngka.Text=""
TxtAngka.Focus()
End Try
```

On Error Resume Next

Merupakan perintah penanganan kesalahan apabila terjadi kesalahan terhadap suatu baris tertentu maka baris tersebut akan di abaikan.

Contoh :

```
On Error Resume Next
Dim A As Integer
X = TextBox1.Text
TextBox1.Text="String"
MsgBox(ex.ToString)
```

```
MsgBox("Abaikan Kesalahan")
```

On Error Goto

Merupakan perintah penanganan kesalahan apabila terjadi kesalahan terhadap suatu baris tertentu maka perintah akan melompat ke baris yang dituju akan dijalankan.

Contoh :

```
On Error Goto Pesan
Dim A As Integer
X = TextBox1.Text
TextBox1.Text="String"
Exit Sub
Pesan :
MsgBox("Abaikan Kesalahan")
```

Langkah Praktikum

1. Buat **project baru**, dengan nama **Prak12**.
2. Desain Tampilan form seperti gambar berikut ini :

3. Tambahkan kontrol ke form seperti pada gambar diatas dan atur propertinya seperti tabel berikut:

Object	Properties	Nilai
Form1	Name Text	Form1 Aplikasi Buka Gambar
OpenFileDialog1	Text	ofdBuka
PictureBox1	Name	PcbGambar
Button1	Name Text	BtnBuka Buka
Button2	Name Text	BtnExit Exit

4. Klik ganda pada Tombol Buka, kemudian ketik kode program berikut ini :

```
If OfdBuka.ShowDialog() = Windows.Forms.DialogResult.OK Then
  Try
    PcbGambar.Image = Image.FromFile(OfdBuka.FileName)
```

```
    Me.Text = "Membuka File " + OfdBuka.FileName
  Catch ex As Exception
    MessageBox.Show("File Gagal dibuka", "Error", MessageBoxButtons.OK,
    MessageBoxIcon.Error)
  End Try
End If
```

5. Klik ganda pada tombol BtnExit, Kemudian ketikkan kode program berikut ini :

```
Dim Tutup As String
Tutup = MessageBox.Show("Yakin tutup form ini ?", "Konfirmasi",
MessageBoxButtons.YesNo,MessageBoxIcon.Question)
If Tutup = MsgBoxResult.Yes Then
   End
Else
   Exit Sub
End If
```

6. Jalankan aplikasi dengan menekan tombol **F5** (di keyboard), atau melalui icon Start Debugging di toolbar, atau melalui menu **Debug > Start Debuging**.
7. Simpan aplikasi **Anda**.

Pratikum 2:

1. Tambahkan **Form baru**, dengan nama **Form2**.
2. Desain Tampilan form seperti gambar berikut ini :

3. Tambahkan kontrol ke form seperti pada gambar diatas dan atur propertinya seperti tabel berikut:

Object	**Properties**	**Nilai**
Form1	Name Text	Form1 Penanganan Kesalahan
TexBox1	Name	TxtAngka
Button1	Name Text	BtnProses Proses

4. Double klik pada tombol BtnProses, kemudian ketik kode Program berikut ini :

```
Try
   Dim X As Integer
   X = TxtAngka.Text
   Do
      X = X Mod 2
      If X = 0 Then
         MessageBox.Show("Bilangan Genap", "Informasi")
         Exit Do
      ElseIf X = 1 Then
         MessageBox.Show("Bilangan Ganjil", "Informasi")
         Exit Do
      End If
   Loop While Not X

Catch ex As Exception
   MsgBox("Salah menginput data / input angka")
Finally
         TxtAngka.Text=""
   TxtAngka.Focus()
End Try
```

5. Jalankan aplikasi dengan menekan tombol **F5** (di keyboard), atau melalui icon Start Debugging di toolbar, atau melalui menu **Debug > Start Debuging**. Simpan aplikasi **Anda.**

Proyek 6. Penerapan Error Handling pada Aplikasi

Membuat penerapan error handling pada aplikasi yang telah menerapkan database, dapat menggunakan project sebelumnya. Misalkan ketika harusnya inputan data berupa angka tetapi salah input berupa huruf maka akan muncul pesan, Input data angka atau silahkan terapkan error handling yang lainnya

PRAKTIKUM 8. PEMBUATAN REPORT PADA VB

Tujuan Praktikum

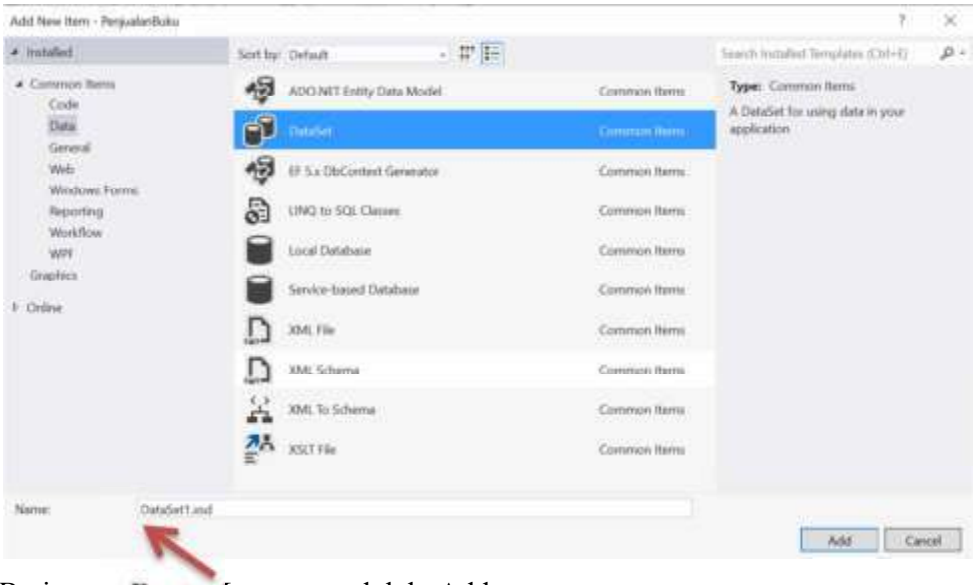
1. Memahami dan mengenal membuat laporan
2. Dapat membuat laporan dengan DataSet dan Report Viewer

Langkah Praktikum

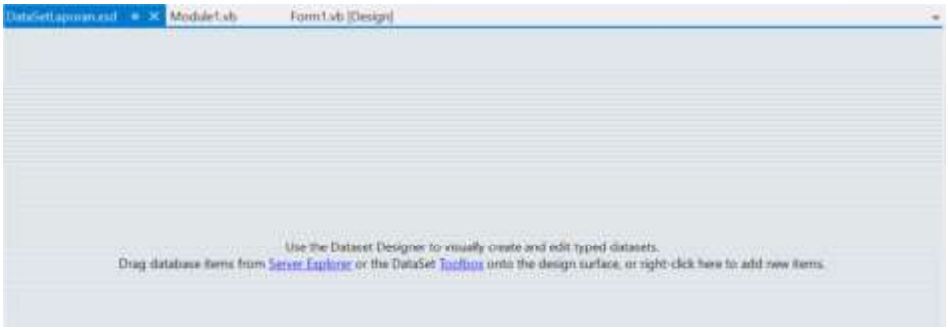
Membuat Laporan dengan DataSet

Menghubungkan DataSet dengan Data Sources ke database, langkah-langkahnya sebagai berikut:

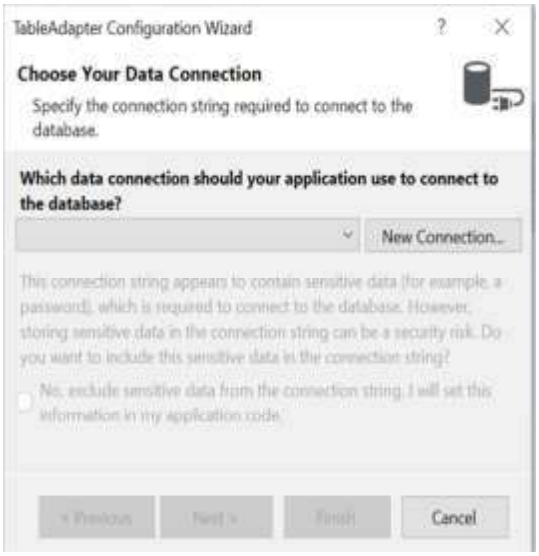
1. Pilih project – add new item - Klik Tab menu Data – pilih Dataset

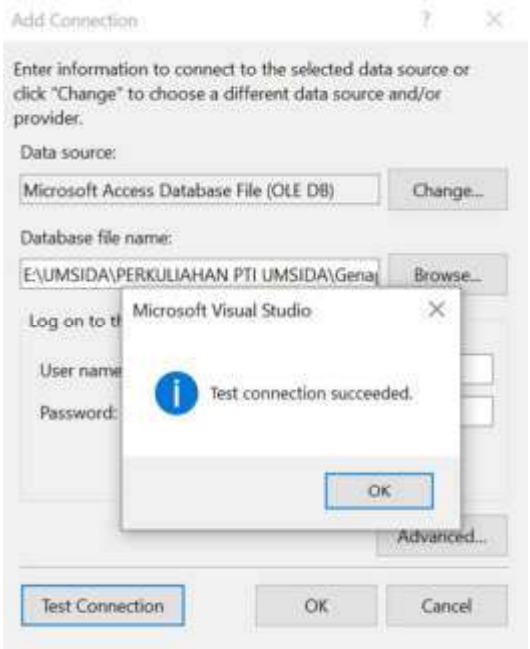


2. Beri nama DatasetLaporan.xsd, lalu Add



3. Klik kanan Add (pada lembar kerja) – Table Adapter – Klik New Connection

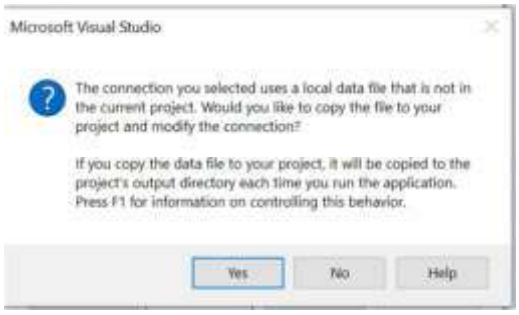




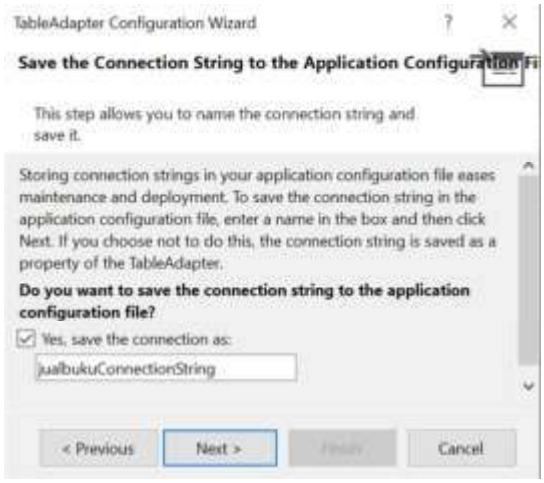
4. Pilih Data source Microsoft Access dengan klik **button Change**

Ambil database access yang telah dibuat sebelumnya, dengan klik browse, lakukan **test Connection**, jika test connection berhasil maka akan muncul seperti gambar di atas

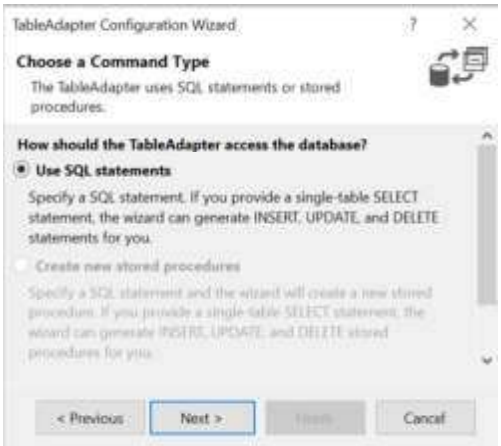
5. Klik Next – maka akan muncul pesan peringatan berikut:



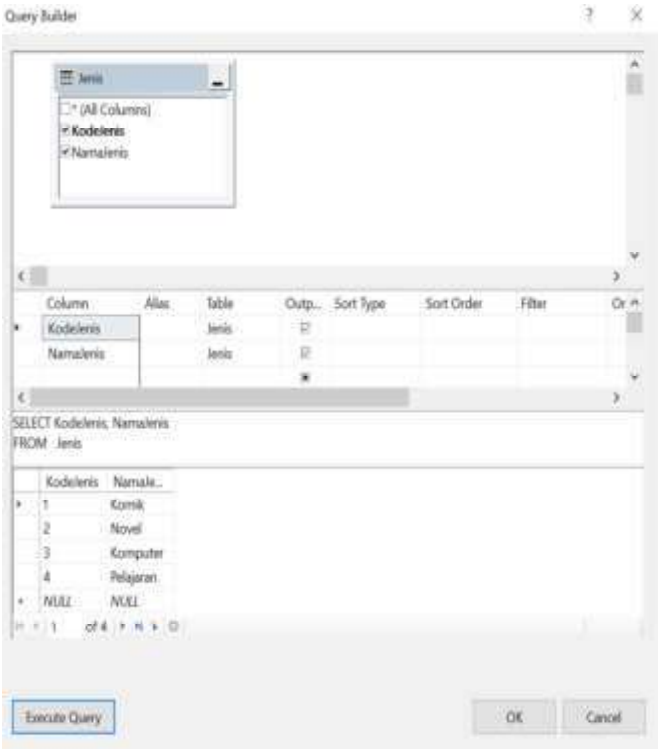
Klik yes jika ingin menyimpan kembali database, maka akan muncul halaman koneksi database, lalu pilih next



6. Masuk pada halaman query SQL



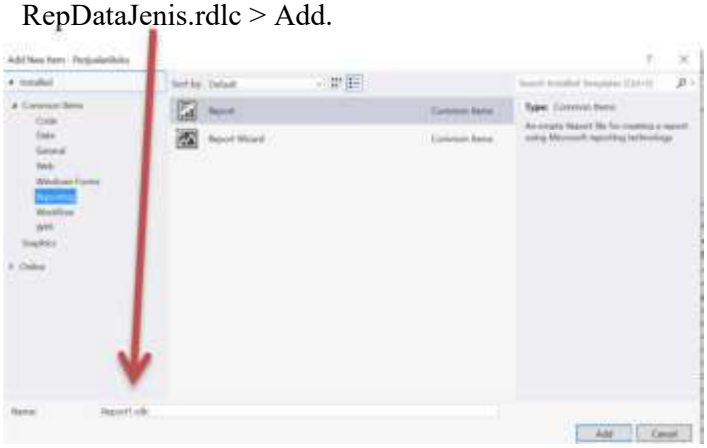
7. Klik Next, ketik query untuk menampilkan seluruh table (ingat query untuk seleksi tabel) **Select * From Jenis** atau bisa **klik Query Builder,** maka akan muncul



8. Centang All Columns, tes dengan **Execute Query**, jika **query berhasil** maka akan muncul isi data dari table tersebut, klik oke lalu Next - Next – Finish. Selanjutnya kita sudah memiliki table adapter yang telah dibuat.
9. Langkah selanjutnya buat yaitu buat form baru dan desain sebuah laporan

Membuat Rancangan Design Laporan dengan Report

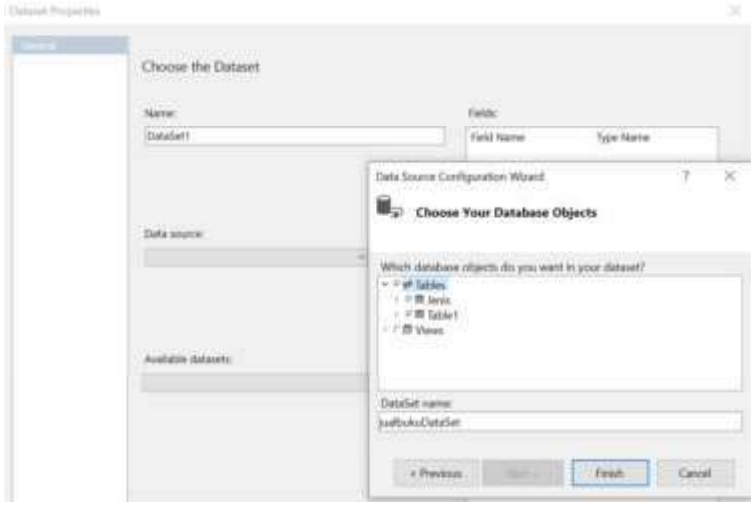
1. Membuat Form Design Report, langkah-langkahnya sebagai berikut:
2. Klik Kanan Project > Add > New Item > Reporting > Klik Report > Beri Name RepDataJenis.rdlc > Add.



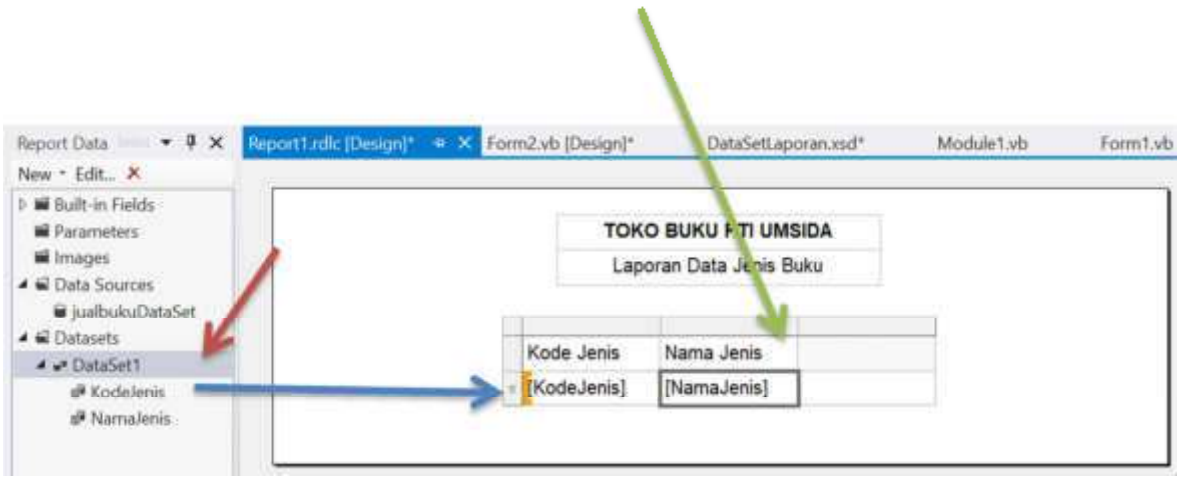
3. Akan tampil sebagai berikut:

4. Klik New, DataSet

5. Atur seperti gambar dibawah ini: lalu klik finish - ok

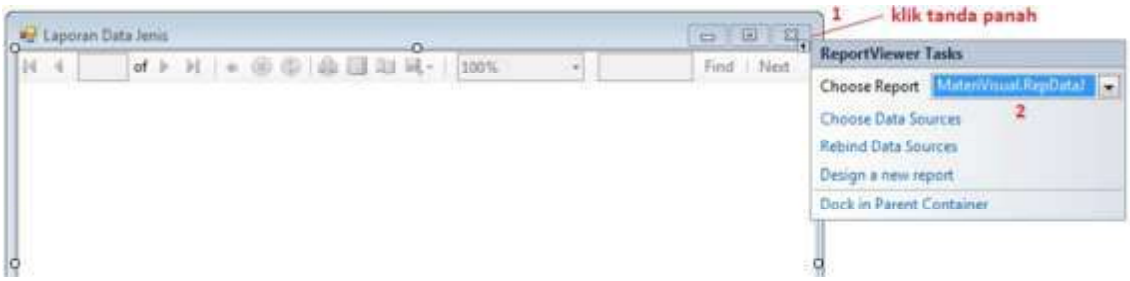


6. Rancang design tampilan report seperti berikut:
   a. Klik kanan pada area kosong pada laporan – insert – textbox – beri judul laporan yang sesuai, misal seperti pada gambar
   b. Insert Tabel – ketik **judul Kolom** pada baris pertama (header)
   c. Masukkan Field pada table dengan drag pada dataset



Membuat Rancangan Form untuk memunculkan Laporan

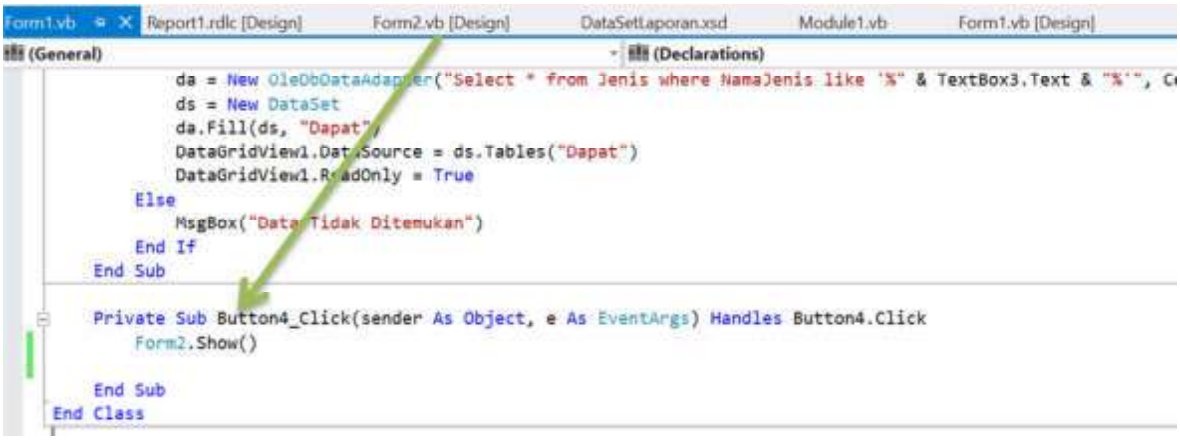
1. Buat Form baru klik kanan Project > Add > Windows Form.
2. Rancang design form laporan dengan klik pada toolbox – pilih Reporting - ReportViewer



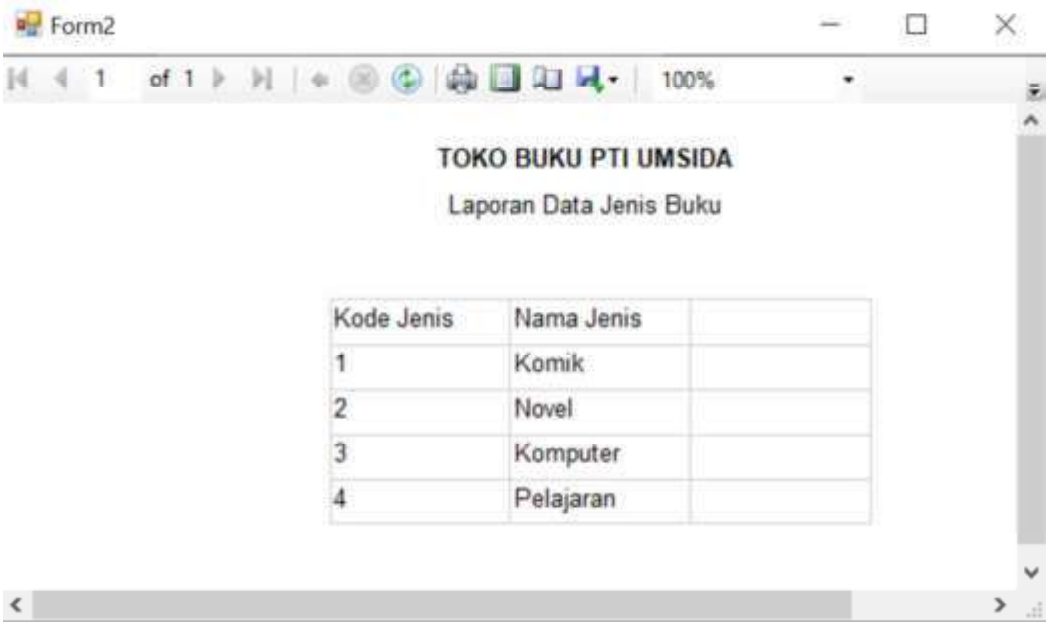
3. Pilih laporan yang telah didesain dengan klik tanda panah dan choose report – pilih report yang telah dibuat sebelumnya.

Agar halaman laporan full di tengah, ubah properties pada bagian Dock – klik tengah

4. Setelah merancang tampilan form untuk laporan langkah selanjutnya adalah menambahkan kode program ke dalam menu utama dengan cara memanggil nama Form laporan
5. Buka Form Utama atau bisa buat menu baru dan Form baru khusus laporan yang telah kalian buat sebelumnya dan double klik button – tambahkan script **NamaForm.show()**



Maka laporan sudah tercetak



Proyek 7. Pembuatan Form Sederhana

Pembuatan report untuk table yang lainnya pada praktikum sebelumnya. Silahkan menambahkan satu menu baru yaitu laporan, laporan berisikan terkait pilihan laporan yang akan dicetak berdasar table transaksi yang telah dibuat sebelumnya.

PROJECT AKHIR

Pembuatan aplikasi berbasis desktop dengan visual studio 2012 dengan memilih salah satu tema :

1. Jual beli barang
2. Perpustakaan
3. Akademik
4. Rental Mobil

Aplikasi yang dikembangkan paling tidak harus memenuhi unsur : MDI (Multipe Form), Database, Pelaporan.

DAFTAR PUSTAKA

Mardiani, dkk. 2016. Aplikasi penggajian menggunakan Visual Basic, My SQL, dan Data report. Jakarta: Elex Media Komputindo

Clark, Daniel. 2006. Beginning Object Oriented Programming with VB 2005. New York: Springer.

Evjen, B., Hollis, B., Sheldon, B., Sharkey, Kent. 2008. Professional Visual Basic 2008. Indianapolis: Wiley Publishing.

Huddleston, James. 2007. Beginning VB 2005 Databases: From Novice to Professional. New York: Springer.

McMillan, Michael. 2005. Data Structures and Algorithms Using Visual Basic.NET. New York: Cambridge University Press

Petroutsos, Evangelos. 2010. Mastering Microsoft Visual Basic 2010. Indianapolis: Wiley Publishing.

Yuswanto. 2008. Pemrograman Dasar Visual Basic.NET 2005. Jakarta: Cerdas Pustaka.

BIODATA PENULIS

Fitria Nur Hasanah, M.Pd. dilahirkan di Lamongan, 23 September 1987. Pada tahun 2008, penulis mendapatkan gelar Diploma Manajemen Informatika di Universitas Brawijaya, lulus Sarjana Pendidikan Teknik Informatika di Universitas Negeri Malang tahun 2011, kemudian melanjutkan gelar magister Pendidikan Kejuruan dengan konsentrasi Teknik Informatika di Universitas Negeri Malang lulus tahun 2015. Tahun 2011 penulis mengawali karirnya sebagai Guru di SMK Nasional Malang bidang Rekayasa Perangkat Lunak dan Teknik Komputer Jaringan. Selanjutnya tahun 2016 menjadi Dosen tetap di Prodi Pendidikan Teknologi Informasi Universitas Muhammadiyah Sidoarjo. Tahun 2018 menjabat sebagai Ketua Program Studi Pendidikan Teknologi Informasi sampai dengan sekarang, sekaligus sebagai Sekretaris Asosiasi Pendidikan Tinggi Informatika dan Komputer (APTIKOM) Provinsi Jawa Timur periode tahun 2020-2024. Beberapa penelitian yang pernah dilakukan oleh penulis adalah tentang model pembelajaran dan pengembangan media pembelajaran.

Rahmania Sri Untari, M.Pd lahir di Malang, 19 April 1989. Pendidikan Sarjana diselesaikan di Program Studi Pendidikan Teknik Informatika, Universitas Negeri Malang (UM) pada tahun 2011. Pendidikan S2 di Program Pascasarjana Pendidikan Kejuruan UM selesai pada tahun 2015. Pada tahun 2016, penulis melanjutkan S3 Pendidikan Kejuruan UM lulus tahun 2021. Pada tahun 2011 penulis memulai karirnya di SMAN 6 Surabaya sebagai guru Teknik Informatika. Selanjutnya, pada tahun 2015 penulis melanjutkan karirnya untuk menjadi Dosen Tetap di Program Studi Pendidikan Teknologi Informasi (PTI) di Universitas Muhammadiyah Sidoarjo (UMSIDA) sampai sekarang. Minat penelitian penulis adalah pada bidang pembelajaran berbasis proyek, pengembangan media pembelajaran, pendidikan kejuruan, dan bidang pendidikan lainnya.

ISBN 978-623-464-016-8 (PDF)

UNIVERSITAS MUHAMMADIYAH SIDOARJO
FAKULTAS PSIKOLOGI DAN ILMU PENDIDIKAN (FPIP)
PRODI PENDIDIKAN GURU ANAK USIA DINI (PG-PAUD) TERAKREDITASI B NOMOR : 2231 SK/BAN-PT/Akred/S/VII/2017
PRODI PENDIDIKAN GURU SEKOLAH DASAR TERAKREDITASI B NOMOR : 743/SK/BAN-PT/Akred/S/III/2018
PRODI PENDIDIKAN BAHASA INGGRIS TERAKREDITASI B NOMOR : 3057/SK/BAN-PT/Akred/S/XI/2018
PRODI PENDIDIKAN ILMU PENGETAHUAN ALAM (IPA) TERAKREDITASI B NOMOR : 432/SK/BAN-PT/Akred/S/III/2019
PRODI PENDIDIKAN TEKNOLOGI INFORMASI (TI) SK NOMOR : 0207/SK/BAN-PT/Akred/S/I/2017
PRODI PSIKOLOGI TERAKREDITASI B NOMOR : 0124/SK/BAN-PT/Akred/S/III/2016
TERAKREDITASI INSTITUSI (UNIVERSITAS) B
BAN-PT No. 225/SK/BAN-PT/Akred/PT/2019
Kampus I : Jl. Mojopahit 666 B Sidoarjo, Telp: 031 – 8945444, Fax: 031-8949333
Website: www.fpip.umsida.ac.id email:fpip@umsida.ac.id

SURAT TUGAS

Nomor: 125/II.3.AU/08.00/B/KEP/V/2019

Yang bertanda tangan di bawah ini;

11. Nama : Dr. Akhtim Wahyuni, M.Ag
12. NIK : 202200
13. Pangkat/ Golongan : Lektor/ IIId
14. Jabatan : Dekan Fakultas Psikologi dan Ilmu Pendidikan
15. Unit Kerja : Fakultas Psikologi dan Ilmu Pendidikan

Menugaskan:

11. N a m a : **Fitria Nur Hasanah, M.Pd**
12. NIK : 216613
13. Pangkat Golongan : Asisten Ahli/ IIIb
14. Jabatan : Dosen Tetap
15. Unit Kerja : Fakultas Psikologi dan Ilmu Pendidikan

Untuk membuat Modul Praktikum Mata Kuliah Object Oriented Programming (Pemrograman Berorientasi Objek) di Fakultas Psikologi dan Ilmu Pendidikan Universitas Muhammadiyah Sidoarjo Semester Ganjil Tahun Akademik 2019/2020. Demikian surat tugas ini dibuat, untuk dapat dipergunakan sebagaimana mestinya.

Sidoarjo, 10 Mei 2019
DEKAN FPIP,

Dr. Akhtim Wahyuni, M.Ag

MODUL

Pemrograman Berorientasi Objek

Berbasis Problem Based Learning

Fitria Nur Hasanah, M.Pd
Cindy Taurusta, M.T

Java

Modul
Pemrograman Berorientasi Objek

Penulis:

Fitria Nur Hasanah

Cindy Taurusta

Diterbitkan oleh

UMSIDA PRESS

Jl. Mojopahit 666 B Sidoarjo

ISBN: 978-602-5914-58-4

Modul

Pemrograman Berorientasi Object

Penulis :

Fitria Nur Hasanah

Cindy Taurusta

ISBN : 978-602-5914-58-4

Editor :
Septi Budi Sartika
M. Tanzil Multazam

Copy Editor :
Fika Megawati

Design Sampul dan Tata Letak :
Mochamad Nashrullah

Penerbit :
UMSIDA Press

Redaksi :
Universitas Muhammadiyah Sidoarjo
Jl. Mojopahit No 666B
Sidoarjo, Jawa TImur

Cetakan pertama, Juli 2019

KATA PENGANTAR

Puji syukur dipanjatkan kehadirat Allah SWT yang telah memberikan kekuatan dan kemudahan dalam menyelesaikan penyusunan modul **Pemrograman Berorientasi Objek Berbasis Problem Base Learning**. Modul ini dilengkapi dengan kajian teori dan panduan praktik yang dapat dijadikan acuan dalam praktikum jaringan dan komunikasi data. Modul ini merupakan luaran dari penelitian hibah institusi. Ucapan terima kasih disampaikan kepada semua pihak yang telah memberikan bantuan dan dukungan terhadap proses penyusunan modul ini. Selanjutnya penulis berharap semoga modul ini bermanfaat khususnya bagi penulis sendiri dan umumnya bagi mahasiswa Prodi Pendidikan Teknologi Informasi (PTI) UMSIDA

.

Sidoarjo, Juli 2019

Penulis

DAFTAR ISI

1 INSTALASI DAN SETTING JAVA DI WINDOWS

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Melakukan instalasi dan setting Java Development Kit.
2. Menggunakan Jcreator sebagai editor pemrograman
3. Menjalankan (eksekusi) program Java sederhana

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

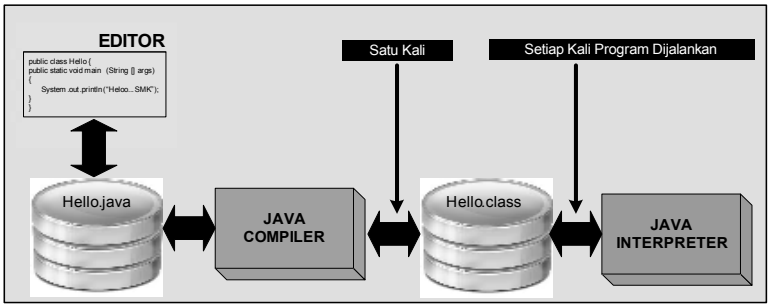
Penulisan code program JAVA dapat dilakukan dengan berbagai editor, dimulai dari yang paling sederhana yaitu notepad. Editor atau IDE (*Integrated Development Environment*) untuk Java, yang lainnya adalah:

a. NetBeans (*open source- Common Development and Distribution License* (CDDL))
b. NetBeans yang disponsori Sun Microsystems, yang dilengkapi dengan GUI Editor.
c. Eclipse JDT (*open source- Eclipse Public License*)
d. Eclipse dibuat dari kerja sama antara perusahaan-perusahaan anggota 'Eclipse
e. Oracle JDeveloper (free)
f. Xinox JCreator (ada versi berbayar maupun free)

Untuk pengembangan aplikasi berbasis Java, maka kegiatan pertama kali yang dilakukan adalah dengan menginstall software aplikasi java atau JSDK (*Java Software Development Kit* . JSDK atau JDK adalah sebuah aplikasi yang dibuat oleh perusahaan Sun Microsystems untuk membuat dan memodifikasi program java.

Fase-Fase pemrograman Java

Gambar 1 menjelaskan aliran proses kompilasi dan eksekusi sebuah program Java.



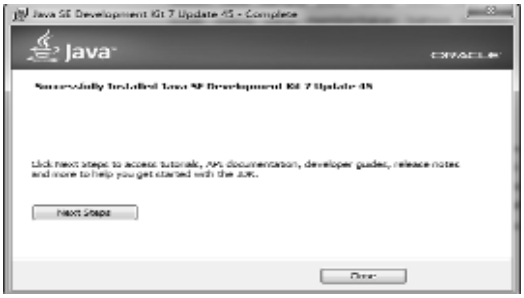
Gambar Fase dari sebuah program Java

A. PRAKTIK INSTALASI

1. Java Standart Development Kit (SDK) tersedia untuk di download pada situs Web software Java Sun Microsystem pada :http://java.sun.com.
2. Open folder tempat file-file instalasi Java SDK
3. Jalankan setup program java (contoh: **jdk-7u5-nb-7_1_2-windows-ml**)
4. Muncul kotak dialog awal instalasi JDK

Gambar Kota dialog awal instalasi java

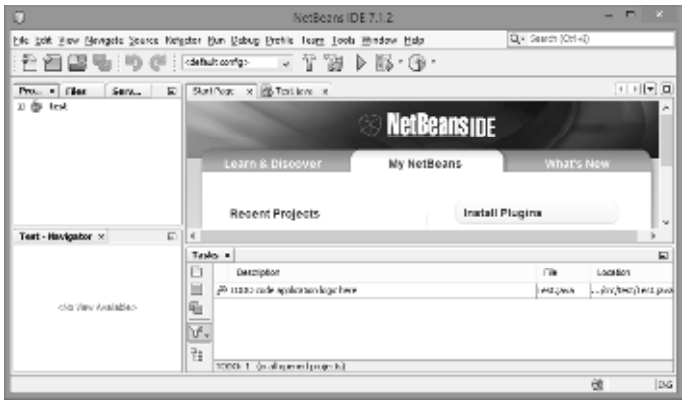
Tahapan-tahapan proses instalasi Java SDK dapat dilakukan dengan mudah dengan mengikuti petunjuk proses instalasi dengan menekan button next sampai pada tahap finish. Saat instalasi selesai, muncul kotak dialog yang memberitakan bahwa instalasi Java SDK lewatkan dengan mengklik tombol ***Finish***



Gambar Aplikasi java berhasil diinstal

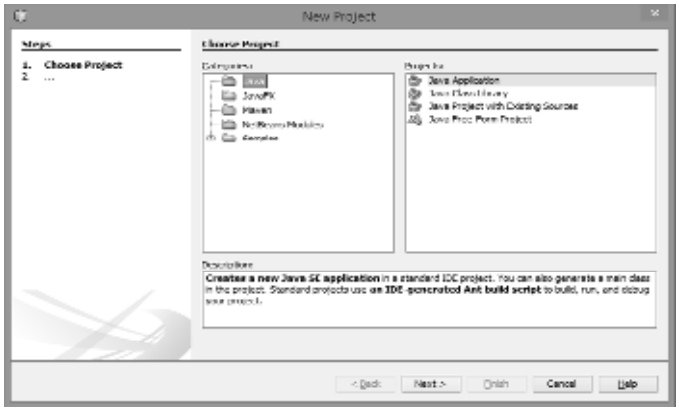
Tanda NetBeans sedang dalam proses membuka modul-modul yang diperlukan untuk

membuat aplikasi.

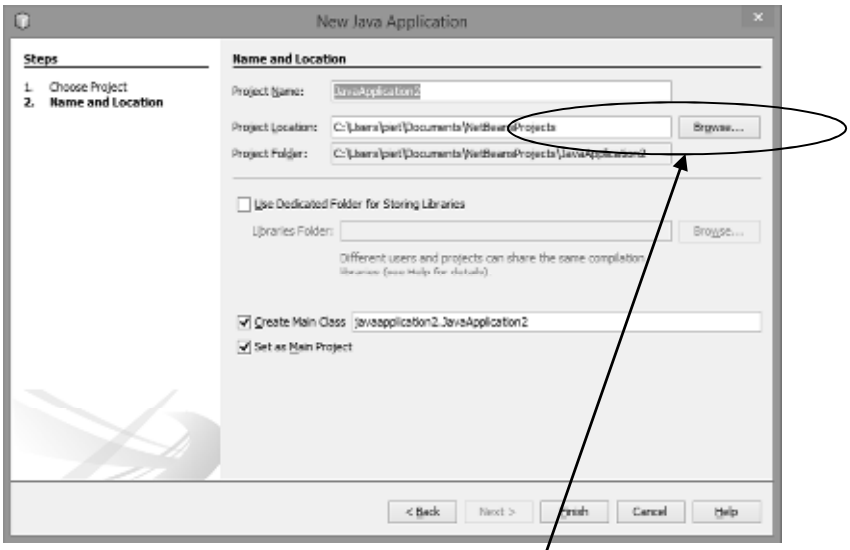


Gambar Aplikasi java dijalankan pertama kali

Untuk membuat program pertama kali buat file - new project, pastikan cursor berada pada folder (categories) java, pada menu project pilih java application. Klik next

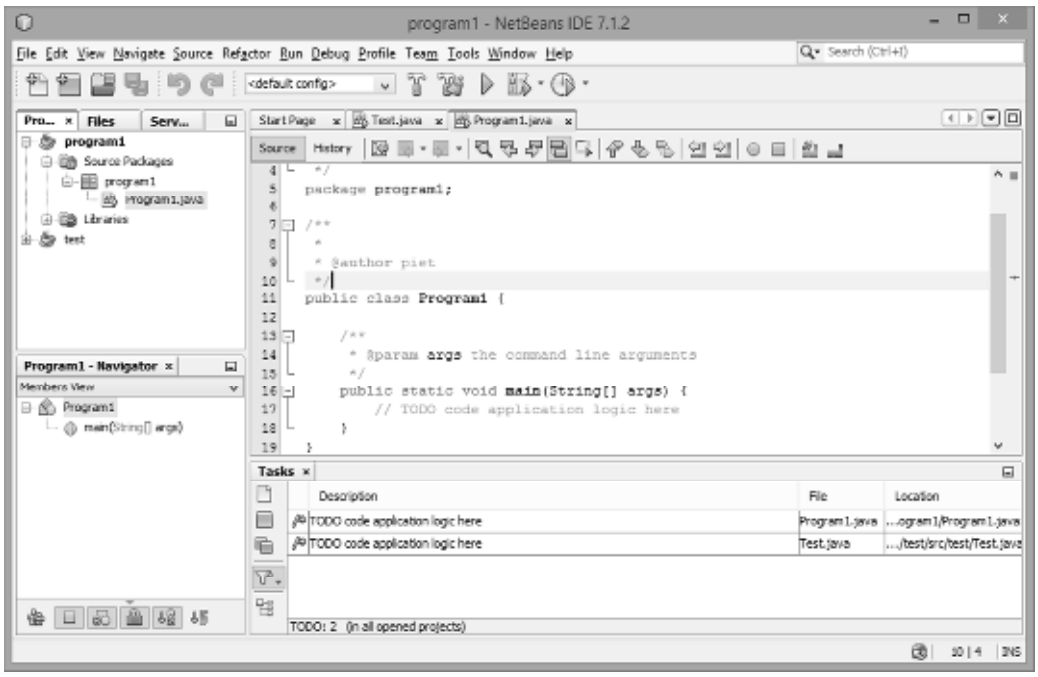


Gambar Tampilan project baru



Gambar Memberi nama dan menentukan lokasi penyimpanan

IDE NetBeans mengharuskan membuat *new Project* terlebih dahulu sebelum menulis *script* program java. Dengan cara klik File new Project , langkah berikutnya memilih aplikasi *Java Aplication*. File dengan *extension .java* dibuat untuk memulai menulis program java.pilih lokasi penyimpanan file project java pada menu project location.



Gambar Lembar kerja java (netbeans)

2 PENGENALAN PEMROGRAMAN BERORIENTASI OBYEK

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

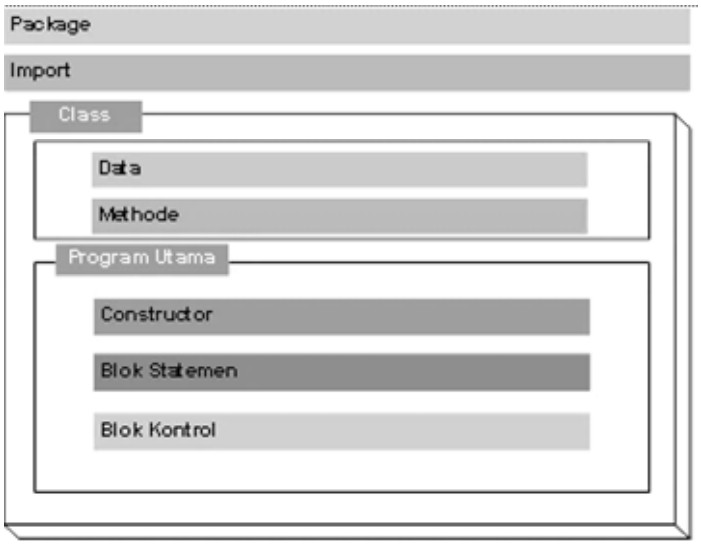
1. Mahasiswa mampu mengidentifikasi bagian dasar dari program java
2. Mahasiswa mampu membedakan tipe data dasar, tipe variabel
3. Mahasiswa mampu mengembangkan program java sederhana
4. Mahasiswa mampu mengalisa program java

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

1) Bagan dasar program java



Gambar Bagian –bagian pemrograman Java

✓ **Package**

Perintah java yang digunakan untuk memberitahukan bahwa suatu class adalah anggota dari package, sedangkan nama Package dapat berupa susunan direktori tempat dimana file class disimpan atau nama folder.

✓ **Import**

Perintah import digunakan untuk memberitahukan kepada program untuk mengacu pada class-class yang terdapat pada package tersebut dan bukan menjalankan class-class tersebut**.**

- ✓ **Class**

  Merupakan bentuk logis yang menjadi landasan bangun seluruh bahasa pemograman berorientasi object. Class mendefinisikan bentuk dan perilaku object. Class merupakan contoh abstrak dari sebuah object yang telah terbentuk dari proses penyederhanaan. Kemudian contoh nyata atau perwujudan dari sebuah object dinamakan instance.

- ✓ **Data dan Methode**

  Data merupakan identitas yang berupa variabel yang menjelaskan properti dari class. Metoda adalah sekumpulan instruksi untuk menjalankan data yang diberi nama dan dapat dipanggil dari manapun di dalam program dengan menuliskan nama metoda tersebut.

- ✓ **Program utama**

  Salah satu metoda yang paling penting di dalam bahasa Java adalah metoda main. Metoda main harus dideklarasikan sendiri oleh programmer di dalam sebuah kelas. Kelas yang mempunyai metoda main disebut dengan kelas main (main class), Interpreter Java akan meminta metoda main saat program aplikasi dieksekusi.

```
package helloworld;

public class HelloWorld {

    //program utama
    public static void main(String[] args) {
     //menampilkan kalimat PTI Bisa
        System.out.println("PTI Bisa");
    }

      }
```

Keterangan :

Baris pertama kode:

public class HelloWorld

nama class : **Helloworld**.

Tiga baris selanjutnya menandakan adanya komentar Java. Komentar adalah sesuatu yang digunakan untuk mendokumentasikan setiap bagian dari kode yang ditulis. Komentar bukan merupakan bagian dari program itu sendiri, tetapi digunakan untuk tujuan dokumentasi.

```
/**
* Program Pertama
*/
```

Komentar dinyatakan dengan tanda "/*" dan "*/". Segala sesuatu yang ada diantara tanda tersebut diabaikan oleh compiler Java, dan mereka hanya dianggap sebagai komentar.

public static void main(String[] args) {

Mengindikasikan nama suatu method dalam class **Helloworld** yang bertindak sebagai **method utama.** Method utama adalah titik awal dari suatu program Java. Baris selanjutnya juga merupakan komentar,

//Menampilkan kalimat PTI Bisa!!!

Baris selanjutnya,

System.out.println("PTI Bisa!!!");

menampilkan teks “HelloWorld!” pada layar. Perintah **System.out.println(),** menampilkan teks yang diapit oleh tanda *doublepute* (“”) pada layar. Dua baris terakhir yang terdiri atas dua kurung kurawal digunakan untuk menutup method utama dan masing-masing class secara berurutan.

E. Tipe Data Primitif

Bahasa pemrograman Java mendefinisikan delapan tipe data primitif. Diantaranya adalah boolean (untuk bentuk logika), char (untuk bentuk tekstual), byte, short, int, long (integral), double and float (floating point).

1. ***logika - boolean***

   Tipe data boolean diwakili oleh dua pernyataan : true dan false. Sebagai contoh adalah, boolean result = true; Contoh yang ditunjukkan diatas, mendeklarasikan variabel yang dinamai **result** sebagai tipe data **boolean** dan memberinya nilai **true**.

2. ***teksual – char***

   Tipe data character (char), diwakili oleh karakter single Unicode. Tipe data ini harus memiliki ciri berada dalam tanda *single quotes*(’ ’). Sebagai contoh,

   ‘a’ //Huruf a
   ‘\t’ //A tab
   Untuk menampilkan karakter khusus seperti ' (single quotes) atau " (double quotes), menggunakan karakter escape \. Sebagai contoh,
   '\'' //untuk single quotes
   '\"' //untuk double quotes

   Mereka memiliki literal yang terdapat diantara tanda double quotes(“”). Sebagai contoh,
   String message=“Hello world!”

3. ***Integral –byte, short, int & long***

   Tipe data integral dalam Java menggunakan tiga bentuk- yaitu desimal,oktal atau heksa desimal. Contohnya,

   2 //nilai desimal 2
   077 //angka 0awal mengindikasikan nilai oktal
   0xBACC //karakter 0x mengindikasikan nilai heksadesimal

   Tipe-tipe integral memiliki default tipe data yaitu int. Anda dapat merubahnya ke bentuk long dengan menambahkan huruf l atau L

4. ***Floating Point –float dan Double***

Tipe Floating point memiliki double sebagai default tipe datanya. Floating-point literal termasuk salah satunya decimal point atau salah satu dari pilihan berikut ini:

E or e //(add exponential value) F or f //(float)

Contohnya :

3.14 //nilai floating-point sederhana (a double)
6.02E23 //A nilai floating-point yang besar

Tabel Tipe Data Primitif

Grup	Type Data	Size	Min Value	Max Value
Integral	byte	8 bits	-128	128
	short	16 bits	-32768	32768
	int	32 bits	-2147483648	2147483648
	long	64 bits	-9223372036854775808	9223372036854775808
Real	float	32 bits	± 1.40239846E-45	±3.40282347E+8
	double	64 bits	±4.94065645841246544E-324	±1.79769313486231570E+308
Karakter	char	16 bits	\u0000	\uFFFF
Boolean	boolean	n/a	true atau false	

A. Variabel

Variabel adalah item yang digunakan data untuk menyimpan pernyataan objek. Variabel memiliki tipe data dan nama. Tipe data menandakan tipe nilai yang dapat dibentuk oleh variabel itu sendiri. Nama variabel harus mengikuti aturan untuk identifier.

a. Deklarasi dan Inisialisasi Variabel

Untuk deklarasi variabel adalah sebagai berikut,

<data tipe><**name>** [=initial value];

Catatan: Nilainya berada diantara <> adalah nilai yang disyaratkan, sementara nilai dalam tanda [ ] bersifat optional. Berikut ini adalah contoh program yang mendeklarasikan dan menginisialisasi beberapa variabel,

Listing Program

```
short x;
int umur;
float gaji;
```

Inisialisasi variabel dapat dilakukan dengan memberikan nilai pada variabel yang telah dideklarasikan, contoh :

Listing Program

```
int x=21;
double d = 3.5;
```

b. ***Variabel Reference dan Variabel Primitif***

Pada program java, terdapat dua type variabel **variabel reference** dan **variabel primitif** .**Variabel primitif** adalah variabel dengan tipe data primitif. Mereka menyimpan data dalam lokasi memori yang sebenarnya dimana variabel tersebut berada. **Variabel Reference** adalah variabel yang menyimpan alamat dalam lokasi memori. Yang menunjuk ke lokasi memori dimana data sebenarnya berada. Sebagai contoh, apabila kita mempunyai dua variabel dengan tipe data int dan String.

Listing Program

```
int no = 10;
String nama = "PTI ";
```

PRAKTIKUM

1) Type data karakter : **Contoh1.Java**

```
public class Contoh1 {

    public static void main(String[] args) {
    char ch1 = 65;
    char ch2 = 'B';

    System.out.println("ch1 = " + ch1);
    System.out.println("ch2 = " + ch2);
    }

}
```

a) Jalankan source kode di atas, bagaimana hasilnya?

……

……

b) Lakukan analisis pada output program jika tanda petik koma (;) dihilangkan, apa yang terjadi?

……

……

2) Penerapan method

Buat project baru dengan nama **Barang** seperti di bawah ini :

```
1   package barang;
2
3   public class Barang {
4    //pendeklarasian variabel dan inisialisasi variabel
5       String merk = "Sepatu Adidas" ;
6       int jumlah = 100;
7
8        /**
9        * main program
10       */
11      public static void main(String[] args) {
12
13
14      }
15  }
```

a) Jalankan program, Jelaskan pendapat Anda tentang hasil program!

……

……

b) Tambahkan method dengan nama **tampilkanbarang,** seperti listing program di bawah ini

```
1   package barang;
2
3   public class Barang {
4    //pendeklarasian variabel dan inisialisasi variabel
5       String merk = "Sepatu Adidas" ;
6       int jumlah = 100;
7
8       //definisi method tampilkanbarang()
9       public void tampilkanbarang(){
10        //perintah untuk menampilkan/mencetak merk
11        System.out.println("merk :" +merk);
12
13        //perintah untuk menampilkan/mencetak jumlah
14        System.out.println("jumlah:" +jumlah);
15      }
16
17       //main program
18      public static void main(String[] args) {
19
20
21      }
22  }
```

c) Tambahkan listring program pada **main program**, seperti di bawah ini

```
1  package barang;
2
3  public class Barang {
4   //pendeklarasian variabel dan inisialisasi variabel
5      String merk = "Sepatu Adidas" ;
6      int jumlah = 100;
7
8      //definisi method tampilkanbarang()
9      public void tampilkanbarang(){
10       //perintah untuk menampilkan/mencetak merk
11       System.out.println("merk :" +merk);
12
13       //perintah untuk menampilkan/mencetak jumlah
14       System.out.println("jumlah:" +jumlah);
15          }
16
17     //main program
18     public static void main(String[] args) {
19         // instanstiasi objek Contoh1
20           Barang barang = new Barang();
21
22           //perintah untuk memanggil method tampilkancontoh1()
23           //perintah untuk menampilkan data Contoh1
24           barang.tampilkanbarang();
25     }
26 }
```

Lakukan analisis terhadap listing program:

a. Bagaimana output dari program tersebut?

……

……

……

b. Jelaskan fungsi dari method **tampilkanbarang**!

……

……

……

c. Jika pada **baris ke 6** type data diubah menjadi **double,** maka output dari program adalah?

……

Tuliskan alasan jawaban Anda!

……

……

d. Jelaskan fungsi dari baris ke **24!**

……

……

TUGAS

Buatlah listing program untuk **kelas mahasiswa** yang menampilkan method dari mahasiswa tersebut. Tunjukkan keyword, tipe data dasar, dan tipe variabel di dalamnya.

Dikusikan dengan anggota kelompok Anda. Baca dengan cermat dan teliti setiap soal dibawah ini. Kemudian berdasarkan uraian materi diatas tulislah jawabannya pada lembar jawaban test yang telah disediakan.

1. Coba amati sebuah obyek di lingkungan sekitar Anda. Selanjutnya tentukan package, class, data, dan method-nya.
2. Sebutkan keyword, tipe data dasar, dan tipe variabel dari sebuah obyek yang Anda amati.

c. Lembar Jawaban

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

3 Dasar dan Aturan PBO (Operator Logika)

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Mengidentifikasi operator dalam program Java.
2. Menyajikan dalam perbedaan antara syntax error dan runtime error.

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

OPERATOR

Terdapat beberapa tipe operator pada java, yaitu operator aritmatika, operator relasi, operator logika, dan operator kondisi. Operator ini mengikuti bermacam-macam prioritas yang pasti sehingga compilernya akan tahu yang mana operator untuk dijalankan lebih dulu dalam kasus beberapa operator yang dipakai bersama-sama dalam satu pernyataan.

1. Operator Aritmatika

Berikut ini adalah dasar operator aritmatik yang dapat digunakan untuk membuat suatu program Java,

Tabel 3. Operator Aritmatika dan Fungsi-Fungsinya

Operator	*Penggunaan*	*Keterangan*
+	*op1+ op2*	*Menambahkan op1 dengan op2*
*	*op1*op2*	*Mengalikan op1 dengan op2*
/	*op1/op2*	*Membagi op1 dengan op2*
%	*op1%op2*	*Menghitung sisa dari pembagian op1 denganop2*
-	*op1-op2*	*Mengurangkan op2 dari op1*

2. Operator Increment dan Decrement

Dari sisi operator dasar aritmatika, Java juga terdiri atas operator *unary increment* (++) dan operator *unary decrement* (--). Operator increment dan decrement menambah dan mengurangi nilai yang tersimpan dalam bentuk variabel angka terhadap nilai 1.

Sebagai contoh, pernyataan,

count = count + 1
count++;

Tabel 4. Operator Increment dan Decrement

Operator	***Penggunaan***	***Keterangan***
++	*op++*	*Menambahkan nilai 1 pada op; mengevaluasi nilai op sebelum diincrementasi/ ditambahkan*
++	*++op*	*Menambahkan nilai 1 pada op; mengevaluasi nilai op setelah diincrementasi/ ditambahkan*
--	*op--*	*Mengurangkan nilai 1 pada op; mengevaluasi nilai op sebelum didecrementasi/ dikurangkan*
--	*--op*	*Mengurangkan nilai 1 pada op; mengevaluasi nilai op setelah didecrementasi/ dikurangkan*

Operator increment dan decrement dapat ditempatkan sebelum atau sesudah operand. Ketika digunakan sebelum operand, akan menyebabkan variabel diincrement atau didecrement dengan nilai1, dan kemudian nilai baru digunakan dalam pernyataan dimana dia ditambahkan. Ketika operator increment dan decrement ditempatkan setelah operand, nilai variabel yang lama akan digunakan lebih dulu dioperasikan lebih dulu terhadap pernyataan dimana dia ditambahkan. Sebagai contoh,

Listing Program 1	**Listing Program 2**

int i =10;
int j = 3;
int k = 0;
k = **++j** + i;
	int i = 10,
	int j = 3;
	int k = 0;
	k = j++ + i;

3. Operator Relasi

Operator Relasi membandingkan dua nilai dan menentukan keterhubungan diantara nilai- nilai tersebut. Hasil keluarannya berupa **nilai boolean** yaitu true atau false.

Tabel 5. Operator Relasi

Operator	***Penggunaan***	***Keterangan***
>	*op1>op2*	*op1 lebih besar dari op2*
>=	*op1>= op2*	*op1 lebih besar dari atau sama dengan op2*
<	*op1<op2*	*op1 kurang dari op2*
<=	*op1<= op2*	*op1kurang dari atau sama dengan op2*
==	*op1== op2*	*op1 sama dengan op2*
!=	*op1!= op2*	*op1 tidak sama dengan op2*

4. *Operator logika*

Operator logika memiliki satu atau lebih operand Boolean yang menghasilkan nilai boolean. Terdapat enam operator logika yaitu : && (logika AND), & (Boolean logika AND), || (logika OR), | (Boolean logika inclusive OR), ^ (Boolean logika exclusive OR), dan ! (logika NOT). Pernyataan dasar untuk operasi logika adalah x1 op x2, dimana x1,x2 dapat menjadi pernyataan boolean. Variabel atau konstanta, dan op adalah salah satu dari operator &&, &, ||, | atau ^. Tabel kebenaran yang akan ditunjukkan selanjutnya, merupakan kesimpulan dari hasil dari setiap operasi untuk semua kombinasi yang mungkin dari x1 dan x2.

5. *Operator Kondisi (?:)*

Operator kondisi **?:** adalah operator ternary. Berarti bahwa operator ini membawa tiga argumen yang membentuk suatu ekspresi bersyarat. Struktur pernyataan yang menggunakan operator kondisi adalah, **exp1?exp2:exp3**

Dimana nilai exp1 adalah suatu pernyataan Boolean yang memiliki hasil yang salah satunya harus berupa nilai true atau false. Jika exp1 bernilai true, exp2 merupakan hasil operasi. Jika bernilai false, kemudian exp3 merupakan hasil operasinya. Berikut ini adalah flowchart yang menggambarkan bagaimana operator ?: bekerja,

ERROR PADA JAVA

1. *Syntax Error*

Syntax Error biasanya terjadi karena kesalahan penulisan. Mungkin kekurangan sebuah perintah di Java atau lupa untuk menulis tanda titik koma pada akhir pernyataan. Java mencoba untuk megisolasi error tersebut dengan cara menunjukkan baris dari kode dan terlebih dahulu karakter yang salah dalam baris tersebut.

Kesalahan umum lainnya adalah dalam kapitalisasi, ejaan, penggunaan dari karakter khusus yang tidak benar, dan penghilangan dari pemberian tanda baca yang sebenarnya. *Syntax error*, merupakan jenis error yang paling mudah dideteksi. *Syntax error* yang terjadi pada kode program disebabkan kode yang diketik tidak sesuai dengan aturan/tata cara penulisan yang dimiliki oleh bahasa pemrograman yang digunakan. Contoh syntax error :

a. Pada OOP, baris kode harus selalu diakhiri dengan (;). Jika tanda ini tidak disertakan maka akan terjadi kesalahan, karena tidak sesuai dengan aturan pengkodean pada OOP.

b. Kesalahan penulisan keyword (perintah baku yang telah disediakan oleh bahasa pemrograman).

2. *Runtime Error*

Sebuah program yang berhasil dikompile belum tentu berhasil dijalankan. Inilah yang dinamakan ***Runtime error***, kesalahan ini tidak akan ditampilkan sampai kita menjalankan program tersebut. Hal ini bisa saja terjadi misalnya dikarenakan struktur yang dibuat programmer tidak jelas atau mungkin tidak logis. seperti pembagian dengan nilai nol (*division by zero error*) atau aplikasi mencoba mengakses network drive yang tidak terhubung. Sintaks benar, logika benar, tetapi pada saat eksekusi sesuatu terjadi yang akan menginterupsi proses. Tipe *error* ini biasanya terjadi saat *event* atau *syntax error* yang tidak terdefinisi menyebabkan terjadinya situasi yang membuat program *crash* atau *hang* tanpa diduga. Untuk mengantisipasi run-time error programmer harus menulis kode program untuk menangani *exception* sehingga aplikasi tidak akan di *halt*. *Exception* merupakan class khusus yang digunakan untuk mengkomunikasikan status error antara berbagai bagian aplikasi.

3. Logical Error

Logical error (kesalahan logika) terjadi pada saat aplikasi di *compile* dan dieksekusi dengan benar tetapi hasilnya tidak sesuai dengan yang diharapkan. Kesalahan ini merupakan kesalahan logika pemrograman. *Logical Error* adalah jenis kesalahan yang paling sulit ditemukan karena tidak ada pesan kesalahan (tidak ada indikasi dimana *error* terjadi). Contoh sederhana adalah terjadi kesalahan hasil perhitungan.

PRAKTIKUM

1) <u>Operator Aritmatika</u>

Buat project baru dengan nama **operator.java,** tuliskan program berikut ini dan simpan.

```
public static void main(String[] args) {
    // TODO code application logic here

System.out.println("Operasi aritmetika " + "pada tipe integer");
int a = 2 + 1
int b = a - 1
int c = a * b
int d = c / 3;
System.out.println("Nilai a: " + a)
System.out.println("Nilai b: " + b)
System.out.println("Nilai c: " + c)
System.out.println("Nilai d: " + d);
System.out.println();
```

```
System.out.println("Operasi aritmetika " + "pada tipe floating-point");
double fa = 2 + 1
double fb = fa - 1
double fc = fa * fb
double fd = fc / 3;
System.out.println("Nilai fa: " + fa);
System.out.println("Nilai fb: " + fb);
System.out.println("Nilai fc: " + fc);
System.out.println("Nilai fd: " + fd);
}
```

Lakukan kompilasi pada file tersebut dan amati hasilnya?

……

……

Kenapa terjadi error pada saat kompilasi? Termasuk jenis error apakah yang terjadi pada program tersebut?

……

……

Lakukan modivikasi untuk memperbaiki kesalahan diatas sehingga program tersebut dapat berjalan dengan baik.

……

……

……

……

2) <u>Operator Decremen dan Incremen</u>

Buat project baru dengan nama **decrement.java,** tuliskan program berikut ini dan simpan.

```
public static void main(String[] args) {
    // TODO code application logic here

System.out.println("Operasi aritmetika " + "pada tipe integer");
int a=5;
System.out.print("Pre-decrement");
System.out.print("a\t: " + a);
System.out.print("--a\t: " + --a);
System.out.println("a\t: " + a);

int b=5;
System.out.println("\nPost-decrement");
System.out.println("b\t: " + b);
System.out.println("b--\t: " + b--);
   }
```

Lakukan kompilasi pada file tersebut dan amati hasilnya?

……

……

Jelaskan perbedaan antara print dengan println?

……

……

3) Contoh modulus

Buat project baru dengan nama **modulus.java,** tuliskan program berikut ini dan simpan.

```
public static void main(String[] args) {
    // TODO code application logic here

int a=11, b=4;
int c = a % b;

double da = 13.75;
double dc = da % b;

System.out.println("Sisa bagi " + a + "/" + b + " adalah " + c);
System.out.println("Sisa bagi " + da + "/" + b + " adalah " + dc);
   }
```

Lakukan kompilasi pada file tersebut dan amati hasilnya?

……

……

……

4) Operator logika

Buat project baru dengan nama **logika.java,** tuliskan program berikut ini dan simpan.

```
public static void main(String[] args) {

System.out.println("Operasi AND");
System.out.println("true && true   = " + (true && true));
System.out.println("true && false  = " + (true && false));
System.out.println("false && true  = " + (false && true));
System.out.println("false && false = " + (false && false));

System.out.println("\nOperasi OR");
System.out.println("true || true   = " + (true || true));
System.out.println("true || false  = " + (true || false));
System.out.println("false || true  = " + (false || true));
System.out.println("false || false = " + (false || false));

System.out.println("\nOperasi XOR");
System.out.println("true ^ true    = " + (true ^ true));
System.out.println("true ^ false   = " + (true ^ false));
System.out.println("false ^ true   = " + (false ^ true));
System.out.println("false ^ false  = " + (false ^ false));

System.out.println("\nOperasi NOT");
System.out.println("!true          = " + (!true));
System.out.println("!false         = " + (!false));
   }
```

Lakukan kompilasi pada file tersebut dan amati hasilnya?

……

……

……

……

5) Operator Relasi

Buat project baru dengan nama **logika.java,** tuliskan program berikut ini dan simpan

```
public static void main(String[] args) {

int a=5, b=10;

System.out.println("a == b bernilai " + (a == b));
System.out.println("a != b bernilai " + (a != b));
System.out.println("a > b bernilai " + (a > b));
System.out.println("a < b bernilai " + (a < b));
System.out.println("a >= b bernilai " + (a >= b));
System.out.println("a <= b bernilai " + (a <= b));
   }
```

Lakukan kompilasi pada file tersebut dan amati hasilnya?

……

……

……

……

TUGAS

1. Buatlah program untuk menghitung suatu harga barang yang bernilai Rp. 200.000 dengan diskon 15%.
2. Buatlah program untuk menghitung volume balok dengan ukuran panjang = 4 cm, lebar 3 cm, dan tinggi 5 cm.

Lembar Kerja

……

……

……

……

……

4 Dasar dan Aturan PBO (Kondisi)

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami struktur kontrol pemilihan (if, else, switch)
2. Menggunakan struktur kontrol pemilihan (if, else, switch) yang digunakan untuk memilih blok kode yang akan dieksekusi

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

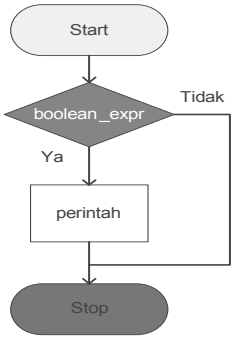
Struktur kontrol pemilihan adalah pernyataan dari Java yang mengijinkan user untuk memilih dan mengeksekusi blok kode spesifik dan mengabaikan blok kode yang lain.

1. Statement if

Pernyataan *if* akan menentukan sebuah pernyataan (atau blok kode) yang akan eksekusi jika dan hanya jika persyaratan bernilai benar *(true).* Bentuk dari pernyataan if,

Sintaks Perintah If	Sintaks Perintah If
`if(boolean_expression) statement;`	`if(boolean_expression) { statement1; statement2; }`

boolean_expression adalah sebuah penyataan logika *(true/false)* atau variabel bertipe *boolean*.



Gambar 4.1 Flowchart Statement if

2. *Statement if-else*

Pernyataan *if-else* digunakan apabila kita ingin mengeksekusi beberapa pernyataan dengan kondisi *true* dan pernyataan yang lain dengan kondisi *false*. Bentuk statement if-else,

Sintaks Perintah If

```
If  (boolean_expression)
statement;
```

Berikut ini contoh code **statement if-else,**

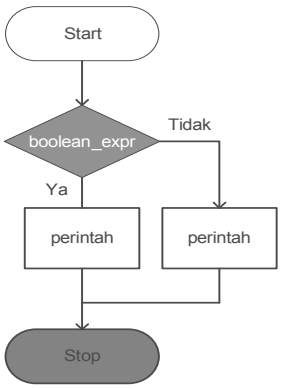
Listing Program

```
Int grade=68;
If (grade>60)
System.out.println("Congratulations!");
else
System.out.println("Sorry you failed");
```

Atau

Listing Program

```
intgrade=68;
if(grade>60)
{
System.out.println(“Congratulations!”);
System.out.println(“You passed!”);
}
Else  {
System.out.println("Sorry you failed");
}
```



Gambar 4.1 Flowchart Statement If-Else

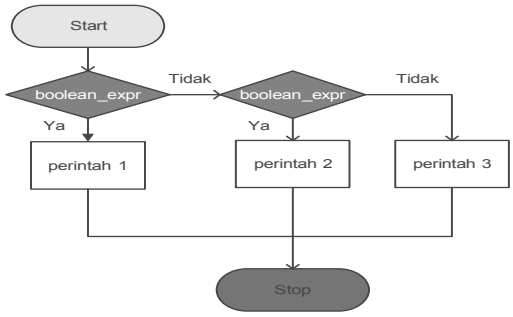
3. *Statement if-else-if*

Pernyataan pada bagian kondisi *else* dari blok *if-else* dapat menjadi struktur *if-else* yang lain. Kondisi struktur seperti ini mengijinkan kita untuk membuat seleksi persyaratan yang lebih kompleks. Bentuk statement *if-else if*.

Sintaks perintah If else If

```
if(boolean_expression1)
statement1;
else if(boolean_expression2)
statement2;
else
statement3;
```

else bersifat opsional dan dapat dihilangkan. Pada contoh yang ditampilkan diatas, jika boolean_expression1 bernilai true, maka program akan mengeksekusi statement1 dan melewati pernyataan yang lain. Jika boolean_expression2 bernilai true, maka program akan mengeksekusi statement2 dan melewati statement2.



Gambar 4.2. Flowchart Statement If-Else-If

Berikut ini contoh code statement if-else-if

Listing Program

```
Int grade=68;
if(grade>90)  {
System.out.println(" Excellent good!");
}
Else if(grade>60)
{
System.out.println("Very good!");
}
else{
System.out.println("Sorry you failed");}
```

4. Statement switch

Cara lain untuk membuat cabang adalah dengan menggunakan kata kunci ***switch***. *Switch* mengkonstruksikan cabang untuk beberapa kondisi dari nilai. Bentuk statement switch adalah sebagai berikut:

Sintaks Perintah Switch

```
switch(switch_expression)
{
Case case_selector1:
statement1;
statement2;
case case_selector2:
statement1;
statement2;
break;
default:
}
statement1;
statement2;
break;
```

PRAKTIKUM

Praktikum 1 (Struktur if satu Kondisi)

1) ***Ketikkan listing program di bawah ini:***

```
1   package struktur;
2
3   public class Struktur {
4
5       public static void main(String[] args) {
6           int grade = 68;
7            if( grade > 60 )
8            {
9                System.out.println("Congratulations!");
10           }
11      }
12      }
```

2) Bagaimana output dari program tersebut?

……

……

……

3) Jika nilai pada variable grade diubah menjadi 52, bagaimana hasilnya?

……

Lakukan analisis terhadap output program!

……

……

Praktikum 2 (Struktur If dua Kondisi)

Ketikkan listing program di bawah ini:

```
1   package struktur;
2
3   public class Struktur {
4
5       public static void main(String[] args) {
6           int a=1, b=10;
7
8       if (a < 5) {
9         System.out.println(a + " lebih kecil dari 5");
10      } else {  // (a >= 5)
11          System.out.println(a + " lebih besar dari 5");
12      }
13      }
14  }
```

Jelaskan output dari program di atas!

……

……

Praktikum 3 (Struktur tiga Kondisi)

Ketikkan listing program di bawah ini

```
1   package struktur;
2
3   public class Struktur {
4
5       public static void main(String[] args) {
6           int nilai = 86;
7
8       if (nilai > 85) {
9         System.out.println("Nilai akhir adalah A" );
10      }
11      else if (nilai>80) {
12         System.out.println("Nilai akhir adalah B");
13      }
14      else {
15        System.out.println("Nilai akhir adalah C");
16      }
17      }
18      }
```

Output dari coding program di atas adalah?

……

……

Berikan penjelasan tentang alur program tersebut!

……
……
……
……
……
……
……

*Jika variable nilai di ubah menjadi **nilai= 75,** Bagaimana hasilnya? Mengapa demikian?*

……
……
……
……

Lakukan modifikasi coding program sehingga syarat untuk mendapatkan nilai berubah menjadi : jika **nilai > 85** maka akan mendapat nilai **A**;
jika **nilai > 80** maka akan mendapat **nilai B-;**
jika **nilai > 75** maka akan mendapat nilai **B**;
jika **nilai > 70** maka akan mendapat **nilai C**;
jika **nilai > 65** maka akan mendapat **nilai C-;**
selain itu mendapatkan **nilai D**
Tuliskan listing program pada lembar kerja:

……
……
……
……
……
……
……
……
……
……
……
……
……

Praktikum 4. Switch Case

System akan menampilkan sesuai pilihan yang diinputkan, contoh listing programnya

```
1   package struktur;
2
3   public class Struktur {
4
5       public static void main(String[] args) {
6          int noHari = 7;
7
8       switch (noHari) {
9         case 1:
10          System.out.println("Hari ke-" + noHari + " adalah Minggu");
11          break;
12        case 2:
13          System.out.println("Hari ke-" + noHari + " adalah Senin");
14          break;
15        case 3:
16          System.out.println("Hari ke-" + noHari + " adalah Selasa");
17          break;
18        case 4:
19          System.out.println("Hari ke-" + noHari + " adalah Rabu");
20          break;
21        case 5:
22          System.out.println("Hari ke-" + noHari + " adalah Kamis");
23          break;
24        case 6:
25          System.out.println("Hari ke-" + noHari + " adalah Jum\'at");
26          break;
27        case 7:
28          System.out.println("Hari ke-" + noHari + " adalah Sabtu");
29          break;
30        default:
31          System.out.println("Tidak ada hari ke-" + noHari);
32      }
33      }
34  }
```

1) *Bagaimana hasil output coding program di atas ?*

   ……
   ……

2) *Jika nilai (input) variable **noHari** adalah selain **angka 1 - 7** bagaimana hasil output program? Berikan penjelasan dari analisis program tersebut?*

   ……
   ……
   ……
   ……
   ……

TUGAS

Tugas 1

Buatlah suatu program yang berfungsi mengecek suatu nilai. Jika nilai lebih dari 75 cetak “Lulus”, jika nilai kurang dari 75 cetak “Gagal”.

Tugas 2

Buatlah program pendaftaran mahasiswa baru di universitas yang memilih jurusan di universitas tersebut. Terdapat 5 program studi PTI, PGSD, PAUD, PBI, dan IPA. Jika pendaftar memilih salah satu jurusan, cetak nama jurusan yang dipilih dengan menggunakan **If** dan **switch case.**

Tuliskan listing program pada lembar kerja berikut

5 Dasar dan Aturan PBO (Perulangan)

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami struktur kontrol pengulangan (while, do-while, for)
2. Menggunakan struktur kontrol pengulangan (while, do-while, for) untuk menjalankan blok tertentu pada program beberapa kali

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Struktur kontrol pengulangan adalah berupa pernyataan dari Java yang mengijinkan kita untuk mengeksekusi blok code berulang-ulang sesuai dengan jumlah tertentu yang diinginkan. Ada tiga macam jenis dari struktur kontrol pengulangan yaitu while, do- while, dan for-loops.

1. while loop

Pernyataan whileloop adalah pernyataan atau blok pernyataan yang diulang-ulang sampai mencapai kondisi yang cocok. Bentuk pernyataan while,

```
while(boolean_expression){ statement1; statement2;}
```

Pernyataan di dalam whileloop akan di eksekusi berulang-ulang selama kondisi boolean_expression bernilai benar (true).

Contoh pada kode dibawah ini.

```
int i=4; //inisialisasi awal
    while(i>0)
    {
    System.out.print(i);
    i--;
    }
```

Contoh di atas akan mencetak angka 4321 pada layar. Perlu dicatat jika bagian i--; dihilangkan, akan menghasilkan pengulangan yang terus menerus (infiniteloop). Sehingga, ketika menggunakan whileloop atau bentuk pengulangan yang lain, pastikan Anda memberikan pernyataan yang membuat pengulangan berhenti pada suatu kondisi.

Berikut ini adalah beberapa contoh while loop,

```
intx=0;
    while(x<10)
    {
    System.out.println(x);
    x++;
    }
```

2. *do-whileloop*

Do-while loop mirip dengan while-loop. Pernyataan di dalam do-whileloop akan dieksekusi beberapa kali selama kondisi bernilai benar (true). Perbedaan antara while dan do-whileloop adalah dimana pernyataan di dalam **do-while loop** akan dieksekusi sedikitnya satu kali.

Sintaks do-while loop

```
do{
    statement1;
    statement2;
    ...
    }while(boolean_expression);
```

Pernyataan di dalam do-while loop akan dieksekusi pertama kali, dan akan dievaluasi kondisi dari boolean_expression. Jika nilai pada boolean_expression tersebut bernilai true, pernyataan di dalam do-whileloop akan dieksekusi lagi. Contoh do-while loop:

```
    intx=0;
    do
    {
    System.out.println(x);
    x++;
    }while(x<10);
```

Contoh ini akan memberikan output 0123456789 pada layar.

3. *For loop*

Pernyataan forloop memiliki kondisi hampir mirip seperti struktur pengulangan sebelumnya yaitu melakukan pengulangan untuk mengeksekusi kode yang sama sebanyak jumlah yang telah ditentukan. Bentuk dari forloop,

```
for(InitializationExpression; LoopCondition;StepExpression)
{
statement1;
statement2;
... }
```

Berikut ini adalah contoh dari for loop,

```
int i;
for(i=0;i<10;i++)
{
System.out.print(i);
}
```

Pada contoh ini, pernyataan i = 0 merupakan inisialisasi dari variabel. Selanjutnya, kondisi < 10 diperiksa. Jika kondisi bernilai true, pernyataan didalam forloop dieksekusi. Kemudian, ekspresii ++ dieksekusi, lalu akan kembali pada bagian pemeriksaan terhadap kondisii < 10 lagi. Kondisi ini akan dilakukan berulang-ulang sampai mencapai nilai yang salah (false).

PRAKTIKUM

Praktikum 1 (For)

1) Ketikkan listing program di bawah ini

```
1  package perulangan;
2
3  public class Perulangan {
4      public static void main(String[] args) {
5          for (int i=0; i<10; i++) {
6            System.out.println("Java");
7        }
8        }
9  }
```

2) Beri penjelasan tentang hasil dari coding program

……
……
……
……
……

3) Modifikasi coding program pada point 1, sehingga hasil output coding program menghasilkan angka 1 sampai dengan 10, tuliskan listing program pada lembar yang tersedia!

```
run:
1 2 3 4 5 6 7 8 9 10 BUILD SUCCESSFUL (total time: 1 second)
```

……
……
……
……

……
……
……

4) Dengan menggunakan **for,** modifikasi program untuk membuat perulangan decrement (--) dengan menampilan angka 8 sampai dengan 4. Tuliskan listing programnya

```
run:
8 7 6 5 4 BUILD SUCCESSFUL (total time: 0 seconds)
```

……
……
……
……
……
……

Praktikum 2 (While)

1) Ketikkan listing program berikut

```
1   package perulangan;
2
3   public class Perulangan {
4       public static void main(String[] args) {
5        int i=0;
6       while (i<10) {
7         System.out.println(i);
8         i++;
9           }
10    }
11  }
```

2) Bagaimana hasilnya ? jelaskan output yang dihasilkan!

……
……

3) Modifikasi listing program dengan menggunakan while untuk menampilkan :

```
run:
8 7 6 5 4 BUILD SUCCESSFUL (total time: 0 seconds)
```

Tulis listing programnya!

……
……
……
……
……
……

4) Modifikasi listing program dengan menggunakan while untuk menampilkan:

```
run:
1 + 2 + 3 + 4 + 5 = 15
BUILD SUCCESSFUL (total time: 2 seconds)
```

Tuliskan listing programnya

……
……
……
……
……

Praktikum 3 (While)

Jalankan listing program di bawah ini:

```
int i=4;
while(i>0)
{
System.out.println(i);
i--;

}
```

1) Bagaimana hasilnya ?

……
……

2) Lakukan modifikasi program dengan menghapus **i-- ,** bagaimana hasilnya? Mengapa demikian? Lakukan analisis terhadap program !

……
……
……
……

Praktikum 4 (Do … While)

Jika ingin membuat perulangan angka 1 samapi dengan 4 dengan menggunakan do while

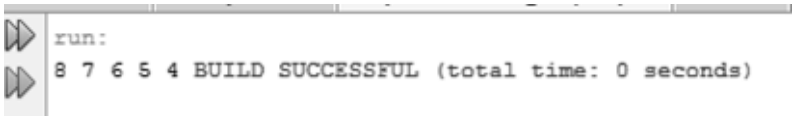
1) Ketikkan listing program berikut

```
1   package perulangan;
2
3   public class Perulangan {
4       public static void main(String[] args) {
5   int i=1;
6       do {
7         System.out.println(i);
8         i++;
9       } while (i < 5);
10    }
11  }
```

2) Bagaimana hasilnya ?

……
……

3) Modifikasi listing program dengan menggunakan do… while untuk menampilkan :



Tulis listing programnya!

……
……
……
……
……
……
……

TUGAS

1. Buatlah program untuk mencetak angka **genap** mulai dari 1 – 20.
2. Tulislah suatu program yang membaca sepuluh bilangan integer dan kemudian menampilan nilai terkecil dan terbesar dari sepuluh integer tersebut.

Tuliskan listing program pada lembar kerja berikut

……
……
……
……
……
……
……
……
……
……
……
……
……
……

6 Dasar dan Aturan PBO (Array)

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

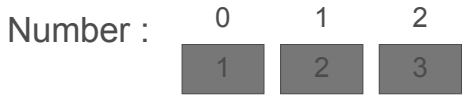
1. Memahami struktur kontrol array
2. Menggunakan struktur kontrol array satu dimensi dan multidimensi

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Pada bahasa pemrograman Java maupun dibahasa pemrograman yang lain, terdapat sebuah kemampuan untuk menggunakan satu variable yang dapat menyimpan beberapa data dan memanipulasinya dengan lebih efektif. Tipe variabel inilah yang disebut sebagai array.



Contoh dari Integer Array

Array adalah suatu wadah bentukan yang menyediakan penyimpanan sejumlah item yang bertipe sama. Array digunakan untuk mengelompokkan informasi yang berhubungan. Dalam Java, item dalam array selalu dinomori dari nol hingga nilai maksimum tertentu, yang nilainya ditentukan pada saat array tersebut dibuat.

Pendeklarasian Array

Array harus dideklarasikan seperti layaknya sebuah variabel. Pada saat mendeklarasikan array, Anda harus membuat sebuah daftar dari tipe data, yang diikuti oleh sepasang tanda kurung [], lalu diikuti oleh nama identifier-nya. Sebagai contoh,

```
int[]ages;
```

Atau Anda dapat menempatkan sepasang tanda kurung [] sesudah nama *identifier*. Sebagai

contoh,

```
Int ages[];
```

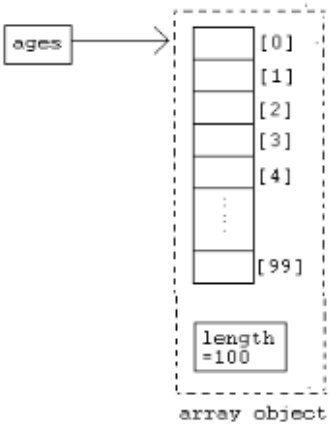
Setelah pendeklarasian array, kita harus membuat array dan menentukan berapa panjangnya dengan sebuah konstruktor. Proses ini di Java disebut sebagai *instantiation* (istilah dalam Java yang berarti membuat). Sebagai catatan bahwa ukuran dari array tidak dapat diubah setelah anda menginisialisasinya. Sebagai contoh,

Deklarasi Array

```
Int ages[];    //deklarasi
ages=new int[100];    //instantiate obyek
```

Atau bisa juga ditulis dengan,

```
//deklarasi dan instantiate obyek
Int ages[]=new int[100];
```



Pada contoh di samping, pendeklarasian tersebut akan memberitahukan kepada compiler Java, bahwa identifier ages akan digunakan sebagai nama array yang berisi data bertipe integer, dan dilanjutkan dengan membuat atau meng-*instantiate* sebuah array baru yang terdiri dari 100 elemen. Selain menggunakan sebuah pernyataan *new* untuk meng-instantiate array, Anda juga dapat mmendeklarasikan, membangun, kemudian memberikan sebuah nilai pada array sekaligus dalam sebuah pernyataan.

Contoh Listing Program

```
//Membuat sebuah array yang terdiri dari penginisialisasian
//4 variabel double bagi value{100,90,80,75}
double[]grades={100,90,80,75};
//Membuat sebuah array String dengan identifier days.Array
//ini terdiri dari 7 elemen.
String days[]={“Mon”,“Tue”,“Wed”,“Thu”,“Fri”,“Sat”,“Sun”};
```

b. Pengaksesan sebuah elemen array

Indeks adalah sebuah angka yang menyatakan *urutan* sebuah elemen pada suatu variabel array. Nomor indeks variabel array selalu dimulai dari 0 (nol), sehingga nomor indeks bagi elemen terakhir adalah sebesar (N-1), dimana N adalah jumlah total elemen. Untuk mengakses setiap elemen dalam variabel array cukup memanggil nomor indeksnya.

Gambar Elemen Array

Secara umum, deklarasi *array* dapat ditulis sebagai berikut :

```
type namavariabel[ ] ;
```

Atau

```
type[ ] namavariabel ;
```

Keterangan :

- type adalah mendeklarasikan tipe basis dari *array*. Dimana tipe basis ini menentukan data bagi masing-masing elemen yang membentuk *array*.
- namavariabel merupakan inisialisasi dari variabel yang akan didefenisikan.
- Untuk pembatas array maka diperlukan kurung siku [ ].

Sintaks Elemen Array

```
ages[0]=10;    //memberikan nilai 10 kepada elemen pertama array
System.out.print(ages[99]); //mencetak elemen array yang terakhir
```

Perlu diperhatikan bahwa sekali array dideklarasikan dan dikonstruksi, nilai yang disimpan dalam setiap anggota array akan diinisialisasi sebagai nol. Oleh karena itu, apabila Anda menggunakan tipe data seperti String, array tidak akan diinisalisasi menjadi string kosong. Untuk itu Anda tetap harus membuat String array secara eksplisit. Berikut ini adalah contoh kode untuk mencetak seluruh elemen di dalam array. Dalam contoh ini digunakanlah pernyataan *forloop*, sehingga kode kita menjadi lebih pendek.

```
Public class Array Sample
{
Public static void main (String[]args)
{
int[]ages=new int[100];
for(int i=0;i<100;i++)
{
System.out.print(ages[i]);
}
}
}
```

c. Array 1 Dimensi

Array 1 dimensi merupakan deklarasi array dengan satu variabel yang bertipe serupa. Untuk lebih memahami marilah Kita lihat contoh berikut:

```
class bulan{
        public static void main(String args[]){
//Deklarasi Variabel Array
int haribulan[];
//Penciptaan array index 12
haribulan = new int[12];
haribulan[0]  = 31;
haribulan[1]  = 29;
haribulan[2]  = 31;
haribulan[3]  = 30;
haribulan[4]  = 31;
haribulan[5]  = 30;
haribulan[6]  = 31;
haribulan[7]  = 31;
haribulan[8]  = 30;
haribulan[9]  = 31;
haribulan[10] = 30;
haribulan[11] = 31;

//Menampilkan Output
System.out.println("\nSeptember mempunyai "+haribulan[8]+" hari");
System.out.println("Februari mempunyai "+haribulan[1]+" hari\n");
}
}
```

Dari program di atas dapat Kita lihat bahwa arrray satu dimensi memiliki 12 index dengan masing- masing element diinisialisasikan satu persatu. Cara ini sering Kita jumpai dalam pemograman profesional. Dengan cara ini, array baru terbentuk ketika menggunakan *operator new()*. Di samping itu penciptaan atau inisialisasi nilai element pada array dapat juga langsung diberikan pada saat dideklarasikan. Proses ini hampir sama dengan inisialisasi variabel sederhana. Dimana nilai element array dikelompokkan ke dalam kurung kurawal dan dipisahkan dengan tanda koma. Hal ini akan secara otomatis menciptakan nilai element array sehingga *operator new()* tidak digunakan lagi. Untuk lebih memahami mari Kita lihat Contoh berikut

```
class bulan{
        public static void main(String args[]){

//Deklarasi Variabel Array dan sekligus pemberian nilai elemen

int haribulan[] = {31,29,31,30,31,30,31,31,30,31,30,31};

//Menampilkan Output
System.out.println("\nSeptember mempunyai "+haribulan[8]+" hari");
System.out.println("Februari mempunyai "+haribulan[1]+" hari\n");
}
}
```

d. Array Multidimensi

Array multidimensi diimplementasikan sebagai array yang terletak di dalam array, index array yang terdapat di dalam array. Pada beberapa kondisi diperlukan penulisan variabel

array yang menggunakan nomor indeks dua bilangan, misalnya pada aplikasi matrik. Data pada suatu matrik diketahui berdasarkan nilai baris dan kolomnya. Baris adalah sebuah bilangan dan kolom adalah sebuah bilangan juga. Pada dasarnya array 2 dimensi hampir sama dengan 1 dimensi. Hanya saja pada 2 dimensi terdapt indeks array di dalam array yang pertama. Di mana pada kelompok kurung siku yang pertama menyatakan elemen baris dan yang kedua menyatakan elemen kolom. Adapun bentuk umum array 2 dimensi sebagai berikut

type namavariabel[ ][ ];	*Atau*	*type[ ][ ] namavariabel ;*

Sebagai contoh,

Sintaks Array Multidimensi

```
Elemen512x128dariintegerarray
int[][]twoD = newint[512][128];
char[][][]threeD = new char[8][16][24];   //karakter array 8x16x24

//String array4 baris x 2 kolom
String[][]dogs= { {"terry","brown"}, {"Kristin","white"}, {"toby","gray"},
{"fido","black"}};
```

Untuk mengakses sebuah elemen di dalam array multidimensi, sama saja dengan mengakses array satu dimensi.

Sintaks Array Multidimensi

```
System.out.print(dogs[0][0]);
```

Array 1 Dimensi

Jika akan menampilan jumlah hari pada bulan tertentu dengan menggunakan array. Ketikkan listing program berikut:

```
1   package array;
2
3   public class Array {
4
5       public static void main(String[] args) {
6           // mendeklarasikan variabel bertipe array dengan tipe int
7       int [] jumlahHari;
8
9       // menentukan jumlah elemen array
10      jumlahHari = new int[12];
11
```

```
12          // mengisikan nilai dari setiap elemen array yang ada
13          jumlahHari[0] = 31;
14          jumlahHari[1] = 28;
15          jumlahHari[2] = 31;
16          jumlahHari[3] = 30;
17          jumlahHari[4] = 31;
18          jumlahHari[5] = 30;
19          jumlahHari[6] = 31;
20          jumlahHari[7] = 31;
21          jumlahHari[8] = 30;
22          jumlahHari[9] = 31;
23          jumlahHari[10] = 30;
24          jumlahHari[11] = 31;
25
26          // menampilkan salah satu elemen array
27          System.out.println("Bulan Maret memiliki " + jumlahHari[2] + " hari.");
28          }
29      }
```

1) Bagaimana hasilnya ? jelaskan output yang dihasilkan!

   ……
   ……
   ……
   ……

2) Jika pada baris ke **27** index pada variable jumlahHari diganti **[11] maka** output program yang dihasilkan adalah? Jelaskan alas an anda terkait output tersebut!

   ……
   ……
   ……
   ……

Array Multi Dimensi

Array multi dimensi 2 baris 2 kolom untuk menampilkan nama pemilik tas dan warna tas yang digunakan. Ketikkan listing program berikut:

```
1   package array2;
2
3   public class Array2 {
4
5       public static void main(String[] args) {
6          //Elcmcn512x128dariintcgcrarray
7                  int[][]twoD=new int[512][128];
8
9           //karakter array 8x16x24
10          char[][][]threeD=new char[8][16][24];
11
12          //String array4 baris x 2 kolom
13          String[][]bags= { {"terry","brown"}, {"Kristin","white"},{"toby","gray"},{"fido","black"} };
14
15            System.out.print(bags[0][1]);
16      }
17  }
```

1) Bagaimana hasilnya ? jelaskan output yang dihasilkan!

……
……
……

2) Jika ingin menampilkan warna tas yang dimiliki oleh toby, maka tuliskan listing programnya!

……
……
……
……
……
……

TUGAS

1. Buatlah program dengan array untuk mencetak angka genap mulai dari 1 – 20.
2. BUAT AGENDA HARIAN dengan menggunakan array, untuk menampilkan:
   Hari senin saya mengerjakan PR matematika
   Hari selasa saya mengantar ibu ke salon
   Hari rabu saya mencuci motor
   Hari sabtu saya mencuci baju

Tuliskan listing program pada lembar kerja berikut

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

7 Class dan Object

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami perbedaan class dan obyek
2. Menyajikan pembuatan class

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Perbedaan *Class* dan Obyek

Pada dunia perangkat lunak**, sebuah obyek adalah** sebuah komponen perangkat lunak yang stukturnya mirip dengan obyek pada dunia nyata. Setiap obyek dibangun dari sekumpulan data (atribut) yang disebut **variable** untuk menjabarkan karakteristik khusus dari obyek, dan juga terdiri dari **sekumpulan method** yang menjabarkan tingkah laku dari obyek. Bisa dikatakan bahwa **obyek adalah** sebuah perangkatlunak yang berisi sekumpulan variable dan method yg berhubungan.Variabel dan method dalam obyek Java secara formal diketahui sebagai **variabel instance** dan **method instance**.

Class adalah sturktur dasar dari OOP. ***Class* terdiri dari dua tipe** dari anggota dimana disebut dengan *field* (attribut/properti) dan method. ***Field*** merupakan tipe data yang didefinisikan oleh *class*, sementara ***method*** merupakan operasi. Sebuah obyek adalah sebuah *instance* (keturunan) dari *class*.

- ✓ ***Instansiasi Class***

  Untuk membuat sebuah obyek atau sebuah *instance* pada sebuah class. Kita menggunakan operator **new**. Sebagai contoh, jika anda ingin membuat *instance* dari *class string*, kita menggunakan kode berikut: String str2=new String(“Hello world!”);
  Ini juga sama dengan, String str2= "Hello";

✓ ***Variabel Class dan Variabel Method***

Selain dari variabel *instance*, kita juga memungkinkan untuk mendefinisikan variabel dari *class*, yang nantinya variabel ini dimiliki oleh *class*. Ini berarti variabel ini dapat memiliki nilai yang sama untuk semua obyek pada *class* yang sama. Mereka juga disebut *static member variables*.

1) Pembuatan *Class*

Sebelum menulis *class* Anda, pertama pertimbangkan dimana Anda akan menggunakan *class* dan bagaimana *class* tersebut akan digunakan. Pertimbangkan pula nama yang tepat dan tuliskan seluruh informasi atau *property* yang ingin Anda isi pada *class*. Jangan **sampai terlupa untuk menuliskan secara urut *method* yang akan** Anda gunakan dalam *class*.

Dalam pendefinisian *class*, dituliskan:

Sintaks Pembuatan Class

```
<modifier>class<name>
{
<attributeDeclaration>*
<constructorDeclaration>*
<methodDeclaration>*
}
```

Dimana :

<modifier> adalah sebuah *access modifier*, yang dapat dikombinasikan denganti pe*modifier* lain. Pada bagian ini, kita akan membuat sebuah *class* yang berisi *record* dari mahasiswa. Jika kita telah mengidentifikasikan tujuan dari pembuatan *class*, maka dapat dilakukan pemberian nama yang sesuai. Nama yang tepat pada *class* ini adalah StudentRecord.

Untuk mendefinisikan *class*, kita tuliskan:

```
Public class StudentRecord
{
//area penulisan kode selanjutnya
}
```

dimana,

Public	-	*Class* ini dapat di akses dari luar *package //modifier*
Class	-	*Keyword* yang digunakan untuk pembuatan *Class* dalam Java
StudentRecord	-	*Identifier yang menjelaskan class*

1) Deklarasi Atribut

Dalam pendeklarasian atribut,kita tuliskan:

modifier><type><name>[=<default_value>];

Langkah selanjutnya adalah mengurutkan atribut yang akan diisikan pada *class.* Untuk setiap informasi, urutkan juga tipe data yang yang tepat untuk digunakan.

✓ *Instance Variable*

Jika kita telah menuliskan seluruh atribut yang akan diisikan pada *class*, selanjut nya kita akan menuliskannya pada kode.Jika kita menginginkan bahwa atribut–atribut tersebut adalah unik untuk setiap *object* (dalam hal ini untuk setiap mahasiswa), maka kita harus mendeklarasikannya sebagai *instance variable*

Sintaks Deklarasi Atribut

```
Public class StudentRecord
{
Private String name;
Private String address;
Private int age;
Private double mathGrade; private double englishGrade;
private double scienceGrade; private double average;
}
```

Private disini menjelaskan bahwa variabel tersebut hanya dapat diakses oleh *class* itu sendiri. *Object* lain tidak dapat menggunakan variabel tersebut secara langsung. Kita akan membahas tentang kemampuan akses pada pembahasan selanjutnya.

✓ *Class Variable atau StaticVariables*

Kita dapat mendeklarasikan *class variable* atau variabel yang dimiliki *class* sepenuhnya. Nilai pada variabel ini sama pada semua *object* di *class* yang sama. Anggaplah kita menginginkan jumlah dari mahasiswa yang dimiliki dari seluruh *class*, kita dapat mendeklarasikan satu *staticvariable* yang akan menampung nilai tersebut. Kita beri nama variabel tersebut dengan nama studentCount.

Berikut penulisan *staticvariable:*

Sintaks Class Variable

```
Public class StudentRecord
{  //area deklarasi instance variables
Private static int student Count;
//area penulisan kode selanjutnya
}
```

Kita gunakan *keyword : 'static'* untuk mendeklarasikan bahwa variabel tersebut adalah *static*. Maka keseluruhan kode yang dibuat terlihat sebagai berikut:

```
Public class StudentRecord
{
  private String name;
  private int age;
  Private double  average;
  //area penulisan kode selanjutnya
}
```

PRAKTIKUM

Latihan Class : Membuat class kotak (Simpan dengan nama **class**)

```
1   package pkgclass;
2
3   class Kotak{ //buat sendiri
4   double panjang;
5     double lebar;
6     double tinggi;
7   }
8
9   public class Class {
10      public static void main(String[] args) {
11          // TODO code application logic here
12      double volume1, volume2;
13
14      Kotak k1 = new Kotak();  // mendeklarasikan objek k1
15      Kotak k2 = new Kotak();  // mendeklarasikan objek k2
16
17      // Mengisikan nilai ke dalam objek k1
18      k1.panjang = 4;
19      k1.lebar = 3;
20      k1.tinggi = 2;
21
22      // Mengisikan nilai ke dalam objek k2
23      k2.panjang = 6;
24      k2.lebar = 5;
25      k2.tinggi = 4;
26
27          // Menghitung isi/volume dari objek k1
28      volume1 = k1.panjang * k1.tinggi * k1.lebar;
29
30     // Menghitung isi/volume dari objek k2
31     volume2 = k2.panjang * k2.tinggi * k2.lebar;
32
33     // Menampilkan nilai volume k1 dan k2 ke layar monitor
34     System.out.println("Volume k1 = " + volume1);
35     System.out.println("Volume k2 = " + volume2);
36   }
37     }
```

1) Bagaimana hasilnya ? jelaskan output yang dihasilkan!

……

……

……

Latihan 2. Mendefinisikan Method,

Ketikkan listing program di bawa ini dan simpan dengan nama **DemoMethod**

```
class Kotak {
  double panjang;
  double lebar;
  double tinggi;


  // Mendefinisikan method void (tidak mengembalikan nilai)
  void cetakVolume() {
    System.out.println("Volume kotak = " +   (panjang * lebar * tinggi));
  }
}

class DemoMethod {
  public static void main(String[] args) {
    Kotak k1, k2, k3;

    // instansiasi objek
    k1 = new Kotak();
    k2 = new Kotak();
    k3 = new Kotak();

    // mengisi data untuk objek k1
    k1.panjang = 4;
    k1.lebar = 3;
    k1.tinggi = 2;

    // mengisi data untuk objek k2
    k2.panjang = 6;
    k2.lebar = 5;
    k2.tinggi = 4;

    // mengisi data untuk objek k3
    k3.panjang = 8;
    k3.lebar = 7;
    k3.tinggi = 6;

    // memanggil method cetakVolume() untuk masing-masing objek
    k1.cetakVolume();
    k2.cetakVolume();
    k3.cetakVolume();
  }
```

Bagaimana hasilnya?

……
……

TUGAS

1. Buatlah listing program dengan kelas Mobil dan obyeknya Pajero. (capture listing program dan hasil)
2. Buatlah listing program untuk menampilkan behavior macan tutul (obyek) dari kelas binatang buas. (. (capture listing program dan hasil)

Tuliskan listing program pada lembar kerja berikut

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

8 Encapsulasi

A. TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami konsep enkapsulasi
2. Menerapkan konsep enkapsulasi dalam *class*

B. PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

C. TEORI

1) Enkapsulasi dan *modifier*

Enkapsulasi merupakan teknik yang membuat variabel/*field class* menjadi bersifat *private* dan menyediakan akses ke variabel/*field* melalui *public method*. Jika *field* di deklarasikan sebagai *private*, maka *field* ini tidak bisa diakses oleh siapapun diluar *class*, dengan demikian *field* disembunyikan di dalam *class*.

Manfaat utama teknik enkapsulasi adalah kita mampu memodifikasi kode tanpa merusak kode yang telah digunakan pada *class* lain. Enkapsulasi memiliki manfaat sebagai berikut:

- ✓ Modularitas

  Source code dari sebuah *class* dapat dikelola secara independen dari *source code class* yang lain. Perubahan internal pada sebuah *class* tidak akan berpengaruh bagi *class* yang menggunakannya.
- ✓ *Information Hiding*

  Penyembunyian informasi yang tidak perlu diketahui objek lain.

jika Anda ingin beberapa atribut hanya dapat diubah hanya dengan *method* tertentu, tentu Anda ingin menyembunyikannya dari obyek lain pada *class*. Di Java, implementasi tersebut disebut dengan *access modifiers*.

2) Penerapan enkapsulasi dalam *class*

Kita dapat menyembunyikan information dari suatu *class* sehingga anggota-anggota

class tersebut tidak dapat diakses dari luar. Adapun caranya adalah cukup dengan memberikan akses *control private* ketika mendeklarasikan suatu atribut atau *method*. Contoh:

private int nip;

Encapsulation (Enkapsulasi) adalah suatu cara untuk menyembunyikan implementasi detail dari suatu class. Enkapsulasi mempunyai dua hal mendasar, yaitu:

- ✓ *information hiding*
- ✓ menyediakan suatu perantara (*method*) untuk pengaksesan data

Contoh:

```
public class Siswa {
private int nip;
public void setNip(int n) {
nip=n; } }
```

Contructor (konstruktor) adalah suatu *method* yang pertama kali dijalankan pada saat pembuatan suatu obyek. Konstruktor mempunyai ciri yaitu:

- ✓ mempunyai nama yang sama dengan nama class,
- ✓ tidak mempunyai return type (seperti void, int, double, dan lain-lain).

Contoh:

```
public class Siswa {
private int nrp; private String nama;

public Siswa(int n, String m) {
nrp=n;    nama=m;
}
}
```

Suatu *class* dapat mempunyai lebih dari 1 konstruktor dengan syarat daftar parameternya tidak boleh ada yang sama.

Contoh:

```
public class Siswa {
private int nrp;
private String nama;
public Siswa(String m) {
nrp=0;
nama="";
}
public Siswa(int n, String m) {
nrp=n;    nama=m;
}  }
```

Terdapat 4 macam *access modifiers* di JAVA, yaitu : *public*, *private*, *protected* dan *default*. 3 tipe akses pertama tertulis secara ekplisit pada kode untuk mengindikasikan tipe akses,

sedangkan yang keempat yang merupakan tipe default, tidak diperlukan penulisan *keyword* atas tipe.

No	Modifier	Class	Paket Sama		Paket Berbeda	
			Extend	Instan	Extend	Instan
1	Public	√	√	√	√	√
2	Protected	√	√	√	√	
3	Default	√	√	√		
4	Private	√				

PRAKTIKUM

✓ ***Public***

Dapat dilihat pada table diatas bahwa keyword Public dapat diakses didalam class itu sendiri, dapat diakses dengan menggunakan metode *extend* dan instan pada paket yang sama, serta dapat diakses dengan metode *extend* maupun instan dalam paket yang berbeda. Artinya hak akses *public* dapat diakses oleh sembarang object manapun dan dimanapun posisinya serta dengan apapun caranya. Data maupun *method* yang bersifat *public* dapat diakses oleh semua bagian didalam program. Untuk mendeklarasikan suatu data atau method dengan tingkat akses *public*, gunakan kata kunci *public*.

Berikut contoh program sederhana : buat file baru dengan nama **encapsulasi**

Listing Program Praktikum 1

```
1   package encapsulasi1;
2
3    class atas
4   {
5   public int a;
6   protected int b;
7   private int c;
8   }
9
10  public class Encapsulasi1 {
11
12  public static void main(String[] args) {
13          // TODO code application logic here
14      atas objek = new atas();
15      objek.a=2;
16      objek.b=3;
17
18  System.out.println("nilai a:" + objek.a);
19  System.out.println("nilai a:" + objek.b);
20
21      }
22  }
```

Bagaimana hasil dari listing program tersebut?

……
……

program diatas terdiri dari dua kelas yaitu kelas sekunder yang berisi variabel a, b dan c dengan tingkat akses yang berbeda, dan kelas primer yang berisi objek untuk melakukan *instance* pada kelas turunan, objek pada kelas primer hanya dapat mengisi nilai pada variabel a dan b karena kedua variabel tersebut memiliki tingkat akses *public* dan *protected*, karena variabel c memiliki tingkat akses *private* maka obyek pada kelas primer tidak bisa mengisi variabel tersebut.

- ✓ ***Protected***

  Suatu data maupun *method* yang dideklarasikan dengan tingkat akses *protected* dapat diakses oleh kelas yang memilkinya dan juga oleh kelas–kelas yang masih memiliki oleh hubungan turunan. Sebagai contoh, apabila data x dalam kelas A dideklarasikan sebagai *protected*, maka kelas B (yang merupakan turunan dari kelas A) diizinkan untuk mengakses data x. Namun apabila terdapat kelas lain, misalnya C (yang bukan merupakan turunan dari kelas A maupun B), tetap tidak dapat mengakses data – data yang dideklarasikan dengan tingkat akses *protected*. Untuk mendeklarasikan suatu data atau *method* dengan tingkat akses *protected*, gunakan kata kunci *ptotected*.

- ✓ ***Private***

  Dengan mendeklarasikan data dan method menggunakan tingkat akses private, maka data dan *method* tersebut hanya dapat diakses oleh kelas yang memilikinya saja. Ini berarti data dan *method* tersebut tidak boleh diakses atau digunakan oleh kelas-kelas lain yang terdapat didalam program. Untuk mendeklarasikan suatu data atau *method* dengan tingkat akses *private*, gunakan kata kunci *private*.

```
public class Siswa
{
  private String nama; //akses dasar terhadap variabel
private String getNama()    //akses dasar terhadap metode
{
        return name;
    }
}
```

Pada contoh diatas, variabel nama dan *method* getNama() hanya dapat diakses oleh *method internal class* tersebut.

Berikut contoh program sederhana : buat file baru dengan nama **encapsulasi**

Listing Program Praktikum 2

```
1   package encapsulasi1;
2
3    class atas
4   {
5   public int a;
6   protected int b;
7   private int c;
8   }
9
10  public class Encapsulasi1 {
11
12  public static void main(String[] args) {
13          // TODO code application logic here
14      atas objek = new atas();
15      objek.a=2;
16      objek.b=3;
17
18  System.out.println("nilai a:" + objek.a);
19  System.out.println("nilai a:" + objek.b);
20
21      }
22  }
```

Jalankan listing program tersebut, bagaimana hasilnya?

……

……

Pada baris ke 17, tambahkan script program untuk memanggil "c" **objek.c =5;**

Bagaimana hasilnya? Mengapa demikian? **Jelaskan!**

……

……

……

……

……

- ✓ ***Default***

Untuk hak akses *default*, sebenarnya hanya ditujukan untuk *class* yang ada dalam satu paket, atau istilahnya hak akses yang berlaku untuk satu folder saja (tidak berlaku untuk *class* yang tidak satu folder/package).

```
public class Siswa{
String nama;       //akses dasar terhadap variabel
String getName(){        //akses dasar terhadap method
   return nama;
  }   }
```

Pada contoh diatas, variabel nama dan *method* getNama() hanya dapat diakses oleh *method internal class* tersebut.

b. Praktikum 3

Modifikasi listing program pada praktikum 2, tambahkan variable alamat dengan type data String.

```
package encapsulasi1;

 class atas
{
public int a;
protected int b;
private int c;
private String alamat;

}
```

Tambahkan acessor method dan mutator method

```
package encapsulasi1;

 class atas
{
public int a;
protected int b;
private String alamat;

public String getAlamat(){
    return alamat;

}

public void setAlamat (String tempString){
alamat =tempString;
}
}
```

Panggil di program utama

```
public class Encapsulasi1 {
public static void main(String[] args) {
        // TODO code application logic here
    atas objek = new atas();
    objek.a=2;
    objek.b=3;
    objek.setAlamat("Malang");

System.out.println("nilai a:" + objek.a);
System.out.println("nilai a:" + objek.b);
System.out.println("Alamat:" + objek.getAlamat());

    }
}
```

Bagaimana cara mengakses atribut tersebut?

……
……
……

Apa fungsi acessor method dan mutator method?

……
……
……

Bagaimana hasilnya? Simpulkan!

……
……
……

TUGAS

1. Apa yang anda pahami terkait *keyword private*, *protected*, *public*?
2. Apa yang terjadi jika anda membuat sebuah *property* atau *method* menjadi *private*, *protected*, *public* ?

Buatlah program untuk menghitung gaji bersih dari seorang pegawai, pajak ppn sebesar 10% dari gaji kotor.

Tuliskan jawaban dan listing program pada lembar kerja berikut

……


Java

9 INHERRITANCE (PEWARISAN)

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami konsep pewarisan
2. Menciptakan superclass dan subclass
3. Memahami penggunaan kata kunci super
4. Menerapkan penggunaan kata kunci cuper dalam inheritas
5. Memahami konsep overloading dan overriding

PERALATAN PRAKTIKUM

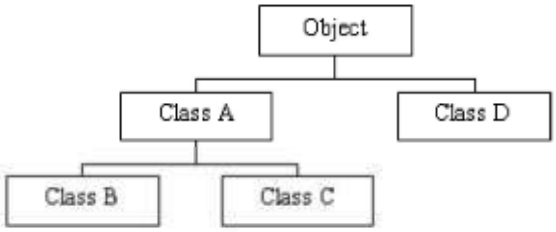
1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Konsep Inheritas

Dengan *inheritance,* sebuah class dapat mempunyai class turunan.Suatu class yang mempunyai class turunan dinamakan parent class atau base class. Sedangkan class turunan itu sendiri seringkali disebut subclass atau child class. Suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class. Karena suatu subclass dapat mewarisi apa apa yang dipunyai oleh parent class-nya, maka member dari suatu subclass adalah terdiri dari apa-apa yang ia punyai dan juga apa-apa yang ia warisi dari class parent-nya.

Dalam Java, semua class, termasuk class yang membangun Java API, adalah subclasses dari superclass Object. Contoh hirarki class diperlihatkan di bawah ini. Beberapa class di atas class utama dalam hirarki class dikenal sebagai superclass. Sementara beberapa class di bawah class pokok dalam hirarki class dikenal sebagai sub class dari class tersebut.



Class hierarchy in Java.

Pewarisan adalah keuntungan besar dalam pemrograman berbasis object karena suatu sifat atau method didefinisikan dalam ***superclass*****,** sifat ini secara otomatis diwariskan dari semua ***subclasses***. Jadi, Anda dapat menuliskan kode method hanya sekali dan mereka dapat digunakan oleh semua subclass. Subclass hanya perlu mengimplementasikan perbedaannya sendiri dan induknya.

Suatu class yang mempunyai class turunan dinamakan parent class atau base class. Sedangkan class turunan itu sendiri sering kali disebut subclass atau child class. Suatu subclass dapat Mewarisi siapa-apa yang dipunyai oleh parent class-nya, sehingga member dari suatu subclass adalah terdiri dari apa-apa yang ia punyai dan juga apa-apa yang diawarisi dari classparent-nya. Kesimpulannya, boleh dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (**extend**) parentclass-nya.

Mendefinisikan Superclass dan Subclass

Untuk memperoleh suatu class, kita menggunakan kata kunci **extend** yang digunakan untuk melakukan proses penurunan terhadap suatu kelas. Bentuk umum dari penggunaan kata kunci tersebut adalah sebagai berikut:

```
Class nama_subclass extends nama_superclass{
//badan class
}
```

✓ Kapan menerapkan inheritas?

Kita baru perlu menerapkan inheritance pada saat kita jumpai ada suatu class yang dapat diperluas dari class lain.

Misal terdapat class Pegawai public class Pegawai
{public String nama;public double gaji;}

Misal terdapat class Manager public class Manajer
{public String nama;public double gaji;public String departemen;}

Dari 2 buah class diatas, kita lihat class Manajer mempunyai data member yang identik sama dengan class Pegawai, hanya saja ada tambahan data member departemen.Sebenarnya yang terjadi disana adalah class Manajer merupakan perluasan dari class Pegawai dengan tambahan data member departemen.

Disini perlu memakai konsep inheritance, sehingga class Manajer dapat kita tuliskan seperti berikut :

*public class Manajer **extends** Pegawai {public String departemen;}*

- ✓ Keuntungan inheritas
  - ➢ **Subclass** menyediakan state/behaviour yang spesifik yang membedakannya dengan **superclass**, hal ini akan memungkinkan programmer Java untuk menggunakan ulang source code dari superclass yang telah ada.
  - ➢ Programmer Java dapat mendefinisikan superclass khusus yang bersifat generik, yang disebut abstract class, untuk mendefinisikan class dengan behaviour dan state secara umum.
- ✓ Deklarasi inheritas

Di dalam Java untuk mendeklarasikan suatu class sebagai sub class dilakukan dengan cara menambahkan kata kunci **extends** setelah deklarasi nama class, kemudian diikuti dengan nama parentclass-nya. Kata kunci extends tersebut memberitahu compiler Java bahwa kita ingin melakukan perluasan class.

Berikut adalah contoh deklarasi inheritance:

```
public class B extends A{
.....
}
```

Contoh di atas memberitahukan compiler Java bahwa kita ingin meng-extend class A ke class B. Dengan kata lain, class B adalah subclass (class turunan) dari class A, sedangkan class A adalah parent class dari class B.

Java hanya memperkenankan adanya single inheritance. Konsep single inheritance hanya memperbolehkan suatu sublass mempunyai satu parent class. Dengan konsep single inheritance ini, masalah pewarisan akan dapat diamati dengan mudah.

Dalam konsep dasar inheritance dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parentclass-nya. Contoh:

```
Public class Pegawai{
Public String nama;
Public double gaji;
        }
Public class Manajer extends Pegawai{
Public String departemen;
        }
```

Pada saat class Manajer menurunkan atau memperluas (extend) class Pegawai, maka ia mewarisi data member yang dipunyai oleh class Pegawai. Dengan demikian, class

Manajer mempunyai data member yang diwarisi oleh Pegawai (nama, gaji), ditambah dengan data member yang ia punyai (departemen).

1. **Kata Kunci Super**

   Constructor yang terdapat pada kelas induk dapat dipanggil dari kelas turunannya menggunakan kata kunci ***super***. Bentuk umum pemanggilannya adalah :

   Super (daftar-parameter);

   Daftar-parameter adalah daftar parameter yang didefinisikan pada constructor kelas induk.

   Ada beberapa hal yang harus diingat ketika menggunakan pemanggil constuktor super:

   a. Pemanggil super() **harus dijadikan pernyataan pertama dalam** constructor.
   b. Pemanggil super() hanya dapat digunakan dalam definisi constructor.
   c. Termasuk constructor this() dan pemanggil super() **tidak boleh terjadi dalam** constructor **yang sama**.

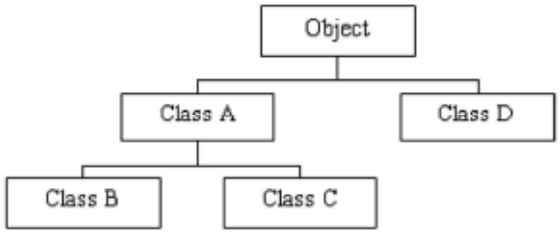
   Pemakaian lain dari super adalah untuk menunjuk anggota dari superclass (seperti reference **this**). Sebagai contoh,

```
public Student()
{
super.name = “somename”;
super.address = “some address”;
}
```

Untuk lebih memperjelas pembahasan, berikut contoh program yang menunjukkan penggunaan kata kunci *super* dalam pemanggilan *constructor* kelas induk

PRAKTIKUM

Praktikum 1. Perhatikuan ilustrasi berikut : class B turunan dari class A



```
Source  History
 1  package turunan;
 2  class A {
 3    private int a;
 4
 5    public void setA(int nilai) {
 6      a = nilai;
 7    }
 8
 9    public int getA() {
10      return a;
11    }
12  }
13
14  class B extends A {  // membuat kelas turunan (subclass) dari kelas A
15    private int b;
16
17    public void setB(int nilai) {
18      b = nilai;
19    }
20
21    public int getB() {
22      return b;
23    }
24  }
25
26  public class Turunan {
27
28      /**
29       * @param args the command line arguments
30       */
31      public static void main(String[] args) {
32          // TODO code application logic here
33          B obj = new B();      // melakukan instansiasi terhadap kelas B
34
35      obj.setA(100);     // mengeset nilai objek dari kelas B
36      obj.setB(200);
37
38      // mendapatkan nilai yang terdapat dalam objek dari kelas B
39      System.out.println("Nilai a : " + obj.getA());
40      System.out.println("Nilai b : " + obj.getB());
41
42      }
43  }
```

Bagaimana hasilnya?

……
……
……

Lengkapi listing program dengan menambahkan class C merupakan turunan dari kelas A. tulis listing programnya!

……
……
……
……

.....

.....

.....

Praktikum 2. Class Balok turunan dari class persegipanjang

```
Source | History
 1  package pewarisan;
 2  class Persegipanjang {
 3    protected double panjang;
 4    protected double lebar;
 5
 6     Persegipanjang() {   //default constructor
 7      panjang = lebar = 0;
 8    }
 9
10    Persegipanjang(int p, int l) {
11      panjang = p;
12      lebar = l;
13       }
14
15    public double hitungLuas() {
16      return (panjang * lebar );
17    }
18  }
19
20  class Balok extends Persegipanjang {
21    private double tinggi;
22
23    Balok(int p, int l, int t) {
24
25      super(p, l);  // memanggil constructor kelas Kotak
26      tinggi = t;
27    }
28
29    public double getTinggi() {
30      return tinggi;
31    }
32  }
33
34  public class Pewarisan {
35      public static void main(String[] args) {
36          Balok k=new Balok(6,5,4);
37          System.out.println("Luas Persegi Panjang : " + k.hitungLuas());
38        System.out.println("tinggi balok  : " + k.getTinggi());
39        System.out.println("Volume balok : " + (k.hitungLuas()*k.getTinggi()));
40
41          // TODO code application logic here
42      }
43  }
```

Bagaimana hasilnya?

.....

.....

.....

.....

.....

2. konsep overloading dan overriding

1) Metode Overloading

Overloading adalah suatu keadaan dimana beberapa method sekaligus dapat mempunyai nama yang sama, akan tetapi mempunyai fungsionalitas yang berbeda. Overloading ini dapat terjadi pada class yang sama atau pada suatu parent class dan subclass-nya. Overloading mempunyai ciri-ciri sebagai berikut:

- ✓ Nama method harus sama
- ✓ Daftar parameter harus berbeda
- ✓ Return type boleh sama, juga boleh berbeda

Contoh penggunaan overloading dilihat di bawah ini:

```
Gambar(intt1)              →     1parametertitik,untuk menggambartitik
Gambar(intt1, int t2)      →     2parametertitik,untukmenggambargaris
Gambar(intt1, int t2, int  →     3parametertitik,untukmenggambarsegitiga
t3)
Gambar(intt1, int t2, int  →     3parametertitik,untukmenggambarsegiempat
t3, int t4)
```

2) Overriding Method

Overriding adalah suatu keadaan dimana method pada subclass menolak method pada parent class-nya. Overriding mempunyai ciri-ciri sebagai berikut:

- ✓ Nama method harus sama
- ✓ Daftar parameter harus sama
- ✓ Return type harus sama

Untuk beberapa pertimbangan, terkadang class asal perlu mempunyai implementasi berbeda dari method yang khusus dari *superclass* tersebut. Oleh karena itulah, method overriding digunakan. *Subclass* dapat mengesampingkan method yang didefinisikan dalam *superclass* dengan menyediakan implementasi baru dari method tersebut. Misalnya kita mempunyai implementasi berikut untuk method getName dalam superclass Person,

```
public class Person
{
public String getName()
{
System.out.println("Parent: getName");
return name;
}
}
```

Untuk override, method getName dalam subclass Student, kita tulis,

```
public class Student extends Person
{
public String getName()
{
System.out.println("Student: getName");
return name;
}
}
```

Jadi,ketika kita meminta method getName dari object class Student, method overridde akan dipanggil, keluarannya akan menjadi, *Student: getName*

3) Method final dan classfinal

Dalam Java, juga memungkinkan untuk mendeklarasikan class-class yang tidak lama menjadi subclass. Class ini dinamakan **class final**. Untuk mendeklarasikan class untuk menjadi final kita hanya menambahkan kata kunci **final** dalam deklarasi class. Sebagai contoh, jika kita ingin class Person untuk dideklarasikan final, kita tulis,

```
public final class Person
{
//area kode
}
```

Beberapa class dalam Java API dideklarasikan secara final untuk memastikan sifatnya tidak dapat di-*override*. Contoh-contoh dari class ini adalah Integer , Double , dan String. Ini memungkinkan dalam Java membuat method yang tidak dapat di-*override.* Method ini dapat kita panggil **method final**. Untuk mendeklarasikan method untuk menjadi final,kita tambahkan kata kunci final kedalam deklarasi method. Contohnya, jika kita ingin method getName dalam class Person untuk dideklarasikan final, kita tulis,

```
public final String getName(){
return name;
}
```

Method static juga secara otomatis final.Ini artinya Anda tidak dapat membuatnya override.

TUGAS

1. Buatlah program untuk menampilkan luas segitiga dengan superclass bangun_datar dan sub class Segitiga. Gunakan prinsip overriding dan atau overloading.
2. Buatlah program untuk menampilkan luas permukaan dan volume tabung. Gunakan parent-class Luas Lingkaran (method: jari-jari).

. Tuliskan listing program pada lembar kerja berikut

10 Polimorphism

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

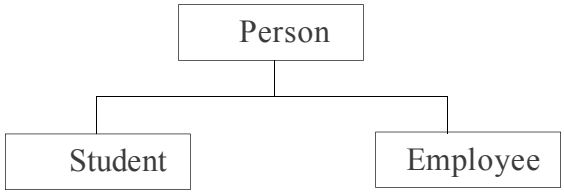
1. Memahami konsep polimorfisme
2. Menyajikan overloading dan overriding dalam class

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Class induk Person dan subclass Student dari contoh sebelumnya, kita tambahkan subclass lain dari Person yaitu Employee. Di bawah ini adalah hierarkinya,



Gambar Hirarki dari class induk Person

Dalam Java, kita dapat membuat referensi yang merupakan tipe dari superclass ke sebuah object dari subclass tersebut. Sebagai contohnya,

```
        public static main( String[] args )
        {
        Person      ref;
        Student     studentObject = new Student();
        Employee employeeObject = new Employee();

        ref = studentObject; //Person menunjuk kepada
        // object Student
//beberapa kode di sini
        }
```

Sekarang dimisalkan kita punya method getName dalam superclass Person kita, dan

kita override method ini dalam kedua subclasses Student dan Employee,

```
        public class Person {
public String getName(){
System.out.println("Person Name:" + name);
return name;
        }
        }

        public class Student extends Perso {
public String getName(){
System.out.println("Student Name:" + name); return name;
        }
        }

          public class Employee extends Person   {
              public String getName(){
              System.out.println("Employee Name:" + name);
 return name;
                    }
```

Kembali ke method utama kita, ketika kita mencoba memanggil method getName dari reference Person ref, method getName dari object Student akan dipanggil. Sekarang, jika kita berikan ref ke object Employee, method getName dari Employee akan dipanggil.

```
public static main( String[] args )
{
Person ref;
Student   studentObject = new Student();
Employee     employeeObject = new Employee();

ref = studentObject; //Person menunjuk kepada
String temp = ref.getName();       //getName dari Student
System.out.println( temp );                //class dipanggil

ref = employeeObject; //Person menunjuk kepada
// object Employee

String temp = ref.getName(); //getName dari Employee
System.out.println( temp );       //class dipanggil

}
```

Kemampuan dari reference untuk mengubah sifat menurut object apa yang dijadikan acuan dinamakan polimorfisme. Polimorfisme menyediakan multiobject dari subclasses yang berbeda untuk diperlakukan sebagai object dari superclass tunggal, secara otomatis menunjuk method yang tepat untuk menggunakannya ke *particular* object berdasar subclass yang termasuk di dalamnya.

Contoh lain yang menunjukkan properti polimorfisme adalah ketika kita mencoba melalui reference ke method. Misalkan kita punya method static **printInformation** yang mengakibatkan object Person sebagai reference, kita dapat me-reference dari tipe Employee dan tipe Student ke method ini selama itu masih subclass dari class Person.

```
public static main( String[] args )
{
Student     studentObject = new Student();
Employee    employeeObject = new Employee();

printInformation( studentObject );
printInformation( employeeObject );
}
public static printInformation( Person p ){
. . . .
}
```

PRAKTIKUM

```
class EkspresiWajah{
    public String respons(){
    return ("Lihatlah reaksi wajah saya");
        }
}

class Gembira extends EkspresiWajah{
   publuc String respons(){
   return ("Ha..ha..ha... saya sedang gembira");
   }
}

class Sedih extends EkspresiWajah{
    publuc String respons(){
    return ("Hiks..hiks..hiks... tega sekali kau..");
    }
}

class Marah extends EkspresiWajah{
    publuc String respons(){
    return ("Hei!... jangan dekati saya!");
    }
}
```

```
public class EkspresiEmosi {
    public static void main (String [] srgs){

    EkspresiWajah Oekspresi = new EkspresiWajah();
    Gembira Ogembira = new Gembira();
    Sedih Osedih = new Sedih();
    Marah Omarah = new Marah();

    EkspresiWajah [] ekspresi = new EkspresiWajah[4];
    ekspresi[0] = Oekspresi;
    ekspresi[1] = OGembira;
    ekspresi[2] = OSedih;
    ekspresi[3] = OMarah;

    System.out.println("Ekspresi : " + ekspresi[0].respons());
    System.out.println("Ekspresi Pertama : " + ekspresi[1].respons());
    System.out.println("Ekspresi Kedua : " + ekspresi[2].respons());
    System.out.println("Ekspresi Ketiga : " + ekspresi[3].respons());
    }
}
```

Bagaimana output dari program tersebut?

……
……
……
……
……
……

Jelaskan jalan programnya sesuai dengan konsep polimorfisme!

……
……
……
……

TUGAS

Buatlah program yang mengaplikasikan abstract class, interface, dan polimorphism

11 Exception Handling (Penanganan Eksepsi)

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami konsep, tipe, dan cara penanganan eksepsi.
2. Memahami cara melontar dan menangkap eksepsi.
3. Memahami konsep try catch

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Eksepsi (exeption) adalah suatu mekanisme yang digunakan oleh beberapa bahasa pemrograman untuk mendeskripsikan apa yang harus dilakukan jika ada suatu kondisi yang tidak diinginkan terjadi. Eksepsi adalah keadaan tidak normal yang muncul pada suatu bagian program pada saat dijalankan. Penanganan eksepsi pada java membawa pengelolaan kesalahan program saat dijalankan kedalam orientasi-objek. Eksepsi java adalah objek yang menjelaskan suatu keadaan eksepsi yang muncul pada suatu bagian program.

Eksepsi dapat dijumpai saat:

a. Mengakses method dengan argumen yang tidak sesuai.
b. Membuka file yang tidak ada.
c. Koneksi jaringan yang terganggu.
d. Manipulasi operand yang nilainya keluar dari batasan yang didefinisikan.
e. Pemanggilan class yang tidak ada.

Eksepsi dapat muncul tidak beraturan dalam suatu method, atau dapat juga dibuat secara manual dan nantinya melaporkan sejumlah keadaan kesalahan ke method yang memanggil.

1. Dasar-dasar penanganan Eksepsi

Penanganan eksepsi pada java diatur dengan lima kata kunci : **try, Catch, throw,**

throws dan finally. Pada dasarnya try digunakan untuk mengeksekusi suatu bagian program, dan jika muncul kesalahan, sistem akan melakukan throw suatu eksepsi yang dapat anda catch berdasarkan tipe eksepsinya, atau yang anda berikan finally dengan penangan default.

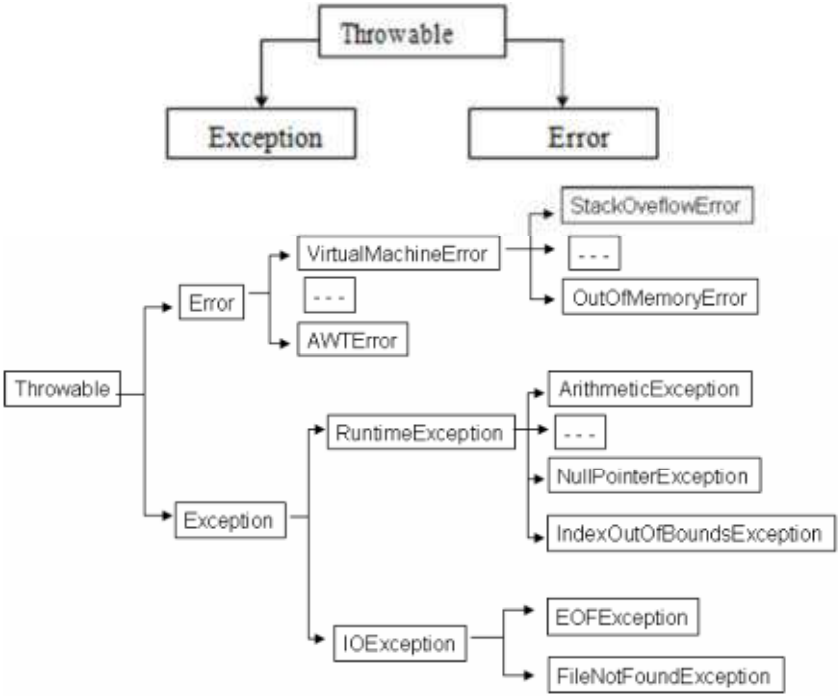
Berikut ini bentuk dasar bagian penanganan eksepsi :

```
try {
// Block of Code
}
catch (ExceptionType1 e) {     // Exception Handler for ExceptionType1
}
catch (ExceptionType2 e) {      // Exception Handler for
ExceptionType2 throw (e); // re-throw the Exception
}
finally { }
```

2. Tipe Eksepsi

Dipuncak hirarki class eksepsi terdapat satu class yang disebut throwable. Class ini digunakan untuk merepresentasikan semua keadaan eksepsi. Setiap *ExceptionType* pada bentuk umum diatas adalah subclass dari *throwable.*

Dua subclass langsung throwable didefinisikan untuk membagi class throwable menjadi dua cabang yang berbeda. Satu, class Exception, digunakan untuk keadaan eksepsi yang harus ditangkap oleh program yang kita buat. Cabang kedua throwable adalah class error, yang mendefinisikan keadaan yang tidak diharapkan untuk ditangkap dalam lingkungan normal.



Dalam hal ini class ***java.lang.Throwable*** merupakan class parent dari seluruh objek yang bisa dilempar dan ditangkap menggunakan mekanisme exception handling.

Sehingga dalam contoh di atas pernyataan ***catch (ArrayIndexOutOfBoundsException e)*** merupakan jenis *RuntimeException.* Jika tidak hafal jenis eksepsinya secara pasti, bisa mengambil parent classnya. Eksepsi Umum Java menyediakan beberapa eksepsi yang telah didefinisikan.

Beberapa eksepsi yang umum diantaranya adalah :

a. ***ArithmeticException*** hasil dari operasi divide-by-zero terhadap integer. Contoh : int i = 12 / 0;
b. ***NullPointerException*** mengakses membernya (atribut atau method) ketika reference-nya masih menunjuk ke null, misalnya ketika belum dibuat obyek instannya. Contoh : Date d = null; //tanpa membuat instance object System.out.println(d.toString());
c. ***NegativeArraySizeException*** membuat array dengan size yang diset negatif.
d. ***ArrayIndexOutOfBoundsException*** mengakses array melebihi indeks terbesarnya.
e. ***SecurityException*** biasanya terjadi pada sebuah browser, ketika class SecurityManager melempar eksepsi kepada applet yang melakukan operasi yang membahayakan host atau file-filenya (tidak berhak mengaksesnya), misalnya mengakses file sistem lokal, membuka soket ke sebuah host yang berbeda dengan host yang melayani applet, dan lain-lain.

3. Eksepsi Yang Tidak Dapat Ditangkap

Obyek eksepsi secara otomatis dihasilkan oleh runtime java untuk menanggapi suatu keadaan eksepsi. Perhatikan contoh berikut :

```
class Exc0 {
public static void main (String args[]) {
int d = 0;
int a = 42 / d;
} }
```

Saat runtime java mencoba meng-eksekusi pembagian, akan terlihat bahwa pembaginya adalah nol, dan akan membentuk objek eksepsi baru yang menyebabkan program terhenti dan harus berurusan dengan keadaan kesalahan tersebut. Kita belum mengkodekan suatu penangan eksepsi, sehingga penanganan eksepsi default akan segera dijalankan. Keluaran dari program diatas : java.lang.ArithmeticExpression : /by zero at Exc0.main (Exc0.java:4)

Berikut adalah contoh lainnya dari eksepsi :

```
class Exc1 {
static void subroutine() {
int d = 0;
int a = 42 / d;
}

public static void main (String args[]) {
Exc1.subroutine();
}
```

```
}
```

Output-nya : "Exception in thread main java.lang.ArithmeticException : / by zero at Exc1.subroutine(Exc1.java :4) at Exc1.main(Exc1.java : 7)

a. Try dan Catch

Kata kunci try digunakan untuk menentukan suatu blok program yang harus dijaga terhadap semua eksepsi, setelah blok try masukkan bagian catch, yang menentukan tipe eksepsi yang akan ditangkap. Perhatikan contoh berikut:

```
class Exc2 {
public static void main (String args[]) {
try {
int d = 0;
int a = 42 / d;
}
catch (ArithmeticException e) {
System.out.println( "Division By Zero" );
}  } }
```

Outputnya:
C:\Documents and Settings \My Documents>java Exc2 Division By Zero

b. Throw

Pernyataan throw digunakan untuk secara eksplisit melemparkan suatu eksepsi. Pertama kita harus mendapatkan penanganan dalam suatu instance throwable, melalui suatu parameter kedalam bagian catch, atau dengan membuatnya menggunakan operator new.

Bentuk umum pernyataan throw :

throw eksepsi

throw ThrowableInstance;

eksepsi yang dimaksud harus berupa objek Throwable maupun objek dari kelas-kelas turunannya. Kita dapat melempar objek non-throwable, misalnya objek string. Contoh jika kita ingin membangkitkan eksepsi NullPointerException, maka kita dapat menuliskan kodenya sebagai berikut: *throw NullPointerException();*

Aliran eksekusi akan segera terhenti setelah pernyataan throw, dan pernyataan selanjutnya tidak akan dicapai. Blok try terdekat akan diperiksa untuk melihat jika telah memiliki bagian catch yang cocok dengan tipe instance Throwable. Jika tidak ditemukan yang cocok, maka pengaturan dipindahkan ke pernyataan tersebut. Jika tidak, maka blok pernyataan try selanjutnya diperiksa, begitu seterusnya sampai penanganan eksepsi terluar menghentikan program dan mencetak penelusuran semua tumpukan sampai pernyataan throw. Contoh :

```
class throwDemo {
static void demoProc() {
try {
throw new NullPointerException( demo ); }
catch (NullPointerException e) {
System.out.println( "caught inside demoproc" );
```

```
        throw e; }
        }
        public static void main (String args[]) {
        try {
        demoProc();
        }
        catch (NullPointerException e)      {
        System.out.println( "recaugt" :    + e);
        }  }
}
```

Output :
caught inside demoproc
recaught : java.lang.NullPointerException : demo

PRAKTIKUM

Praktikum 1. *ArrayIndexOutOfBoundsException*

C

.

```
public class Eksepsi {
    public static void main(String[] args) {
        // TODO code application logic here
        int[] A = new int[5];
    A[5] = 100;
    }
```

Bagaimana output dari program tersebut?

……

……

Jelaskan alasannya !

……

……

.Modifikasi program pada praktikum 1 menjadi script berikut: **Eksekusi dan analisa**

```
11  public class Eksepsi {
12      public static void main(String[] args) {
13          // TODO code application logic here
14          int[] A = new int[5];
15      try{
16          A[5] = 100;
17      }
18      catch (Exception e){
19          System.out.println ("Anda memasukkan index yang melewati batas");
20       }
21      }
22  }
```

Bagaimana hasilnya?

……

……

Mengapa demikian?

……

……
……
……

Praktikum 2. Kata Kunci **Try dan catch**

```
1   package eksepsi2;
2
3   public class Eksepsi2 {
4       public static void main(String[] args) {
5          int pembilang = 2;
6       int penyebut = 0;
7       try {
8         int hasil = pembilang/penyebut; // menimbulkan eksepsi
9         System.out.println("Hasil = " + hasil); // tidak dieksekusi
10      } catch (ArithmeticException ae) {
11        System.out.println("KESALAHAN: " +
12          "Terdapat pembagian dengan nol");
13      }
14      System.out.println("Statemen setelah blok try-catch");
15    }
16      }
```

Tasks | Output - eksepsi2 (run) | Breakpoints | Variables

Bagaimana hasilnya?

……
……

Mengapa demikian, jelaskan hasil analisa anda!

……
……
……
……

Praktikum 3. Try dan Catch

```
6       public static void main(String[] args) {
7           int pembilang = 2;
8       int penyebut = 0;
9       try {
0         int hasil = pembilang/penyebut;          // SALAH
1         System.out.println("Hasil = " + hasil); // tidak dieksekusi
2       } catch (Exception e) {
3         System.out.println("KESALAHAN: " +
4           "Terdapat pembagian dengan nol");
5       }
6       System.out.println("Statemen setelah blok try-catch");
7     }
```

Bagaimana hasilnya?

……
……

Mengapa demikian, jelaskan hasil analisa anda!

……

……

……

Pada kasus tertentu (biasanya pada proses debugging) mungkin saja anda menginginkan pesan sebenarnya yang terkandung dalam eksepsi yang ditimbulkan. Untuk melakukan hal ini dapat menggunakan method **getMessage()** yang terdapat pada objek eksepsi yang dilewatkan.

Praktikum 4. Method **getMessage()**

```
16      public static void main(String[] args) {
17    int pembilang = 2;
18      int penyebut = 0;
19      try {
20        int hasil = pembilang/penyebut;          // SALAH
21        System.out.println("Hasil = " + hasil); // tidak dieksekusi
22      } catch (Exception e) {
23        System.out.println(e.getMessage());
24      }
25      System.out.println("Statemen setelah blok try-catch");
26      }
27  }
```

Bagaimana hasilnya?

……

……

Praktikum 5. Throw

```
1   package demothrow;
2
3   class Barang {
4     private String kode;
5     private String nama;
6     private double harga;
7
8     public void setKode(String vKode) {
9       try {
10        kode = vKode;
11        if (kode == null) {
12          throw new NullPointerException();
13        }
14      } catch (NullPointerException npe) {
15        System.out.println("KESALAHAN: " +
16          "Kode barang tidak boleh null");
17      }
18    }
19
20    public String getKode() {
21      return kode;
22    }
```

```
24    public void setNama(String vNama) {
25      try {
26        nama = vNama;
27        if (nama == null) {
28          throw new NullPointerException();
29        }
30      } catch (NullPointerException npe) {
31        System.out.println("KESALAHAN: " +
32          "Nama barang tidak boleh null");
33      }
34    }
35
36    public String getNama() {
37      return nama;
38    }
39
40    public void setHarga(int vHarga) {
41      harga = vHarga;
42    }
43
44    public double getHarga() {
45      return harga;
46    }
47  }
49  public class DemoThrow {
50      public static void main(String[] args) {
51         Barang obj = new Barang();
52
53      obj.setKode(null);
54      obj.setNama("Buku tulis");
55      obj.setHarga(2500);
56
57      System.out.println("\nKode  : " + obj.getKode());
58      System.out.println("Nama   : " + obj.getNama());
59      System.out.println("Harga  : " + obj.getHarga());
60      }
61  }
```

Bagaimana output program?

……
……
……
……

Mengapa demikian, jelaskan hasil analisa anda!

……
……
……
……

12 Input dan Output

TUJUAN PRAKTIKUM

Setelah praktikum ini, mahasiswa diharapkan dapat:

1. Memahami dasar-dasar I/O
2. Melakukan input
3. Menampilkan output
4. Kelas dan interface yang berkaitan dengan I/O

PERALATAN PRAKTIKUM

1. Personal Komputer
2. *Software Java Netbeans*

TEORI

Pengenalan Stream

Dalam Java, operasi I/O menggunakan *streams*.*Streams* adalah abstraksi dari sesuatu yang digunakan untuk menulis/menghasilkan dan membaca/mendapatkan suatu informasi. Semua streams memiliki sifat yang sama walaupun peralatan fisik yang berhubungan dengan suatu stream berbeda-beda. Secara umum streams dalam Java dibagi dalam 2 bagian besar :

a. *Byte streams, Byte Streams* digunakan untuk operasi I/O yang menggunakan data biner (byte)
b. *character streams, Character Stremas* digunakan untuk menangani operasi I/O yang menggunakan character.

Karakter dalam java menggunakan Unicode, sehingga penggunaan *character streams* dapat digunakan untuk menangani karakter-karakter internasional (karakter diluar kode ASCII Standar). Semua class & interface yang berhubungan dengan streams ada dalam package **java.io.**

Byte Stream

Java menyediakan dua class abstrak yang merupakan superclass tertinggi untuk Byte Stream, yaitu :

1) *InputStream* untuk membaca input
2) *OutputStream* untuk menuliskan output

Character Stream

Untuk character streams, Java menyediakan dua class abstrak yang merupakan superclass tertinggi yaitu :

1) *Reader* untuk membaca input
2) *Writer* untuk menuliskan output

Variable Stream Standar

Secara default, Java telah menyediakan 3 buah variabel streams yang dapat langsung digunakan, karena variabel ini member **public static** dari class **System**, yaitu : **in,out,err**.

1) **System.out** : output stream standar. Secara default outputnya adalah console.
2) **System.in** : input stream standar. Secara default inputya adalah keyboard.
3) **System.err** : output stream untuk mencetak pesan kesalahan pada console (default).

Membaca Input dari Console menggunakan Byte Streams

1) Untuk membaca dari console digunakan variabel stream standar yang telah disediakan oleh class **System**, yaitu **in**.
2) Variabel ini memegang referensi dari objek dengan tipe **InputStream** sehingga untuk membaca dari console (yang diketik lewat keyboard), dapat menggunakan method **read**

PRAKTIKUM

Praktikum 1. Membaca Input dari Console menggunakan Byte Streams (karakter)

```
1   package demostream;
2   import java.io.*;
3
4   public class Demostream {
5
6       public static void main(String[] args) throws IOException{
7           System.out.print("Masukkan sembarang karakter: ");
8
9       char ch;
10
11      InputStreamReader isr = new InputStreamReader(System.in);
12      BufferedReader br = new BufferedReader(isr);
13
14      ch = (char) br.read();
15
16      System.out.println("Karakter yang dimasukkan adalah \'" + ch + "\'");
17      }
18  }
```

Tasks | Output - demostream (run) | Breakpoints | Variables

Praktikum 2. Membaca Input dari Console menggunakan Data String

Untuk melakukan input berupa string digunakan method *readLine(), bukan read()*

Deklarasi dari method : *String readLine() throws IOException*

```
1  package demoinput;
2  import java.io.*;
3
4  public class Demoinput {
5
6      public static void main(String[] args) throws IOException{
7      System.out.print("Masukkan nama Anda: ");
8
9      String nama;
10
11     InputStreamReader isr = new InputStreamReader(System.in);
12     BufferedReader br = new BufferedReader(isr);
13
14     nama = br.readLine();
15
16     System.out.println("Halo " + nama + ", sudahkah Anda mengerti Java?");
17     }
18 }
```

Praktikum 4. Membaca Input dari Console menggunakan Byte Streams

```
1.  import java.io.*;
2.  public class DemoStream2
3.  {
4.  public static void main(String[] args) {
5.  byte[] data = new byte[10];
6.  int panjang=0;
7.  System.out.print("Masukkan data    : ");
8.  try {
9.  panjang=System.in.read(data);
10. } catch (IOException e) {
11. System.out.print("Terjadi Exception");
12. }
13. System.out.println("Yang anda ketik  : ");
14. for (int i=0;i<panjang;i++) {
15. System.out.print((char)data[i]);
16. }
17. System.out.println("Panjang Karakter : "+panjang);
18. }
19. }
```

Menulis Output ke Console menggunakan Byte Streams

Untuk menulis ke console digunakan variabel stream standar yang telah disediakan oleh class **System**, yaitu **out**. Variabel ini memegang referensi dari objek dengan tipe **PrintStream**. PrintStream merupakan turunan dari class OutputStream. Method yang biasa digunakan : **print(),println(),write().**

Praktikum 5. Menulis Output ke Console menggunakan Byte Streams

```
1.  import java.io.*;
2.  public class DemoStream3
3.  {
4.  public static void main(String[] args) {
5.  byte[] data = new byte[10];
6.  int panjang=0;
7.  System.out.print("Masukkan data    : ");
8.  try {
9.  panjang=System.in.read(data);
10.System.out.print("Yang anda ketik  : ");
11.System.out.write(data);
12.System.out.println("Panjang Karakter : "+panjang);
13.System.out.print("index ke-1 sebnyk 3 : ");
14.System.out.write(data,1,3);
15.} catch (IOException e) {
16.System.out.print("Terjadi Exception");
17.}
18.
19.}
20.}
```

Praktikum 6. Menampilkan Output

Untuk menampilkan output ke layar console, dapat dilakukan dengan method print()maupun println(). Method yang digunakan untuk melakukan proses ini adalah write().

```
1   package demowrite;
2   public class Demowrite {
3
4       public static void main(String[] args) {
5           int i = 'A';
6       byte b = 65;
7       char c = 'B';
8
9       System.out.write(i);
10      System.out.write(b);
11      System.out.write(c);
12      }
13  }
```

Tasks | Output - demowrite (run) | Breakpoints | Variables

```
run:
BUILD SUCCESSFUL (total time: 2 seconds)
```

Membaca Input dari Console menggunakan Character Streams.

Deklarasi konstruktor nya : **InputStreamReader (Input Stream in)**

Membaca Input dari Console menggunakan Character Streams

byte streams (8 bit) ➔ ukurannya lebih kecil dari char (16bit). Bagaimana mengkonversi dari byte streams menjadi char streams dengan benar?

Solusi : input stream sebaiknya dibaca dari buffer, bukan dari peralatan fisik langsung. Untuk bungkus objek dari **InputStreamReader** ke dalam class **BufferedReader.**

Deklarasi konstruktornya : **BufferedReader (Reader in)**

InputStreamReader input = new InputStreamReader(System.in);

BufferedReader buff = new BufferedReader(input);

Atau dengan cara :

BufferedReader buff = new BufferedReader(new InputStreamReader(System.in));

Untuk membaca character streams, dapat menggunakan method :

1) int read() throws IOException
2) int read(char[] cbuf)throws IOException
3) int read(char[] cbuf,int off,int len) throws IOException
4) String readLine() throws IOException

Praktikum 7.

```
1.  import java.io.*;
2.  public class DemoStream6 {
3.  public static void main(String[] args) throws IOException {
4.  char data;
5.  String str="";
6.  BufferedReader buff =
7.  new BufferedReader(new InputStreamReader(System.in));
8.  System.out.println("Ketik  : ");
9.  data = (char) buff.read();
10. while (data!='\r') {
11. str+=data;
12. data = (char) buff.read();
13. }
14. System.out.println("Yang diketik : "+str);
15. System.out.println("Program Selesai");
16. }
17. }
```

TUGAS

Buatlah program kalkulator sederhana dengan menggunakan input dan output dari keyboard

. Tuliskan listing program pada lembar kerja berikut :

DAFTAR RUJUKAN

Raharjo, Budi dkk. 2012. *Mudah Belajar Java*. Bandung: Informatika

Graham, I. 1991. *Object Oriented Methods.* New York: Addison Wesley Inc.

Subiyantoro, Eko. 2013. *Pemrograman Berorientasi Object. Kementerian Pendidikan & Kebudayaan*

Sun Java Course. 2004. *Java Fundamental Programming*.

Sutopo, Aristo Hadi. 2005. *Pemrograman Berorientasi Objek dengan java*. Graha Ilmu

Sukamto, Rosa A., & Shalahuddin, M. 2010. *Modul Pembelajaran: Pemrograman Berorientasi Objek*. Bandung: Modul

ISBN 978-602-5914-58-4