YAYASAN PRIMA AGUS TEKNIK

BELAJAR DENGAN MUDAH
PEMROGRAMAN
BERORIENTASI OBJEK
BAGI PEMULA

Oleh
Muhamad Sidik, S.Kom., M.Kom

BELAJAR DENGAN MUDAH
PEMROGRAMAN
BERORIENTASI OBJEK
BAGI PEMULA

Oleh
Muhamad Sidik, S.Kom., M.Kom

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
JL. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

BELAJAR DENGAN MUDAH PEMROGRAMAN BERORIENTASI OBJEK BAGI PEMULA

Penulis :
Muhamad Sidik, S.Kom., M.Kom

ISBN : 978-623-8120-03-1

Editor :
Dr. Joseph Teguh Santoso, S.Kom., M.Kom.

Penyunting :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom.

Penebit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. (024) 6723456
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Alkhamdulillah, segala syukur bagi Allah SWT yang hanya dengan ijin dan kehendaknya buku karya tulis Pemrograman Berbasis Objek ini dapat selesai. Tujuan pembuatan Buku karya tulis adalah untuk ilmu pengetahuan dan memahami penuh konsep Pemrograman Objek dalam mengembaangkan suatu sistem.

Materi yang dikaji adalah : Pemahaman PBO, Pengenalan PBO dan Install Konfigurasi Netbeans Java, Dasar Bahasa Java, Pengertian Java Swing, Membuat Java Swing Sederhana, Struktur Kontrol, Pengaplikasian Struktur Kontrol, Array, Database, iReport.

Semoga harapan penulis apa yang ada di dalam buku barokah dan berkah untuk pembaca semua. Tak lupa saya ucapkan penuh rasa sukur pada pihak - pihak yang terlibat dan telah menolong dalam merampungkan karya tulis mata kuliah ini. Harapan penulis adanya keberadaan karya tulis buku mata kuliah ini bisa berguna dan barokah bagi pembaca.

Semarang , 2 Desember 2022

Penulis

Muhammad Sidik, S.Kom, M.Kom

BAB 1

PENGANTAR PERMPROGRAMAN BERBASIS OBJEK

Pengantar Permprograman Berbasis Objek Computer Memiliki System Kerja Seperti Control Switch Panel Dan Hanya Membaca Mengenali angka 0 Dan angka 1 Tetapi brainware tidak memahami bahasa mesin Assembler 0 Dan 1 Tersebut Karna Itu Bahasa Mesin. Perlu Bahasa Pemrogarman Yang Dapat Dijadikan Penghubung Atau Pelantara Percapakap Antara Brainware Dan Hadware Tersebut. Bahsa Pemrograman Dirubah Kedalam Bentuk Bahasa Manusia Yang Mudah Di Mengerti, Begitu Sebaliknya Yaitu Menggunakan Compiler Dan Atau Interpreter.

1.1 Compiler Or Interpreter?

Tingkat bahasa pemrograman

Compiler :

Mengkompilasi source code menjadi bentuk file yang bisa di eksekusi

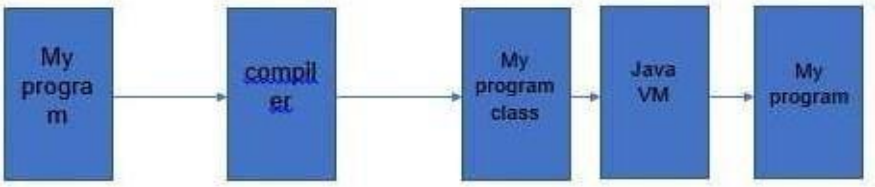
Interpreter :

Mengompilasi dan menjalankan source code secara langsung

C language (compiler)

Gambar 1.1 Compiler

Java lengaue compiler + interpreter java



Gambar 1.2 compiler + interpreter

Bahasa pemrograman memiliki beberapa tingkatan, di antaranya yaitu bahasa pemrograman asember yaitu bahasa pemrogaman tingka rendah atau bahasa mesin. Selanjutnya ada bahasa pemrograman tingkat sedang yaitu program c pascal dan yang terakhir adalah bahasa pemrograman tingkat tingi dan lebih mudah dalam mengembangkanya lebih terstruktur dan bahasa pemrogaman lebih mudah di pahami manusia diantaranya yaitu bahasa c ++ dan bahasa java.

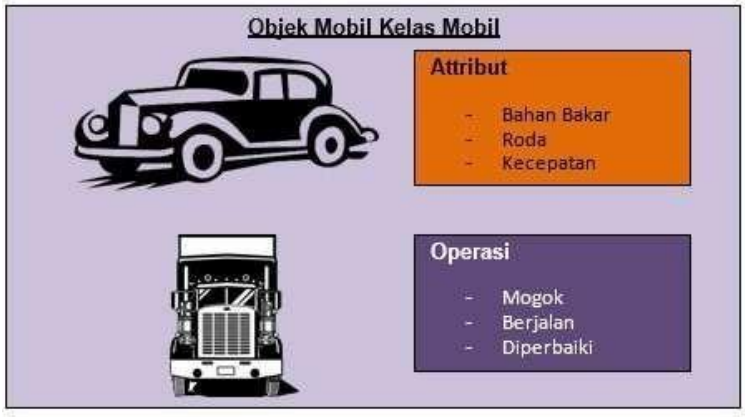
1.2 Paradigma Pemrograman

Sudut dan style program yang saling berinteraksi agar bagaimana suatu problem digunakan formulasikan dalam bahasa pemrograman function programing ini merupakan fungsi urutan secara sekuensial , procedural programing: penyelesaian problem sesuai langkah-langkah kerja yang sessuai terkelompoik dalam suatu unit program pengembangan C dan program pascal.

Objek programing oriented mengoneksikan objek yang saling terhubung , class merupakan unit program bahasa java c ++ dan lain sebagainya.

Teknologi Bahasa Java Dan Pendukung Perangkat Kompilasi, Instalasi, Perangkat Bahasa Java Family Karakteristik Obyek Untuk lebih jelasnya karakteristik objek tersebut dijelaskan sebagai berikut :

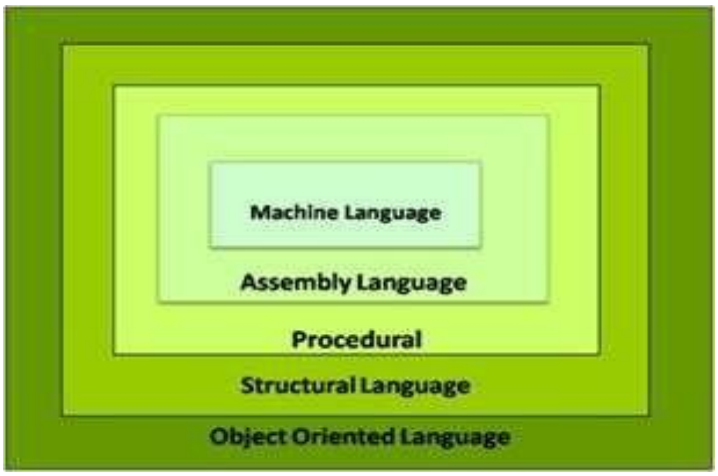
Gambar 1.3 Contoh obyek



Gambar 1.4 Contoh obyek dengan properti

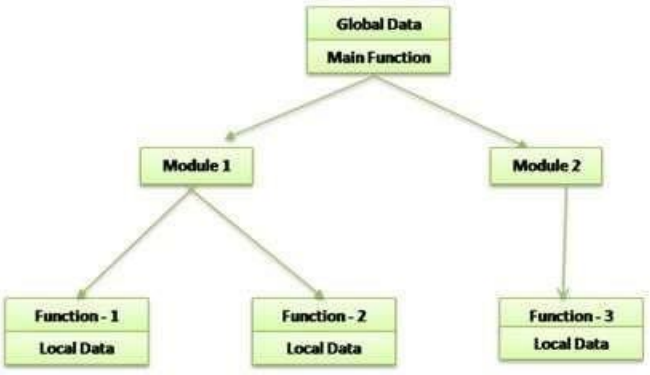
Perbandingan Pemrograman Berorientasi Obyek dengan Pemrograman Terstruktur

Paradigma bahasa pemrograman memberikan model untuk programmer dalam menulis listing program. Paradigma perbedaan dalam bahasa pemrograman sebagai berikut :



Gambar 1.5 Bahasa pemrograman

Gambar 1.6 Diagram bahasa pemrograman



Gambar 1.7 Diagram bahasa pemrograman

a. Terstruktur

Program ini dibagi menjadimodul dan modul tersebut kemudian dibagi menjadi fungsi.

Penggunaan Pernyataan *goto*dihapus atau dikurangi. Setiap modul dapat bekerja independen satu sama lain.

Tabel 1.1 Fase Sebuah Pemrograman Java

Pemrograman Terstruktur	Pemrograman Berorientasi Obyek
Pendekatan *top-down*	Pendekatan *bottom-up* yang diikuti.
Fokus adalah pada algoritma dan kontrol aliran.	Fokus pada model obyek.
Program dibagi menjadi beberapa sub modul atau fungsi atau prosedur.	Program ini diselenggarakan dengan memiliki sejumlah kelas dan objek.
Fungsi yang independen satu sama lain.	Setiap kelas berhubungan secara hirarkis.
Tidak ada penerima yang ditunjuk dalam panggilan fungsi.	Ada penerima yang ditunjuk untuk setiap lewat pesan.
Data dan fungsi sebagai dua entitas yang terpisah *Views*.	Data dan fungsi sebagai satu kesatuan pandangan.
Pemeliharaan mahal.	Pemeliharaan relatif lebih murah.
Reuse Software tidak mungkin.	Membantu dalam penggunaan kembali perangkat lunak.
Fungsi panggilan digunakan.	Message passing digunakan.
Fungsi abstraksi digunakan.	Data abstaction digunakan.
Algoritma diberikan penting.	Data diberikan penting.
Solution adalah solusi spesifik-domain.	*Solution* adalah spesifik masalah domain.
Tidak ada enkapsulasi. Data dan fungsi yang terpisah	Enkapsulasi paket kode dan data sama sekali. Data dan fungsi disatukan dalam satu kesatuan.
Hubungan antara programmer danprogram ditekankan.	Hubungan antara programmer danpengguna ditekankan.
Teknik data-driven digunakan.	Didorong oleh delegasi tanggung jawab.

Bahasa Pemrograman Java

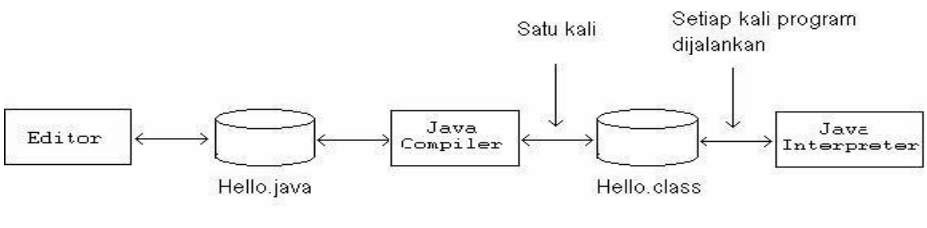
Pada bulan juni tahun 1992 bahasa java awalnya di sebut oak setelah nya kemudian diberinama menjadi java, kemudian menerapkan virtual machine and memiliki gaya notasi c ++, kemudian sun microsystem merilis public pertama bahasa java pada tahun 1995 bulan mei tahun 2007 sun menyelesaikan suatu perkembangan membuat kode java tersedia di open source.

1.3 Bahasa Pemrograman Java

Java dikembangkan oleh sebuah tim yang dipimpin oleh James Gosling di perusahaan Sun Microsystems. Sebelumnya bernama “Oak” yang dirancang pada tahun 1991 untuk digunakan dalam chip tertanam di peralatan elektronik. Pada tahun 1995, berganti nama menjadi Java dan didesain ulang untuk mengembangkan aplikasi web. Pada tahun 2010, Sun Microsystems diambil-alih oleh perusahaan Oracle. Java merupakan bahasa pemrograman yang sangat populer. Pengembangan pemrograman Java yang cepat dan kompatibel dapat dilihat dari karakteristik desainnya, dimana Anda dapat menulis program sekali dan dapat dijalankan/dieksekusi dimana saja.

Fase – Fase Pemrograman Java

Gambar dibawah ini menjelaskan aliran proses kompilasi dan eksekusi sebuahprogram Java :

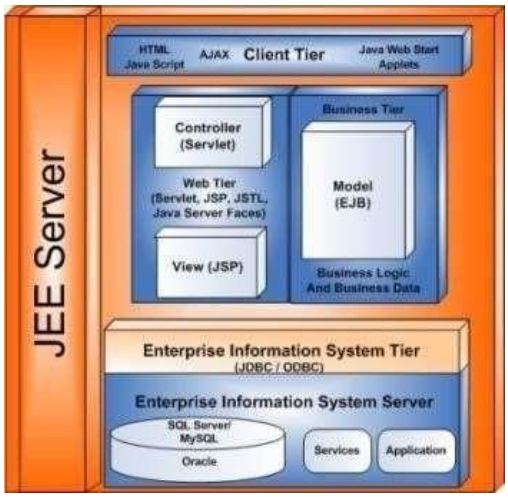


Gambar 1.7 Fase dari sebuah Program Java

1.4 Pemahasan Bahasa Java Suite Family

Java Standard Edition (JavaSE) untuk desktop, client/serverapplication

Java Enterprise Edition



Gambar 1.8 java server

Instalasi konfigurasi java se dan tols netbeans ide

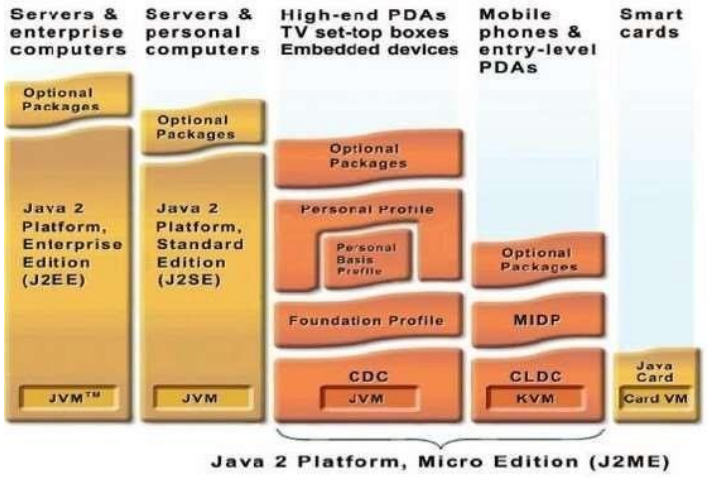
Gambar 1.9 Arsitektur Teknologi Java

Instal java

Instal netbeans

Path set instalasi dengan cara klik star > control panel > system>advance>environment vaiables set path pada c/programfile/java/sdk/bin

1.5 Arsitektur Teknologi Java



Gambar 1.10 Arsitektur Teknolog java

a. Berorientasi objek (*ObjectOriented*)

Java mengunakan pemrograman berorientasi objek yang membuat program dapat dibuat secara modular dan dapat dipergunakan kembali. Pemrogramanberorientasi objek memodelkan dunia nyata ke dalam objek dan melakukan interaksi antar objek-objek tersebut.

Dapat didistribusi dengan mudah

Java dibuat untuk membuat aplikasi terdistribusi secara mudah denganadanya *libraries networking* yang terintegrasi pada Java.

Interpreter

Program Java dijalankan menggunakan interpreter yaitu *Java Virtual Machine* (JVM). Hal ini menyebabkan *sourcecode* Java yang telah dikompilasi menjadi Java *bytecodes*

1.6 Mengenal Java SDK

Java dikembangkan oleh tim yang dipimpin oleh Patrick Naughton dan James Gosling di Proyek Kode Hijau Sun Microsystems. Hal ini bertujuan untuk membuat bahasa komputasi sederhana yang dapat berjalan pada perangkat sederhana tanpa terikat pada arsitektur tertentu. James Gosling awalnya menamai bahasa pemrograman yang dihasilkan OAK, tetapi OAK sendiri adalah nama bahasa pemrograman komputer yang ada, yang diubah Sun menjadi Java.

Setelah beberapa konversi dan proses, Sun akhirnya meluncurkan browser Java bernama Hot Java yang dapat menjalankan applet. Banyak versi Java dikembangkan hingga JDK hadir dengan

teknologi Swing untuk menampilkan GUI. Teknologi Java yang dikenal dengan J2ME (Java 2 Micro Edition) telah diadopsi oleh Nokia, Siemens, SonyEricson, Motorola, dan Samsung untuk mengembangkan aplikasi mobile, baik game maupun software enterprise, serta berbagai macam aplikasi yang berjalan di perangkat mobile seperti sebagai smartphone. Menciptakan perangkat lunak untuk B. Ponsel dapat menjalankan telepon.

SDK adalah singkatan dari Standard Development Kit dan merupakan 7 Attribute7 utama bagi pengembang untuk membuat program dan menjalankan program Java. Pada tingkat tinggi, Java SDK terdiri dari:

Kompiler Java (Javac)

Mesin Virtual Java (sering disebut Java Runtime Environment/JRE)

Java Class Library (kumpulan kelas yang dapat digunakan untuk menghasilkan program Java)

Java AppletViewer (program untuk menjalankan applet tanpa browser)

Debugger Java dan alat lainnya

Pertama, kode program yang ditulis dalam Java (file dengan ekstensi .java) dikompilasi oleh compiler menjadi kode objek dalam format file dengan ekstensi .class yang disebut bytecode. Jadi di Java, hasil akhir dari program ini adalah file dengan ekstensi .class daripada file dengan ekstensi .EXE. File bytecode ini juga dieksekusi baris demi baris oleh interpreter.

Oleh karena itu, proses kompilasi hanya dilakukan satu kali, tetapi proses interpretasi dilakukan setiap kali program dijalankan. Istilah “tulis sekali, jalankan di mana saja” dikenal dalam teknologi Java sebagai konsep bytecode ini. Ini berarti bahwa setelah Anda menulis dan mengkompilasi program di Java, Anda dapat menjalankan bytecode tersebut pada platform 7ttrib operasi apa pun selama 7ttribut Anda memiliki Java Virtual Machine (JVM) yang diinstal.

a. Tipe Data

Dalam bahasa Java, tipe data dasar dapat dibagi menjadi empat kelompok: tipe data integer, tipe data floating-point (real), tipe data karakter, dan tipe Boolean. Perhatikan bahwa String di Java adalah objek, bukan tipe data, tetapi penggunaannya mirip dengan menggunakan tipe data.

b. Tipe Data Integer

Tipe data integer (bilangan bulat) di Java dibagi menjadi empat tipe: byte, short, int, dan long. Semua tipe data ini ditandatangani. Yaitu, tipe data yang dapat mewakili nilai 7ttribut atau positif. Tidak seperti kebanyakan bahasa pemrograman lainnya, Java tidak mendukung nilai yang tidak ditandatangani (tipe tidak ditandatangani, tipe data yang hanya dapat menyimpan nilai positif).

Tabel 1.3 tipe data integer

Tipe Data	Ukuran (*bit*)	Rentang
Byte	8	-128 s.d. 127
Short	16	-32.768 s.d. 32.767
Int	32	-2.147.483.648 s.d. 2.147.483.647
Long	64	-9.223.372.036.854.775.808 s.d. 9.223.372.036.854.775.807

c. Tipe Data Floating-Point

Tipe float (nyata) digunakan untuk mewakili nilai yang mengandung pecahan atau 8ttribu setelah titik. Tipe data ini dibagi menjadi dua tipe: float dan double.

Tabel 1.4 tipe data floating-point

Tipe Data	Ukuran (*bit*)	Rentang
Float	32	3.4e+038 s.d. 3.4e+038
double	64	1.7e+308 s.d. 1.7e+308

d. Tipe Data Karakter

Di Java, tipe data char adalah karakter Unicode yang dapat mewakili karakter apa pun yang ada, yaitu karakter apa pun yang muncul di semua bahasa (internasional). B. Latin, Arab, Yunani, dll.

e. Tipe Data Boolean

Seperti pada kebanyakan bahasa pemrograman, tipe data Boolean Java digunakan untuk menyimpan nilai logika, benar dan salah. Perhatikan bahwa di Jawa, benar tidak sama dengan 1 dan salah tidak sama dengan 0. Boolean adalah tipe data yang dikembalikan oleh semua operator rasional.

Sebuah variabel instan dikaitkan dengan sebuah instance dari sebuah kelas. Oleh karena itu, ini hanya dapat digunakan saat membuat instance kelas. Metode statis tidak terkait dengan instance kelas, jadi Anda tidak dapat menggunakan variabel instan dalam metode statis dan memasukkannya ke dalam metode utama.

Variabel 8ttri adalah variabel yang dideklarasikan dalam badan metode. Jadi variabel hanya dapat digunakan di dalam metode. Metode kelas lain tidak peduli dengan keberadaan variabel. Juga, variabel 8ttri hanya ada ketika metode (yang memiliki variabel 8ttri) dijalankan. Deklarasi tidak perlu menambahkan kata static (seperti saat mendeklarasikan variabel kelas). Jika kata static digunakan dalam deklarasi variabel 8ttri, 8ttribut akan menghasilkan pesan kesalahan dan menolak untuk mengkompilasi program.

Variabel dideklarasikan dengan terlebih dahulu menyebutkan tipe data yang digunakan kemudian menyebutkan nama variabelnya, seperti pada contoh berikut.

Tabel 1.5 variabel

int angka;
int angka1, angka2;
double x,y,z;
char simbol;
String Kalimat;

Pada saat deklarasi variabel, kita juga dapat langsung menginisialisasi nilai dari variabel tersebut, seperti contoh di bawah ini.

Tabel 1.6 Fase Sebuah Pemrograman Java

int angka1 =100;
int angka1 = 100, angka2 = -5;
double x = 1/12;
char simbol = '+';
String kalimat = "Modul Praktikum PBO - UTY";

Operator

Java merupakan bahasa pemrograman yang kaya akan operator. Secara umum, operator di dalam Java dibagi menjadi lima bagian: operator aritmatika, relasional, logika, dan bitwise, dan assignment.

1. Operator Aritmatika

Operator ini digunakan pada operasi-operasi aritmatika seperti penjumlahan, pengu angan, pembagian dan lain-lain. Tabel berikut ini menunjukkan daftar 9ttri operator aritmatika di dalam Java.

Tabel 1.7 Fase Sebuah Pemrograman Java

Operator	Keterangan
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian
%	Modulus (sisa bagi)
++	Increment (menaikkan nilai dengan 1)
--	Decrement (menurunkan nilai dengan 1)

Operator	Contoh	Ekuivalen dengan
+=	b += a	b = b+a
-=	b -= a	b = b-a
*=	b *= a	b = b*a
/=	b /= a	b = b/a
%=	b %= a	b = b%a

BAB 2

DASAR PEMROGRAMAN JAVA

2.1 Penggunaan Variabel, Data Tipe Dan Operator Pada Java

Yang melatar belakangi pembahasan pada bab ini pemrograman java ini,akan mendiskusikan dan membahas mengenai dasar bahasa Java. Meliputi menggunakan variabel, data tipe, dan operator. Pada pembahasan bahasa pemrograman Java dan bagaimana menyelesaikan error ketika menginputkan kode bahasa Java.

Uji coba

uji coba 1 (salam.Java)

```
class salam {

public static void main (String[] args){

System.out.println("salam");

}
```

Hasil :

Salam

Keterangan:

Segala project java, yang nama file nya memiliki extensi .java mempunyai nama kelas yang sama pada code project, setiap blok statemen nya selalu di mulai denan kurawal bukua { dan pada statement akhir di akhire dengan kurawal tutup }, tanpa mengunakan kurawal buka dan tutup suatu code akan di nyatakan mengalaman kesalahn dalan penulisan code sintak tersebut.

Statemen public

Static void main (String[]arg) adalah metode inti titik awal memulai suatu project java, suatu main metode juga dibuka dengan kurawal {dan ditutup menggunakan kurawal tutup } , sama seperti suatu kelas jika tanpa tanda kurawal buka dan tutup makan sintak code akan mengalami eror dalam statement nya.

Statemen System.out.println("Halllo Indonesia " adalah)

metode berguna mencetak string ke tampilan monitor. Teks didalam method

System.out.println () selalu dikawal oleh tanda kutip dua (" ")

Identifier atau atribut tidak menggunakan bentuk kata key atau kunci key yang sudah didefinisikan dan di deklarasikan ke bahasa Java yang dipakai untuk suatu perintah tertentu. Dibawah ini merupakan daftar code unik dalam bahasa Java :

abstract	continue	for	new	switch
assert***	default	goto*	package	synchronized
boolean	do	if	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum****	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp**	volatile
const*	float	native	super	while

a

* not used
** added in 1.2
*** added in 1.4
**** added in 5.0

uji coba 2 (variabel-tipedata.Java)

```
public class variabel {
public static void main (String[] args) {
char var1='Z';
int var2=10;
double var3=0.8;
boolean var4=true;
System.out.println( "Isi variabel var1 :  " +var1);
System.out.println( "Isi variabel var2 :  " +var2);
System.out.println( "Isi variabel var3 :  " +var3);
System.out.println( "Isi variabel var4 :  " +var4);}
}
```

result:

isi variabel1 : z

isi variabel2 : 10

isi variabel3 : 0.8

isi variabel4 : benar

keterangan :

Variabel merupakan item yang dipakai file atau file untuk save pernyataan suatu objek. Variabel memiliki 11ttribute dan data tipe. Mendeklarasi suatu variabel dapat dituliskan dengan :

<tipe-data> nama-variabel [= initial value]

Selanjutnya uji coba d u a (2) diatas, c o d e (statemen var2=10;) adalah data tipe adalah nama suatu varibabel itu memilikinilai di dalamnya tersebut.

Pemahanan variabel yang dapat di bagi menjadi beberapa, di antaranya suatu reference dan sederhanaPenjelasan variabel primitive merupakan suatu type data primitive. Fungsi nya

menggunakan dan menyimpan file dalam local tertentu dalam memory dimana tempat. Reference merupakan variabel yang menyimpan khusus dalam local memory yang mengarahkan tempat tersebut berada.

Variabel menjadi 2 type reference dan primitive masin- masing memiliki definisi dengan tipe data yang berbeda.

Selanjutnya merupakan type data jenis carakter, type data jenis ini memiliki cara penulisan khusus jika dalam tanda single yaitu memberikan tanda petik (“”) contoh “z”

Tabel 2.1 Tipe Float dan range

Float length	Range	Name or type
32 bit	-2 to 2-1	Float
64 bit	-2to2-1	double

Tipe data Boolean yang di nyatakan dalam 2 pernyataan : benar dan salah dalam contoh Boolean varibel pertama adalah true , sedangkan varibael var 1 adalah pernyataan true.

uji coba 3 (operatoraritmatik.java)

public class operatoraritmatik {

public static void main (String[] args){

```
int a = 10;
int b = 20;
int c;
System.out.println( “Operator Aritmatika :” );
System.out.println( “Penambahan : ” );
c = a+b;
System.out.println( “Isi  c =”  +c);
System.out.println( “Pengurangan : ” );
c = b-a;
System.out.println( “Isi  c  =”  +c);
System.out.println( “Perkalian : ” );
c = a*b;
System.out.println( “Isi variabel c =”  +c);
System.out.println( “Pembagian : ” );
c = b/a;
System.out.println( “Isi variabel c =”  +c);
System.out.println( “Sisa Hasil Bagi : ” );
c = b%a;
Sytem.out.println( “Isi variabel c =”  +c);
}
}
```

Hasil :

```
Operator Aritmatika :
Penambahan :
Isi variabel c = 30
Pengurangan :
Isi variabel c = 10
Perkalian :
Isi variabel c = 200
Pembagian :
Isi variabel c : 2
Sisa Hasil Bagi :
```

Isi variabel c : 0

Pembahasan :

Operator aritmatik yang digunakan dalam project program java tau bagaimana operator mengikuti prioritasnya mana yang akan di jalankan terlebih dahulu dan mana yang akan di jalankan secara bersama- sama dalam suatu pernyataanya.

Tabel a2.2 perator aritmatika beserta 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 pembagianop 1 dengan op2
-	op1-op2	Mengurangkan op2 dari op1

Uji coba 5 (operatorrelasi.java)

```
public class operatorrelasi {
public static void main (String  args) {
int x=10;
int y=30;
System.out.println( "x > y adalah  "   (x>y));
System.out.println( "x >= y adalah  " - (x>=y));
System.out.println( "x < y adalah  " + (x<y));
System.out.println( "x <=           "
System.out.println( "x   y adalah  " ! (x==y));
```

System.out.println (x != y adalah " – (x1y1);

Hasilnya :

X > y = salah

X > = y = salah

X < y = benar

X y = salah

X ! = y = benar

Keterangan :

Suatu operator relasi hubunangan adalah membandingkan nilai, dimana nilai- nilai tersebut berasal dari keterangan. Hasil dari keluaran perbandingan nilai tersebut menjadi type data Boolean yang menghasilkan nilai benar atau salah

Tabel 2.3 Operator hubungan atau 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	Op1 kurang dari atau sama dengan op2
==	Op1 == op2	Op1 sama dengan op2
!=	Op1 != op2	Op1 tidak sama dengan op2

Uji coba 6 (operatorlogika.java)

```
public class operatorlogika {
public static void main(String[] args){
int y=40;
int z=80;
tes1=(y>40) && (z<100);
System.out.println( "Hasil tes1 :  " +tes1);
tes2=(y>40) || (z<100);
```

“Hasil tes1 : “ tes2);

}

}

Hasil :

Hasil tes 1 : salah

Hasil tes 2 : benar

Pembahasan :

Statemen operator logika memiki operator Boolean yang outpunya menghasilakn nilai Boolean baru, operator logika and (&&) dan Boolean logika & mempunyai logika nilai untuk table di bawah ini ;

contoh :

Tabel berikut untuk (&&) dan(&) Tabel kebenaran

Tabel 2.4 Fase Sebuah Pemrograman Java

X1	X2	Hasil
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

Seperti contoh ini jika didapati pernyataan exp 1 && exp 2 , operator logical and (&) akan mengevaluasi pernyataan tersebut di atas exp 1. Dan langsung mengembalikan nilai salah dan menyatakan bahawa exp 1 bernilai salah.

Jika exp 1 memiliki nilai salah, operator tidak mengevaluasi exp 2 karena hasil operasi operator akan berubah menjadi salah tanpa memperhatikan nilai dari exp 2 lain, sebaliknya operator logika and (&) selalu melakukan evaluasi dari kedua nilai tersebut exp 1 dan xpt 2 sebelum mengembalikan jawaban nilai tersebut

Operator logika or dan Boolean logika or , table kebenaranya seprti pada table dbawah ini :

Tabel 2.5 berikut Tabel kebenaran untuk (||) dan (|).

Logika1	Logika2	hasil
Benar	Benar	Benar
Benar	Salah	Benar
Salah	Benar	Benar
salah	salah	salah

2.2 Data Tipe Dan Operator Pada Java

Tipe Data

Tipe data merupakan klasifikasi data dengan bertujuan untuk mengenalkan data atau nilai yang digunakan kepada kompiler. Dalam pemrograman Java, terdapat dua kelompok tipe data yaitu tipe data primitif/sederhana dan tipe data referensi/komposit.

Tipe Data Primitif/Sederhana

Tipe data primitif/sederhana yaitu tipe data dasar yang tersedia/tertanam dan merupakan keyword dalam pemrograman Java. Tipe data primitif hanya dapat menampung satu nilai. Contoh penggunaan tipe data primitif adalah sebagai berikut :

Adapun tipe data primitif dalam pemrograman Java dapat dilihat pada tabel 2.1 dibawah ini.

Tabel 2.6 Tipe Data Primitif

Tipe Data	Deskripsi	Ukuran(bit)	Nilai Minimum	Maksimum
boolean	true / false	1-bit		
char	KarakterUnicode	16-bit		
byte	Bilanganbulat	8-bit	-127	128
short	Bilangan bulat	16-bit	-32768	32767
int	Bilanganbulat	32-bit	-2147483648	2147483647
long	Bilanganbulat	64-bit	- 9223372036 854775808	922337203685477 580 7
float	Bilangan riil	32-bit	1.401298464 32481707e- 45	3.40282346638528 86 0e+38

double	Bilangan riil	64-bit	4.940656458 41246544e- 324	1.79769313486231 57 0e+308

Tipe Data Referensi

Tipe data referensi secara umum adalah tipe data buatan yang mereferensikan *Object* atau *Class* dan bukan merupakan *keyword* dalam pemrograman Java. Tipe dataini dapat menampung lebih dari satu nilai. Contoh penggunaan tipe data referensi adalah sebagai berikut :

10 nilai (termasuk spasi) bertipe char

5 nilai bertipe short

5 nilai bertipe int

Enumerasi

Enumerasi (*enum*) merupakan tipe data buatan dimana nilai enum dapat didefenisikan sendiri sesuai kebutuhan. Nama enum harus dituliskan pada saat variabel dideklarasi dan variabel hanya dapat menggunakan nilai yang sudah didefenisikan dalam enum tersebut sebagai konstanta. Berikut sintaks atau bentuk penulisan enum.

modifier *enum* namaEnum {

nilai1, nilai2, ..., nilai-n

}

Contoh :

```
public class enumerasi {

    enum DaftarHari{
        Senin, Selasa, Rabu, Kamis, Jumat, Sabtu, Minggu
    }

    enum DaftarWarna{
        Hitam, Putih, Merah, Biru, Hijau, Kuning
    }

    public static void main (String[] args){
        DaftarHari hari = null;
        DaftarWarna warna = null;

        System.out.println("Hari : " + hari.Senin);
        System.out.println("Warna : " + warna.Merah);
        System.out.println("Hari : " + hari.Jumat);
        System.out.println("Warna : " + warna.Putih);
    }
}
```

```
run:
Hari : Senin
Warna : Merah
Hari : Jumat
Warna : Putih
```

Identifier

Identifier merupakan pengenal atau identitas yang berupa nama untuk variabel, sub rutin (prosedur atau fungsi), kelas, interface, maupun namespace. Berikut aturan penamaan identifier :

Terdiri dari huruf (a-z / A-Z) atau kombinasi dengan angka (0-9). Namun, tidakboleh diawali dengan angka.

Bersifat *case sensitive* yaitu membedakan penggunaan huruf besar dan hurufkecil.

Tidak diperbolehkan menggunakan spasi. Untuk membuat pemisah dapatmenggunakan simbol *underscore* (_).

Tidak diperbolehkan menggunakan kata *keyword* dan simbol-simbol operatoryang dikenal oleh pemrograman Java.

Operator

Secara umum, operator dalam bahasa pemrograman berupa simbol yang menjalankan operasi tertentu. Terdapat beberapa operator dalam pemrograman java yaitu :

Operator Aritmatika

Operator ini menjalankan operasi bilangan dasar (aritmatika) yaitu penjumlahan (+), pengurangan (-), perkalian (*), pembagian (/), dan modulus atau modulo (%). Contoh :

```
short A=5+3;
short B=5-3;
short C=5*3;
short D=5/3;
short E=5%3;
```

```
run:
A = 8
B = 2
C = 15
D = 1
E = 2
```

Operator *Increment/Decrement*

Operator ini menjalankan operasi penjumlahan dengan kelipatan 1 (*increment*) dan operasi pengurangan dengan kelipatan 1 (*decrement*) dari nilai variabel sebelumnya. Operator *increment* (++) dan *decrement* (--) dapat ditempat pada awal maupun akhir variabel. Contoh :

```
short A=5;
short B=3;
```

```
A++;
B--;
```

```
run:
hasil A++ = 6
hasil B-- = 2
```

Pembahasan :

A++

A = A + 1

B--

B = B - 1

A = 5 + 1

B = 3 – 1

A = 6

B = 2

Operator Penugasan (*Assignment*)

Operator ini mirip seperti operator *increment/decrement* namun lebih variatif.Operator penugasan dapat menggunakan seluruh operasi aritmatika dengan kelipatan angka yang berbeda-beda. Contoh :

C = 20

D = 1

E%=6

E = E % 6

E = 5 % 6

E = 5

Operator Pembanding

Operator ini disebut operator relasi yaitu operator yang berfungsi untuk mengetahui kebenaran dari hubungan nilai operan kanan dan kiri dengan cara membandingkannya. Operator ini akan menghasilkan nilai bertipe boolean yaitu *True/False*.

Tabel 2.7 Operator Pembanding

Operator	Keterangan
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar

>=	Lebih besar sama dengan
<	Lebih kecil
<=	Lebih kecil sama dengan

Contoh :

```
boolean A=(5!=7);
boolean B=(10<=7);
```

```
run:
pernyataan 5!=7 adalah true
pernyataan 10<=7 adalah false
```

Operator Logika

Seperti halnya operator pembanding, operator logika juga termasuk kedalam operasi relasi yang berfungsi untuk mengetahui kebenaran dari hubungan nilai operan kanan dan kiri serta menghasilkan nilai bertipe boolean. Namun, perbedaannya adalah operator logika hanya menghubungkan nilai operan bertipe boolean saja.

Tabel 2.8 Operator Logika

Operator	Nama	Keterangan
&&	dan (and)	Menghasilkan *true* jika hanya kedua nilai operan bernilai *true*
||	atau (or)	Menghasilkan *true* salah satu nilai operan bernilai *true*
!	tidak (not)	Menghasilkan kebalikan nilai operan

Contoh :

Operator logika and (&&)

```
boolean A=(true && true);
boolean B=(true && false);
boolean C=(false && true);
boolean D=(false && false);
```

```
run:
logika A adalah true
logika B adalah false
logika C adalah false
logika D adalah false
```

Operator logika or (|||)

```
boolean A=(true || true);
boolean B=(true || false);
boolean C=(false || true);
boolean D=(false || false);
```

```
run:
logika A adalah true
logika B adalah true
logika C adalah true
logika D adalah false
```

Operator logika not (!)

```
boolean A=true;
boolean B=false;
```

```
run:
logika !A adalah false
logika !B adalah true
```

Operator Bitwise

Operasi bitwise adalah operasi pengolahanan/manipulasi bit atau bilangan biner (0/1). Angka 0 menjadi nilai *false* dan angka 1 menjadi nilai *true*. Setiap nilai baik angka (0-9) maupun huruf (a-z dan A-Z) akan diubah kedalam bentuk bilangan biner kemudian dioleh sesuai dengan operator bitwise yang digunakan. Adapun operatorbitwise dapat dilihat pada

Tabel 2.9 Operator Bitwise

Operator	Nama	Keterangan
&	And	Sama seperti operator logika and
|	Or	Sama seperti operator logika or
^	XOR (ExclusiveOr)	Menghasilkan *true* jika salah satu nilai operan bernilai *true* dan salah satu operan bernilai *false*
~	Not	Sama seperti operator logika not
<<	Left Shift	Menggeser bit beberapa digit ke kiri
>>	Right Shift	Menggeser bit beberapa digit ke kanan

Contoh :

Operator AND (&)

```
short A=19;
short B=85;
int C=A&B;
System.out.println("hasil AND pada C adalah "+C);
```

```
run:
hasil AND pada C adalah 17
```

Pembahasan :

Nilai 19 dan 85 (merupakan bilangan desimal dalam tabel ASCII) dikonversimenjadi bilangan biner.

Bilangan biner dari masing-masing angka tersebut akan direlasikan denganlogika AND (lihat operator logika AND) sesuai dengan urutan digitnya.

Hasil relasi logika AND dikonversikan kembali menjadi bilangan desimal yaitu17.

Operator OR (|)

Pembahasan :

Nilai 19 dan 85 (merupakan bilangan

desimal dalam tabel ASCII) dikonversimenjadi bilangan biner.

Bilangan biner dari masing-masing angka tersebut akan direlasikan denganlogika OR (lihat operator logika OR) sesuai dengan urutan digitnya.

Hasil relasi logika OR dikonversikan kembali menjadi bilangan desimal yaitu87.

= 87

Operator XOR (^)

```
short A=19;
short B=85;
int C=A^B;
System.out.println("hasil XOR pada C adalah "+C);
```

```
run:
hasil XOR pada C adalah 70
```

Pembahasan :

Nilai 19 dan 85 (merupakan bilangan desimal dalam tabel ASCII) dikonversimenjadi bilangan biner. Bilangan biner dari masing-masing angka tersebut akan direlasikan denganlogika XOR sesuai dengan urutan digitnya. Logika XOR akan menghasilkan nilai true jika kedua operan memiliki nilai yangberbeda. Hasil relasi logika XOR dikonversikan kembali menjadi bilangan desimal yaitu70.

Operator Ternary

Konsep kerja operator ini mirip seperti pernyataan IF dengan 2 kondisi, namundisederhanakan menjadi 1 baris berupa pertanyaan. Contoh

```
short A=9;
int B=A%2;
String bilangan;
bilangan = (B==1)? "Ganjil" :"Genap";
System.out.println(A+" adalah bilangan "+bilangan);

run:
9 adalah bilangan Ganjil
```

Pembahasan :

Bilang = B==1) “ganjil” : “genap”;

1 2 3

Operator Ternary terdiri dari 3 parameter yaitu :

Variabel yaitu bilangan dan Kriteria/Kondisi yaitu (B==1).

Nilai jika kriteria/kondisi terpenuhi (bernilai *true*).

Nilai jika kriteria/kondisi tidak terpenuhi (bernilai *false*).

Adapun nilai yang terpilih akan disimpan pada variabel yang tertulis pada parameter 1yaitu bilangan.

2.3 Struktur Penseleksian If

Pernyataan if

Pernyataan *if* digunakan untuk menyeleksi 1 kondisi/keadaan dengan 1kriteria/persyaratan (*boolean_expression*). Kondisi (dapat berupa blok kode program) akan dieksekusi jika dan hanya jika kriteria terpenuhi atau bernilai benar (*true*). Sintaks atau bentuk penulisan pernyataan *if* adalah sebagai berikut :

if(boolean_expression) statement;

kondisi

atau

if(boolean_expression){statement1;

statement2; kondisi

. . . ;

}

Keterangan :

boolean_expression merupakan kriteria dari suatu kondisi berupa operatorpembanding.

Pernyataan *if-else*

Pernyataan *if-else* digunakan untuk menyeleksi 2 kondisi/keadaan dengan 1kriteria/persyaratan. Apabila kriteria pertama terpenuhi atau bernilai benar (*true*) maka kondisi ke-1 akan dieksekusi. Namun, jika kriteria pertama tidak terpenuhi atau bernilai salah (*false*) maka secara otomatis kondisi ke-2 akan dieksekusi. Sintaks atau bentuk penulisan pernyataan *if-else* adalah sebagai berikut :

if(boolean_expression) statement;

kondisi ke-1

else

statement;

kondisi ke-2

atau

if(boolean_expression) {statement1;

statement2;

kondisi ke-1

…;

} else {

statement3;

statement4;

kondisi ke-2

…;

}

Contoh :

```
short pwd = 123;

if (pwd == 123)
    System.out.println("Password Anda Benar!");
 else
    System.out.println("Password Anda Salah!");
```

```
run:
Password Anda Benar!
```

Pembahasan:

Operator pembanding yang digunakan sebagai kriteria adalah sama dengan (==). Nilai variabel '*pwd*' adalah 123 sehingga hasil operator pembanding menjadi 123==123 (dibaca 'benarkah 123 sama dengan 123') dan hasilnya adalah benar. Oleh karenahasil dari operator pembanding bernilai benar atau kriteria terpenuhi maka programakan memilih kondisi ke-1. Apabila hasil operator pembanding bernilai salah atau kriteria tidak terpenuhi maka program akan memilih kondisi ke-2.

Pernyataan *if-else-if*

Pernyataan *if-else if* dapat digunakan untuk menyeleksi beberapa kondisi (2 ataulebih) dengan beberapa kriteria (2 atau lebih) pula. Kondisi struktur seperti ini mengijinkan kita untuk membuat seleksi kondisi yang lebih kompleks. Sintaks atau bentuk penulisan pernyataan *if-else if* adalah sebagai berikut :

if(boolean_expression1){

statement1;statement2;

Kondisi ke-1

} else if(boolean_expression2) {

statement3;statement4;

Kondisi ke-2

} else {

statement5;statement6;

Kondisi ke-3

}

Keterangan :

Kondisi ke-3 akan dieksekusi secara otomatisapabila seluruh kriteria (*boolean_expression*) sebelumnya tidak terpenuhi.

Ada 3 kriteria yang digunakan pada contoh di atas dengan operator pembanding yang berbeda-beda. Program akan memulai seleksi dari pernyataan *if* pertama atau kriteria- 1.

Kriteria-1 menggunakan operator sama dengan (==) yaitu angka == 5 (dibaca '*benarkah 8 = 5*') dan hasilnya adalah salah (*false*) sehingga kriteria-1 tidakterpenuhi. Oleh karena itu, program akan melanjutkan proses seleksi ke pernyataan *if* kedua atau kriteria-2.

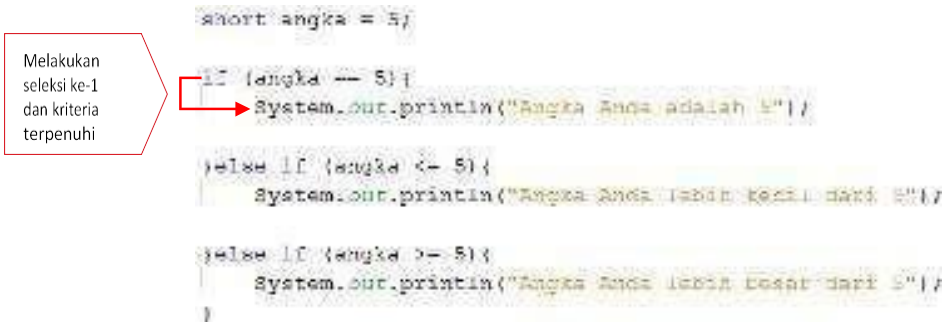
Kriteria-2 menggunakan operator lebih kecil sama dengan (<=) yaitu angka <=5 (dibaca '*benarkah 8<=5*') dan hasilnya adalah salah (*false*) sehingga kriteria- 2 tidak terpenuhi. Oleh karena itu, program akan melanjutkan proses seleksi ke pernyataan *if* ketiga atau kriteria-3.

```
run:
Angka Anda lebih besar dari 5
```

Kriteria-3 menggunakan operator lebih besar sama dengan (>=) yaitu angka

>= 5 (dibaca '*benarkah 8>=5*') dan hasilnya adalah benar (*true*) atau kriteria-3 terpenuhi. Oleh karena itu, program akan memilih kondisi ke 3 dan menghentikan proses seleksi.

Contoh 2:




```
short angka = 5;

if (angka == 5){
    System.out.println("Angka Anda adalah 5");

}else if (angka <= 5){
    System.out.println("Angka Anda lebih kecil dari 5");

}else if (angka >= 5){
    System.out.println("Angka Anda lebih besar dari 5");
}
```

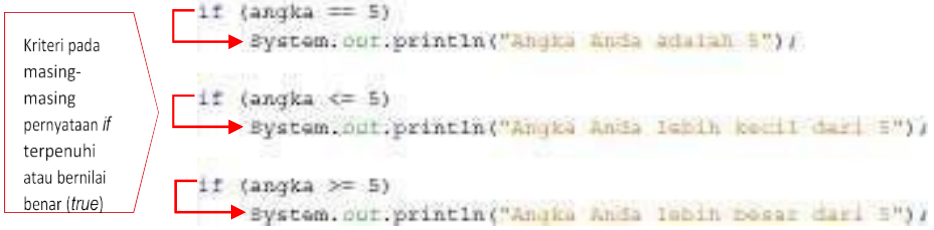
```
run:
Angka Anda adalah 5
```

Pembahasan :

Operator pembanding pada seleksi ke-1 bernilai benar (*true*) atau kriteria terpenuhi.Oleh karena itu, program akan memilih kondisi ke 1 dan menghentikan proses seleksi.

Pada kedua contoh di atas, pernyataan *if* yang digunakan merupakan satu struktur bangunan *if*. Hal ini berarti bahwa apabila proses seleksi pada kriteria pertama telah terpenuhi (operator pembanding bernilai benar/*true*) maka proses seleksi pada kriteri-kriteria berikut akan dihentikan (proses seleksi telah selesai).

Contoh 3:




```
short angka = 5;

if (angka == 5)
    System.out.println("Angka Anda adalah 5");

if (angka <= 5)
    System.out.println("Angka Anda lebih kecil dari 5");

if (angka >= 5)
    System.out.println("Angka Anda lebih besar dari 5");
```

```
run:
Angka Anda adalah 5
Angka Anda lebih kecil dari 5
Angka Anda lebih besar dari 5
```

Pembahasan :

Pada contoh 3 di atas, terdapat 3 pernyataan if dengan struktur bangunan yang berdiri sendiri-sendiri, sehingga proses seleksi akan terjadi pada setiap pernyataan *if* atauterjadi 3x proses seleksi.

Pernyataan *switch*

Cara lain untuk membuat cabang seleksi adalah dengan menggunakanpernyataan *switch*. Pernyataan *switch* mengkonstruksikan cabang seleksi untuk beberapa kondisi dengan beberapa kriteria. Berbeda dengan pernyataan *if*, pernyataan switch hanya menggunakan operator pembanding sama dengan (==) saja dan tidakdapat menggunakan operator logika. Sintaks atau bentuk penulisan pernyataan switch adalah sebagai berikut :

switch(switch_expression){case case_selector1:

statement1;

statement2; kondisi ke-1

...;

break; menghentikan

proses seleksi/keluar dari switch

case case_selector2:statement3;

statement4;

kondisi ke-2

. . . ;

break;

case case_selector-nstatemen-n;

…..;

break;

default: memilih kondisi

secara otomatis apabila kriteria sebelumnya tidak

statement1; terpenuhi

statement2;

. . . ;

break;

}

Contoh :

```
    case 2:
        menu="Nasi Gurame Asam Manis";
        harga=35000;
        break;

    case 3:
        menu="Nasi Soup Iga";
        harga=35000;
        break;

    default:
        menu="Tidak Ada";
        harga=0;
        break;
}
System.out.println("Pilihan: " + pilihan);
System.out.println("Menu: " + menu);
System.out.println("Harga: " + harga);
```

```
run:
Pilihan: 2
Menu: Nasi Gurame Asam Manis
Harga: 35000
```

Penseleksian Bersarang

Penseleksian bersarang merupakan proses seleksi suatu kondisi yang berlapis atau penseleksian didalam penseleksian. Untuk membuat penseleksian bersarang dapat menggunakan pernyataan seleksi ya ng sejenis (*if* didalam *if* atau *switch* didalam *switch*) atau kombinasi diantara keduanya (*if* didalam *switch* atau sebaliknya).

Contoh 1 :

Konversi nilai akhir menjadi nilai huruf dengan ketentuan sebagai berikut :

Tabel 2.10 Fase Sebuah Pemrograman Java

Nilai Akhir		*Nilai Huruf*
80	100	A
65	79	B
55	64	C
45	54	D
0	44	E

dapat disajikan dalam kode program sebagai berikut ini :

Kriteria ke-1

Kriteria ke-2

Kriteria ke-3

if bersarang (*nested if*)

Kriteria ke-4

Kriteria ke-5

```
run:
Nilai Akhir = 70
Nilai Huruf = B
```

Pembahasan :

Contoh di atas menggunakan *if* bersarang (*nested if*) yaitu *if* didalam *if*. Seleksi variabel *NA = 70* memenuhi kriteria *if* yang kedua yaitu else if(NA>=65) dan juga memenuhi kriteria *if* yang ada didalamnya yaitu if(NA<=79) sehingga nilai huruf yang didapat adalah 'B'. Oleh karena itu, untuk mendapatkan nilai huruf maka kedua kriterianya harus terpenuhi atau bernilai benar (*true*).

Kode program di atas dapat disederhanakan menggunakan operator logika and seperti berikut ini.

```
short NA = 70;
char NH = '-';

if(NA>=80 && NA<=100){
        NH='A';

}else if(NA>=65 && NA<=79){
        NH='B';

}else if(NA>=55 && NA<=64){
        NH='C';

}else if(NA>=45 && NA<=54){
        NH='D';

}else if(NA>=0 && NA<=44){
        NH='E';
}

System.out.println("Nilai Akhir = " + NA);
System.out.println("Nilai Huruf = " + NH);
```

Pembahasan :

Kode program ini menggunakan operator pembanding dan operator logika '*and*' (&&) secara bersamaan. Nilai variabel *NA = 70*, sesuai dengan kriteria kedua karena memenuhi persyaratan operator '*and*' (&&) (*NA>=65 && NA<=79*) dimana nilai operankiri bernilai bernilai *true* (*70>=65* ⟧ *true*) dan nilai operan kanan bernilai *true* (*70<=79* ⟧ *true*).

2.4 Merancang Apikasi Awal

Masukkan komponen 'Button' kedalam form. Pada jendela 'Palette' pilih kategori 'Swing Controls', kemudian klik kiri 'Button'. Arahkan pointer mouse kedalamform untuk menentukan posisi 'Button', kemudian klik kiri sekali lagi.

Catatan!

Komponen 'Button' (tombol) merupakan komponen eksekutor yang digunakan untuk menjalankan suatu proses tertentu pada aplikasi yang dibangun dengan cara klik kiri.

Pada jendela 'Navigator', muncul komponen 'Button' yang dimasukkan kedalam

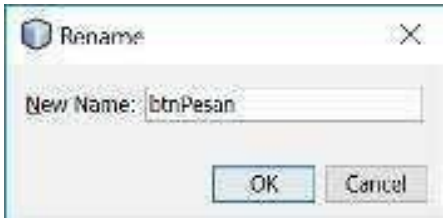
form dengan nama standar 'jButton1'.

Setiap komponen didalam form memiliki nama (*variable name*) untuk membedakan komponen satu dengan komponen lainnya dan memudahkan programer untuk mengoperasikan komponen melalui kode program.

Untuk mengubah nama variabel dari komponen, klik kanan 'jButton1' pada

jendela navigator kemudian klik 'Change Variable Name…'

Ubah menjadi 'btnPesan'. Klik tombol OK.



Gambar 2.4 Arsitektur Teknologi Java

Nama variabel komponen akan berubah pada jendela navigator

Pemberian nama variabel pada komponen bermanfaat untuk memudahkanprogramer dalam :

Mengenali dan membedakan setiap komponen yang digunakan.

Menulis kode program didalam suatu komponen.

Memanipulasi data yang dimasukkan dari komponen atau untuk ditampilkanke komponen melalui kode program.

Untuk mengubah teks button, klik kanan 'btnPesan' pada jendela navigator atauklik kanan komponen pada form, kemudian klik 'Edit Text'.

Ubah menjadi 'Pesan' kemudian tekan enter.

Double click button 'Pesan' pada form sehingga muncul tampilan kode program

untuk menulis kode program pada button 'btnPesan'

Tuliskan kode program berikut:

```
private void btnPesanActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    javax.swing.JOptionPane.showMessageDialog(null,"Selamat Belajar Pemrograman Java");
}
```

Untuk menjalankan program, tekan tombol ‘shift + F6’ pada keyboard atau

Pada program yang berjalan (runtime) klik tombol ‘Pesan’, kemudian munculkotak pesan yang berisi ‘Selamat Belajar Pemrograman Java’.

Gambar 2.8 Arsitektur Teknologi Java

BAB 3

KONSEP OOP

3.1 Percabangan Dan Perulangan

Di Java, percabangan dapat diimplementasikan dengan dua jenis pernyataan:

pernyataan if dan pernyataan switch-case. Keduanya memiliki fungsi yang sama, tetapi ada sedikit perbedaan.

Jika pernyataan

Anda dapat mengimplementasikan pernyataan if untuk menangani percabangan berdasarkan satu, dua, tiga atau lebih kondisi. Jika Anda tidak memiliki banyak kondisi, gunakan pernyataan if.

Conditional if form adalah bentuk paling sederhana yang berisi pernyataan tunggal yang akan dieksekusi jika kondisi terpenuhi. Sintaks dasarnya adalah:

```
if (kondisi){
 statement1;
 statement2;
 }
```

a. Struktur For

Setiap komponen didalam form memiliki nama (*variable name*) untuk membedakan komponen satu dengan komponen lainnya dan memudahkan programer untuk mengoperasikan komponen melalui kode program.

Untuk mengubah nama variabel dari komponen, klik kanan ‘jButton1’ pada

jendela navigator kemudian klik ‘Change Variable Name…’

Ubah menjadi ‘btnPesan’. Klik tombol OK.

Nama variabel komponen akan berubah pada jendela navigator

Pemberian nama variabel pada komponen bermanfaat untuk memudahkanprogramer dalam :

Mengenali dan membedakan setiap komponen yang digunakan.

Menulis kode program didalam suatu komponen.

Memanipulasi data yang dimasukkan dari komponen atau untuk ditampilkanke komponen melalui kode program.

Untuk mengubah teks button, klik kanan ‘btnPesan’ pada jendela navigator atauklik kanan komponen pada form, kemudian klik ‘Edit Text’.

Ubah menjadi ‘Pesan’ kemudian tekan enter.

Double click button ‘Pesan’ pada form sehingga muncul tampilan kode program

untuk menulis kode program pada button ‘btnPesan’

Tuliskan kode program berikut:

Untuk menjalankan program, tekan tombol ‘shift + F6’ pada keyboard atau

tombol run project pada toolbar.

Pada program yang berjalan (runtime) klik tombol ‘Pesan’, kemudian munculkotak pesan yang berisi ‘Selamat Belajar Pemrograman Java’.

Pembahasan :

Tahapan proses eksekusi kode program pada contoh perulangan *while* di atas,digambarkan pada ilustrasi di bawah ini.

m = 1;

if (m<=10){

System.out.print(m + “ “);

m+=2;

}

Tahap 1, menentukan nilai awal dari variabel *m*.

Tahap 2, melakukan proses seleksi dengan kriteria/syarat *m<=10* (operator pembanding lebih kecil sama dengan). Apabila kriteria terpenuhi atau bernilaibenar (*true*) maka eksekusi kode program akan dilanjutkan ke tahap 3.

Tahap 3, menjalankan eksekusi pernyataan atau blok kode program didalamperulangan *while*.

Tahap 4, melakukan proses iterasi dengan operator penugasan (*assignment*).

Selanjutnya proses eksekusi akan diulang kembali dari tahap 2 dan dilanjutkanlagi ke tahap-tahap berikut.

Putaran eksekusi dalam perulangan tetap terjadi selama kriteria terpenuhi ataubernilai benar (*true*).

Apabila kriteria tidak terpenuhi maka eksekusi perulangan akan berhenti(selesai).

Struktur do…while

Seperti halnya perulangan *while*, perulangan ini juga akan bekerja selama (*while*) kriteria kondisi terpenuhi atau bernilai benar (*true*). Namun, berbeda dengan 2 struktur perulangan sebelumnya (*for* dan *while*), *do…while* akan melakukan eksekusi pernyataan terlebih dahulu (*do*) sebelum melakukan seleksi kriterianya (*while*). Berikut adalah sintaks atau bentuk penulisan perulangan *do…while*

do{

statement1; statement2; statement-n; proses_iterasi

}while(kriteria_kondisi);

Contoh :

```
int p=2;
do {
    System.out.print(p + " ");
    p = p + 2;

} while (p <= 10);
run:
2 4 6 8 10
```

Pembahasan :

Tahapan proses eksekusi kode program pada contoh perulangan *do…while* di atas,digambarkan pada ilustrasi di bawah ini.

p = 2;

System.out.print(p + " ");

p = p + 2;

if (p<=10){

<kriteria terpenuhi>

} else {

<kriteria tidak terpenuhi>

}

Tahap 1, menentukan nilai awal dari variabel *p*.

Tahap 2, menjalankan eksekusi pernyataan atau blok kode program didalamperulangan *do…while*.

Tahap 3, melakukan proses iterasi dengan operator penugasan (*assignment*).

Tahap 4, melakukan proses seleksi dengan kriteria/syarat *p<=10* (operatorpembanding lebih kecil sama dengan).

Apabila kriteria terpenuhi atau bernilai benar (*true*) maka eksekusi kode programakan diulang kembali dari tahap 2 dan dilanjutkan lagi ke tahap-tahap berikut.

Putaran eksekusi dalam perulangan tetap terjadi selama kriteria terpenuhi ataubernilai benar (*true*).

Apabila kriteria tidak terpenuhi maka eksekusi perulangan akan berhenti(selesai).

Perulangan Bersarang

Perulangan bersarang adalah penggunaan perulangan didalam perulangan. Perulangan yang digunakan dapat berupa perulangan sejenis atau kombinasi jenisperulangan satu dengan jenis perulangan yang lainnya.

Contoh :

```
short n;
short m;
for (n=1; n<=2; n++){
    m=n;
    while (m<=10) {
        System.out.print(m + " ")
        m+=2;
    }
    System.out.println("");
}

run:
1 3 5 7 9
2 4 6 8 10
```

Pembahasan :

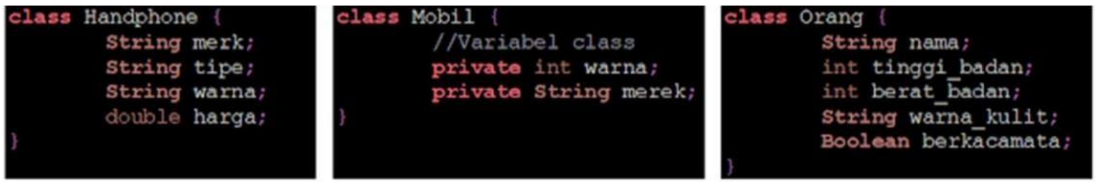
Contoh di atas menggunakan 2 buah jenis perulangan yang berbeda yaitu perulangan *while* didalam perulangan *for*. Dalam putaran perulangan *for* yang pertama, perulangan while akan menyelesaikan putaran proses eksekusinya sebelum melanjutkan ke putaran perulangan *for* berikutnya.

3.2 Kelas

Bahasa Java adalah bahasa pemrograman berorientasi objek, sehingga konsep objek dan kelas menjadi penting. Ada banyak objek di dunia nyata, seperti objek manusia, objek mobil, dan objek pohon. Bagaimana cara memindahkan objek yang ada di dunia nyata ke objek dalam pemrograman khususnya di Java?

Misalnya, objek dunia nyata adalah objek manusia. Anda dapat mengetahui karakteristik suatu objek: tinggi, berat, warna rambut, warna kulit, jenis kelamin, pemakai kacamata, dll. Dalam pemrograman, objek dunia nyata adalah CLASSES, dan 35ttribut objek adalah variabel kelas yang disebut DATA MEMBER.

Berdasarkan contoh di atas, definisi pemrograman adalah:



```
class Handphone {
        String merk;
        String tipe;
        String warna;
        double harga;
}
```

```
class Mobil {
        //Variabel class
        private int warna;
        private String merek;
}
```

```
class Orang {
        String nama;
        int tinggi_badan;
        int berat_badan;
        String warna_kulit;
        Boolean berkacamata;
}
```

Gambar 3.1 Arsitektur Teknologi Java

Struktur pembuatan class, adalah sebagai berikut:

class NamaClass{

//Isi Class

}

Keterangan:
Nama_Kelas harus sesuai dengan nama file.

Contoh: class Handphone, maka nama filenya harus diberi nama dengan Handphone.java.

B. Atribut

Atribut merupakan ciri-ciri yang melekat pada suatu objek.

Berikut adalah contoh syntax atribut.

[access_modifier] [tipe_data] [nama_variabel] = [value];

informasi:

[access_modifier] digunakan untuk membatasi hak akses untuk kelas dan metode. Pengubah akses dijelaskan di subbagian berikutnya

Tipe data menunjukkan apakah variabel bertipe string, int, double, dll.

[nama variabel] adalah nama (definisi) dari variabel

[nilai] adalah nilai dari variabel

contoh:

warna string pribadi = “merah”;

C. Metode

“Metode adalah tindakan yang dapat dilakukan oleh CLASS. Metode adalah fungsi yang digunakan untuk memanipulasi nilai atribut atau melakukan sesuatu yang dapat dilakukan pada objek itu sendiri.

Dalam hal ini, metode berisi paket It can berisi sekumpulan program. Karena suatu metode dapat memanggil sekumpulan program hanya dengan memanggil nama metode, tidak ada pemborosan dalam menulis program. Selain itu, program menjadi lebih terstruktur, praktis dan efisien. Contoh :

METHOD Class Handphone	METHOD Class Mobil	Method Class Orang
- Memasukan warna - Memasukkan tipe - Memasukkan merek	- memasukan data mobil - gerak mobil belok kiri - menampilkan informasi	- memasukkan data orang - tertawa - menampilkan informasi

Gambar 3.2 Arsitektur Teknologi Java

Method yang mengembalikan nilai biasanya berupa sub program berjenis fungsi. Sedangkan method yang tidak mengembalikan nilai biasanya berupa sub program berjenis prosedur. Berikut adalah contoh syntax pembuatan method.

[access_modifier] [tipe_data] nama_method(.....)

informasi:

[access_modifier] digunakan untuk membatasi hak akses untuk kelas dan metode. Pengubah akses dijelaskan di subbagian berikutnya

Tipe data menunjukkan apakah variabel bertipe string, int, double, dll.

[nama_metode] adalah nama (definisi) metode. Secara umum, metode selalu diakhiri dengan tanda kurung ().

(.....) berisi parameter sesuai kebutuhan.

Contoh

```
public void masuk info mobil (int mrk , String nm){
 this.merek = mrk ;
```

Implementasi method dalam pemrograman adalah:

```
public void menangis ()
{
public void tertawa ()
{
```

D. Objek

Dalam pemrograman, suatu group bisa memiliki banyak objek & objek akanmewarisi data member & method yg sama berdasarkan suatu group. Objek diklaim juga instance of Class adalah objek yg diinstan atau dibentuk berdasarkan group.

Untuk menciptakan obyek berdasarkan group Orang, digunakan keyword 'new'. Contoh :

Orang org1 = new Orang ("Samuel") ;

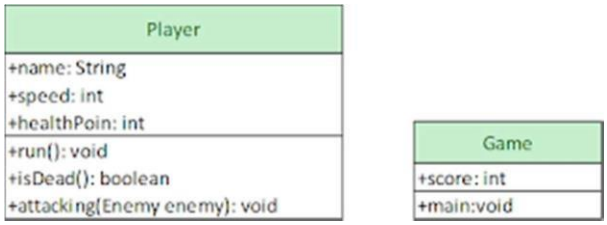
Orang org2 = new Orang ("Ari");"

Sehingga class Orang, saat ini mempunyai 2 obyek.

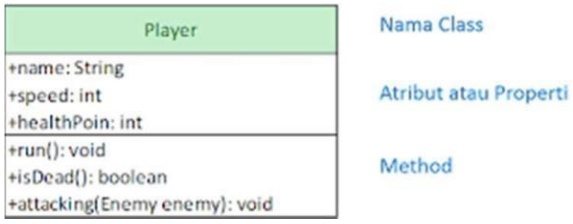
Kata Kunci 'this' kata kunci bdigunakan untuk membedakan variabel yang dideklarasikan pada parameter di dalam method dengan variabel yang dideklarasikan pada class. Untuk penggunaan bdapat anda lihat pada soallatihan.

F. Class Diagram

Class Diagram adalah sebuah diagram yang menggambarkan hubungan aplikasi perancangan (CASE),seperti StarUML



Sebuah class digambarkan dengan sebuah 38ttri 1 kolom dan 3 baris. Baris pertama berisi nama class; Baris kedua berisi atribut; dan Baris ketiga berisi method.



itu, terdapat garis yang menggambarkan hubungan antar class.

- ✓ Association
- ✓ Directed asociation
- ✓ Aggregation

- ✓ Compositon
- ✓ Dependecy
- ✓ Generalization
- ✓ Interface realization

Class Diagram biasanya digunakan oleh software engineer untuk merancang software denganparadigma OOP.

c. Membuat method setter

Setelah membuat variabel yang d ibutuhkan pada class Handphone, langkah selanjutnya anda buat method setter untuk masi ng-masi ng variabel. Hal in i bertujuan untuk mengeset nilai yang diperoleh dari class Utama yang nantinya akan kita gunakan ke dalam class Handphone. Perlu diketahui pula, bahwa dalam pembuatan variabel pada parameter di method setter, harus sesuai dengan tipe data pada variabel yang dideklarasikan di kelas.

Gambar di bawah ini menunjukkan deklarasi setter.

Code

```
class Handphone
{
 //deklarasi
 private String merk, tipe , warna;
 private double harga;

 //setter
 public void setMerk(String merk)
 {
  this.merk=merk;
 }
 public void setTipe(String tipe)
 {
  this.tipe=tipe;
 }
 public void setWarna(String colour)
 {
```

```
 warna=colour;
}
public void setHarga(double harga)
{
 this.harga=harga;
}
//getter
 //method  tambahan
}
```

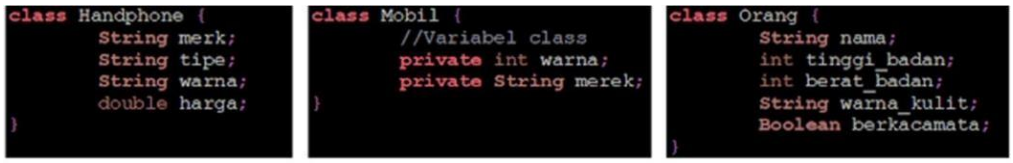
```
class Handphone
{
    //deklarasi
    private String merk, tipe , warna;
    private double harga;

    //setter
    public void setMerk(String merk)
    {
        this merk=merk;
    }
    public voiid setTipe(String tipe)
    {
        this.tipe=tipe;
    }
    public void setWarna(String colour)
    {
        warna=colour;
    }
    public void setHarga(double harga)
    {
        this harga=harga;
    }

    //getter

    //method tambahan
}
```

A. Kelas (class)

Bahasa Java merupakan bahasa pemrograman yang berorientasi obyek sehingga konsep obyek dan kelas (class) menjadi penting. Dalam dunia nyata, ada banyak obyek misalnya obyek orang, obyek mobil, obyek pohon dsb. Bagaimana memindahkan obyek yang ada dalam dunia nyata menjadi obyek dalam pemrograman khususnya Java ?

Misalkan obyek dalam dunia nyata adalah obyek orang. Obyek tersebut dapat diceritakan tentang ciri-cirinya, yaitu tinggi badan, berat badan, warna rambut, warna kulit, jenis kelami, menggunakan kacamata dll. Dalam pemrograman, obyek didunia nyata akan menjadi CLASS, dan ciri-ciri obyek akan menjadi variabel class yang disebut sebagai DATA MEMBER.

Berdasarkan contoh diatas, pendefinisian pada pemrograman adalah :



```
class Handphone {
        String merk;
        String tipe;
        String warna;
        double harga;
}
```

```
class Mobil {
        //Variabel class
        private int warna;
        private String merek;
}
```

```
class Orang {
        String nama;
        int tinggi_badan;
        int berat_badan;
        String warna_kulit;
        Boolean berkacamata;
}
```

Struktur pembuatan class, adalah sebagai berikut:

class NamaClass{

//Isi Class

}

Keterangan:
Nama_Kelas harus sesuai dengan nama file.

Contoh: class Handphone, maka nama filenya harus diberi nama dengan Handphone.java.

B. Atribut

Atribut merupakan ciri-ciri yang melekat pada suatu objek. Berikut adalah contoh syntax atribut.

[access_modifier] [tipe_data] [nama_variabel] = [value];

Keterangan:

[access_modifier] digunakan untuk memberi batasan hak class maupun *method*. Access modifier akan dijelaskan pada sub bab berikutn ya

[tipe_data] menjelaskan apakah variabel tersebut berti pe String, int, double, dan sebagainya

[nama_variabel] merupakan sebutan (definisi) variabel tersebut

[value] merupakan nilai dari variable tersebut

Contoh: private String warna = “merah”;

C. Method

“Method merupakan tindakan aksi yang bisa dikerjakan oleh CLASS. Method merupakan fungsi-fungsi yang digunakan untuk memaipulasi nilai-nilai pada atribut dan atau untuk melakukan hal-hal yang dapat dilakukan oleh objek itu sendiri. Dalam hal ini method dapat berisi sekumpulan program yang telah terbungkus. Dengan method, kita bisa memanggil kumpulan program tersebut hanya dengan memanggil nama methodnya sehingga pekerjaan jadi lebih singkat dan tidak boros menuliskan program. Selain itu, program menjadi lebih terstruktur, praktis, dan efisien. Contoh

:

METHOD Class Handphone	METHOD Class Mobil	Method Class Orang
- Memasukan warna - Memasukkan tipe - Memasukkan merek	- memasukan data mobil - gerak mobil belok kiri - menampilkan informasi	- memasukkan data orang - tertawa - menampilkan informasi

Method yang mengembalikan nilai biasanya berupa sub program berjenis fungsi. Sedangkan method yang tidak mengembalikan nilai biasanya berupa sub program berjenis prosedur. Berikut adalah contoh syntax pembuatan method.

[access_modifier] [tipe_data] nama_method(.....)

Keterangan:

[access_modifier] digunakan untuk memberi batasan hak class maupun method. Access modifier akan dijelaskan pada sub bab berikutnya

[tipe_data] menjelaskan apakah variabel tersebut bertipe String, int, double, dan sebagainya

[nama_method] merupakan sebutan (definisi) method tersebut. Umumnya method selalu diakhiri dengan tanda kurung ()

(.....) berisi parameter apabi la d iperl ukan.

Contoh
public void masuk info mobil (int mrk , String nm){

this.merek = mrk ;

Implementasi method dalam pemrograman adalah:

public void menangis ()

{

public void tertawa ()

{

D. Objek

Dalam pemrograman, suatu class dapat mempunyai banyak objek dan objek akan mewarisi data member dan method yang sama dari suatu class. Objek disebut juga instance of Class merupakan objek yang diinstan atau dibuat dari class.

Untuk membuat obyek dari class Orang, digunakan keyword ‘new’. Contoh :

Orang org1 = new Orang (“Samuel”) ;

Orang org2 = new Orang (“Ari”);”

Sehingga class Orang, saat ini mempunyai 2 obyek.

E. Kata Kunci ‘this’

kata kunci bdigunakan untuk membedakan variabel yang dideklarasikan pada parameter di dalam method dengan variabel yang dideklarasikan pada class. Untuk penggunaan bdapat anda lihat pada soallatihan.

F. Class Diagram

Class Diagram adalah sebuah diagram yang menggambarkan hubungan aplikasi perancangan (CASE),seperti StarUML.

Sebuah class digambarkan dengan sebuah 43ttri 1 kolom dan 3 baris. Baris pertama berisi nama class; Baris kedua berisi atribut; dan Baris ketiga berisi method.

itu, terdapat garis yang menggambarkan hubungan antar class.

- ✓ Asociation

- ✓ Directed asociation
- ✓ Aggregation
- ✓ Compositon
- ✓ Dependecy
- ✓ Generalization
- ✓ Interface realization

Class Diagram biasanya digunakan oleh software engineer untuk merancang software denganparadigma OOP.

c. Membuat method setter

Setelah membuat variabel yang d ibutuhkan pada class Handphone, langkah selanjutnya anda buat method setter untuk masi ng-masi ng variabel. Hal in i bertujuan untuk mengeset nilai yang diperoleh dari class Utama yang nantinya akan kita gunakan ke dalam class Handphone. Perlu diketahui pula, bahwa dalam pembuatan variabel pada parameter di method setter, harus sesuai dengan tipe data pada variabel yang dideklarasikan di kelas.

Gambar di bawah ini menunjukkan deklarasi setter.

Code

```
class Handphone
{
 //deklarasi
 private String merk, tipe , warna;
 private double harga;
  //setter
 public void setMerk(String merk)
 {
  this.merk=merk;
 }
 public void setTipe(String tipe)
 {
  this.tipe=tipe;
 }
```

```
public void setWarna(String colour)
{
 warna=colour;
}
public void setHarga(double harga)
{
 this.harga=harga;
}
 //getter
 //method tambahan
}
```

```
1  class Handphone
2  {
3      //deklarasi
4      private String merk, tipe , warna;
5      private double harga;
6
7      //setter
8      public void setMerk(String merk)
9      {
10         this merk=merk;
11     }
12     public voiid setTipe(String tipe)
13     {
14         this.tipe=tipe;
15     }
16     public void setWarna(String colour)
17     {
18         warna=colour;
19     }
20     public void setHarga(double harga)
21     {
22         this harga=harga;
23     }
24
25     //getter
26
27     //method tambahan
28 }
```

Sebagai tambahan informasi, dalam pembuatan method setter,kita menggunakan sub program berjenis prosedur. Hal ini dikarenakan data yang akan kita set, tidak terdapat umpan balik ke dalam program. Hand phone (lihat script yang diberi kotak berwarna biru). Apabila variabel tersebut tersebut tidak diberi keyword this, maka variabel tersebut akan mengacu kepada variabel yang dideklarasikan pada parameter method setter (lihat script yang diberi kotak berwarna hijau). Anda bisa menggunakan keyword this atau tidak apabila ada perbedaan deklarasi nama variabel pada class Handphone dengan parameter pada method setter (lihat script yang diberi kotak berwarna ungu).”

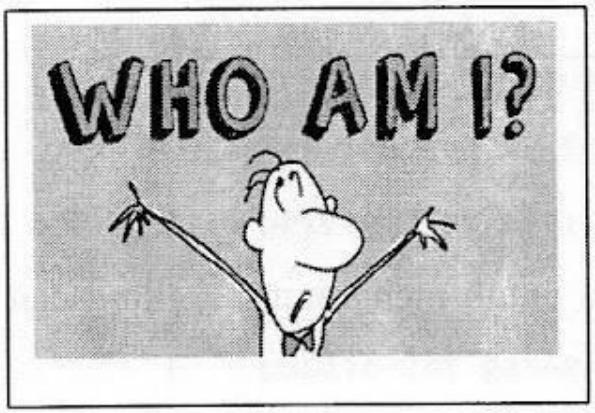
3.3 Enkapsulasi

Encapsulation adalah salah satu dari empat konsep OOP fundamental. Tiga lainnya adalah pewarisan, polimorfisme, dan abstraksi. Enkapsulasi adalah mekanisme membungkus data (variabel) 46ttribut yang bekerja pada data (46ttrib) yang bersama-sama sebagai satu kesatuan. Didalam enkapsulasi, variabel kelas akan disembunyikan dari kelas-kelas lain, dan dapat diakses hanya melalui method kelas itu sendiri. Oleh karena itu, juga dikenal sebagai persembunyian data.

```
public void setPanjang(int pj){
 this.panjang=pj;
}
public void setLebar(int lb){
 this.lebar=lb ;
}
public int getPanjang(){
 return this.panjang;
}
public int getLebar(){
 return this.lebar;
}
public int luas(){
 return this.panjang*this.lebar;
}
}
```

public *setXXX()* dan *getXXX()* adalah method akses dari variabel instance dari kelas EncapTest. Biasanya, method ini disebut sebagai getter dan setter. Oleh karena itu, setiap kelas yang ingin mengakses variabel harus mengaksesnya melalui getter dan setter.

3.4 Constructor



Gambar 1.11 Arsitektur Teknologi Java

Semua manusia membutuhkan nama. Sadar atau tidak, perkenalkan diri Anda dengan nama, alamat, tanggal lahir, hobi, dll. Namun pernahkah Anda bertanya-tanya mengapa begitu banyak orang lebih suka mengingat nama mereka daripada alamat, tanggal lahir, hobi, dll. Mungkinkah nama itu juga digunakan oleh banyak orang lain? Meski dieja berbeda, nama lengkap saya juga digunakan. Itu semua karena namamu unik. Bayangkan lagi, bagaimana jika Anda tidak memiliki nama, ketika orang lain memanggil Anda, itu pasti membingungkan. Seseorang tanpa nama kadang-kadang disebut “47ttrib”. Membuat konstruktor serupa. Saat Anda membuat objek kelas Manusia (misalkan “Orang 1”), Anda dapat menggunakan metode setter() untuk menetapkan nilai nama, alamat, tanggal lahir, dan hobi. Berbeda dengan desainer. Saat objek Person 1 dibuat, ia langsung memberikan nilai berupa nama, alamat, tanggal lahir, dan hobi. Seolah-olah Anda diberi nama segera setelah Anda lahir ke dunia ini. Ini adalah konsep konstruktor. A. Konstruktor

Konstruktor adalah metode yang digunakan untuk menginisialisasi variabel instan yang dimiliki oleh suatu objek. Sebuah konstruktor dipanggil ketika sebuah kelas diinstansiasi ke dalam sebuah objek. Jika suatu kelas tidak memiliki metode konstruktor, semua variabel objek diinisialisasi dengan nilai default sesuai dengan tipe datanya. Ini adalah struktur konstruktor.

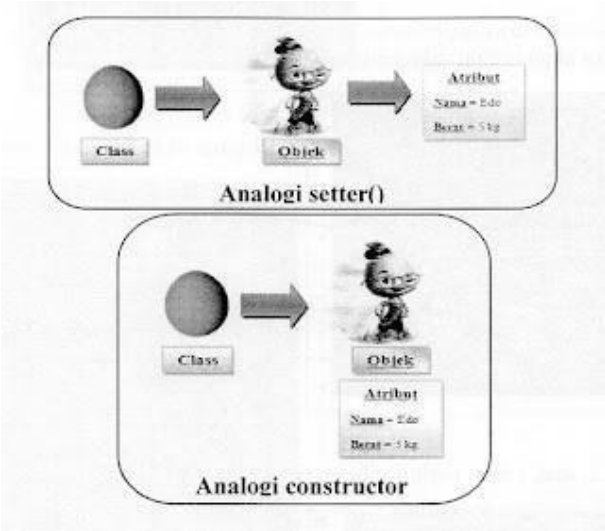
Membuat objek di atas akan menetapkan nilai default untuk nama pengguna dan kata sandi Dalam format admin dan 12345. Sementara itu, jika Anda membuat pengguna sendiri (mis. Usemame=edo, password=pb0) melalui instance kelas dalam metode login dengan parameter berikut:

E. Perbedaan antara menggunakan metode Konstruktor dan Setter()

Jika Anda melihat lebih dekat, tidak ada keraguan bahwa Anda akan bertanya pada diri sendiri, “Jadi apa bedanya?”

“Gunakan setter() dalam konstruktor?”. Metode setter() pertama-tama membuat objek dari kelas (instance dari kelas). Setelah sebuah objek dibangun, itu diberikan atribut. Berbeda dengan desainer.

Objek yang dibuat dari kelas (instance dari kelas) diberi atribut langsung di konstruktor. Ini adalah analogi untuk perbedaan antara setter() dan konstruktor.



Gambar 1.11 Arsitektur Teknologi Java

3.5 Pewarisan (Inheritance)

Seperti yang kita lihat di modul sebelumnya, kelas atau objek dapat dikaitkan dengan kelas lain. Salah satu bentuk hubungan adalah pewarisan. Hubungan ini seperti hubungan keluarga antara orang tua dan anak. Kelas Java dapat memiliki satu atau lebih turunan atau subkelas. Kelas anak mewarisi 48ttribut dan metode dari kelas induknya. Misalnya, dalam gim Anda membuat kelas musuh dengan perilaku berbeda.

Zombie
+name +hp: int +attackPoint: int
+attack(): void +walk(): void

Pocong
+name: string +hp: int +attackPoint: int
+attack(): void +jump(): void

Burung
+name: string +hp: int +attackPoint: int
+attack(): void +fly(): void +walk(): void +jump(): void

Kode untuk masing-masing kelas seperti ini:

File: Zombie.java

File: Pocong.java

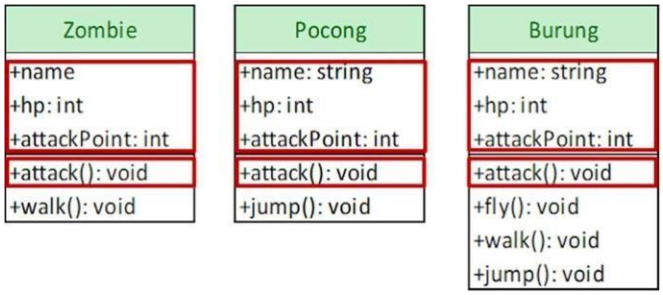
File: Burung .java

```
class Zombie{
        String name;
        int hp;
        int attackPoin;

        void attack(){
```

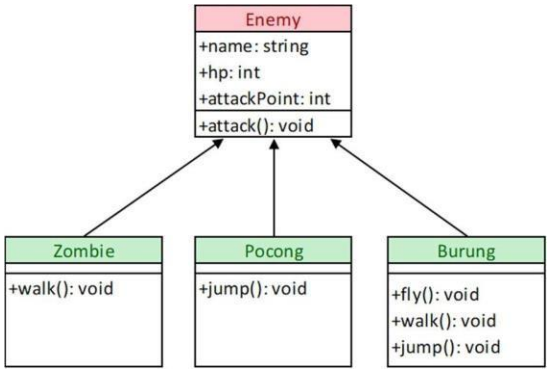
```
class Burung{
        String name;
        int hp;
        int attackPoin;

        void attack(){
                //...
        }

        void walk(){
                //...
        }
```

Penulisan kode seperti di atas, tidak salah, namun tidak efektif, karena 49ttribut dan method yang sama ditulis berulang-ulang. Solusinya adalah menggunakan inheritance. Berikut gambaran member class yang sama:

Setelah menggunakan inheritance, maka akan menjadi seperti ini:



Inheritance bisa digambarkan dengan garis hubung Generalization.



Class Enemy adalah class induk yang memiliki anak Zombie, Pocong, dan Burung. Apapun 49ttribut

yang ada di class induk, akan dimiliki juga oleh class anak. Lalu bagaimana bentuk kodenya dalam Java? Bentuk kodenya akan seperti ini:

File: Enemy.java

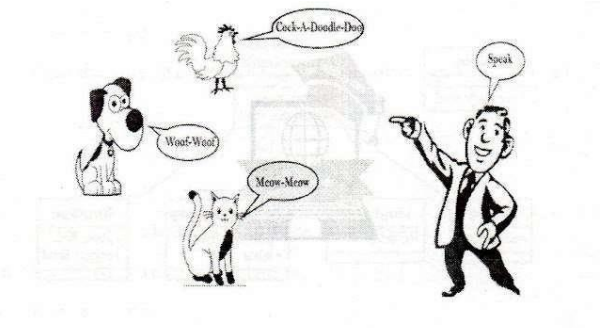
```
class Enemy{
String name;
int hp;
 int attackPoin;

 void attack(){
  System.out.println("Enemy Attack");
 }
}
```

Pada class anak, kita menggunakan kata kunci extends untuk menyatakan kalau dia adalah class turunan dari Enemy.

Output:

```
Enemy Attack
wewe
10000
500
manuk
20000
700
```

3.5 Polimorfismen (Polymorphism)



Gambar 1.11 Arsitektur Teknologi Java

Ada kata "bisa" dan kita akan menggunakannya dalam kalimat berikut.

Praktisi dapat memahami konsep polimorfisme dalam praktik pemrograman berorientasi objek

b. Ular memiliki racun yang sangat berbahaya

Dari dua kalimat berikut, apakah Anda memahami arti kata “bisa” pada kalimat sebelumnya? Apakah kata “mungkin” memiliki arti yang sama dalam dua kalimat?

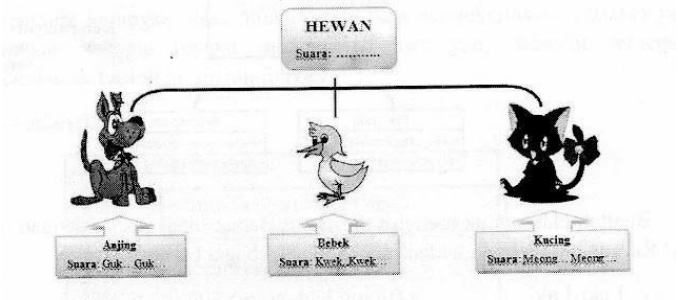
Kata “may” pada kalimat pertama berarti bisa atau bisa. Kata “bisa” pada kalimat kedua berarti racun. Dalam kalimat di atas, kata yang sama memiliki dua arti yang berbeda. Ini adalah salah satu konsep polimorfisme. Untuk memperjelas konsep polimorfisme, kita akan membahasnya pada subbab berikutnya.

Sebuah polimorfisme

Polimorfisme adalah variabel/metode kelas,

Dipanggil kembali ke kelas turunan dengan perilaku yang berbeda antara setiap kelas.

Lihat contoh gambar di bawah.



Gambar 1.11 Arsitektur Teknologi Java

Setiap hewan memiliki suara. Namun, hewan yang berbeda membuat panggilan yang berbeda. Perilaku yang berbeda inilah yang menjadi ciri polimorfisme. Penggunaan teknik polimorfisme dapat diidentifikasi dengan membuat sebuah instance dari kelas di kelas utama.

Seperti yang Anda lihat pada gambar di atas, deklarasi objek yang (segera) dibuat diambil dari kelas induknya, kelas Hewan. Jadi objek adalah array kelas

Penerapan konsep polimorfisme terlihat jelas dalam latihan.

Kriteria untuk menggunakan polimorfisme

Saat menggunakan polimorfisme, dua kriteria harus dipenuhi:

Eksekusi metode kelas turunan yang dipanggil dari objek kelas induk Nama metode yang digunakan di kelas induk harus ditulis ulang di kelas turunan (metode override). Namun, nama metode dan tipe data harus sama.

Mengganti metode

Metode overriding adalah proses mendeklarasikan ulang nama metode di kelas utama dari kelas turunan Saat membuat metode utama, nama metode dan tipe data harus sesuai dengan kelas induk untuk polimorfisme.Pelajari lebih lanjut akan dijelaskan dengan pertanyaan praktis .

3.6 Interface / Antar Muka

Ketika anda diberi tugas menerjemahkan sebuah buku berbahasa Inggris ke dalam bahasa Indonesia, namun ada beberapa kata dalam buku tersebut yang sangat asing bagi anda. Apa yang akan anda lakukan untuk memecahkan masalah tersebut?

Ketika anda meminjam sebuah buku dan anda ingin mengetahui informasi apa saja yang ada pada buku tersebut, apa yang akan anda lakukan?

Meminjam kamus adalah salah satu opsi pada Gambar 1. Mengapa Karena kamus menyimpan kumpulan kata (atau biasa disebut kosa kata) yang membantu Anda mengartikan kata.

Hal yang sama berlaku untuk Gambar 2. Jika jawaban Anda adalah dengan melihat daftar isi, itu berarti Anda memahami cara menggunakan daftar isi. Daftar Isi memiliki halaman untuk setiap bab yang memudahkan untuk menemukan informasi yang Anda cari. Kamus dan daftar isi adalah alat yang berguna untuk mengumpulkan dan memproses informasi. Mengembangkan informasi yang diterima. Mirip dengan menggunakan antarmuka dalam pemrograman berbasis objek. Antarmuka berisi sekumpulan deklarasi konstanta/metode tanpa menyertakan atau mendeskripsikan badan metode. Metode atau variabel yang terdapat dalam kelas antarmuka dapat digunakan di beberapa kelas dengan mendapatkan panggilan ke kelas antarmuka.

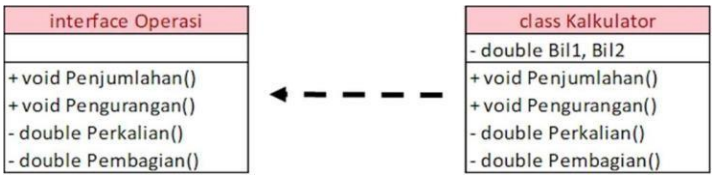
Antarmuka adalah kumpulan metode yang hanya berisi deklarasi dan struktur metode tanpa detail implementasi. Berikut cara mendeklarasikan antarmuka:

```
interface Nama_Interface
{
 //deklarasi variabel dan/ atau method
}
```

Dalam contoh di atas, metode yang dideklarasikan di antarmuka Operasi tidak berisi pernyataan apa pun, baik ekspresi atau hanya mengembalikan nilai. Ini karena antarmuka hanyalah kumpulan konstanta dan metode tanpa menyertakan atau menulis badan metode.

Perhatikan bahwa antarmuka bukan kelas dan kelas hanya dapat mengimplementasikan antarmuka (antarmuka bukan kelas dan kelas hanya dapat mengimplementasikan antarmuka). Jadi jangan berasumsi bahwa antarmuka sebenarnya adalah superclass dengan kelas yang sedang berjalan.

Anda dapat menggunakan skema OOP berikut untuk melihat penggunaan (implementasi) antarmuk di kelas.



Lihat panah ungu. Panah menunjukkan bahwa kelas Kalkulator adalah implementasi dari antarmuka Operasi, dan metode yang terdapat dalam antarmuka Operasi harus dideklarasikan ulang (ditimpa) di kelas Kalkulator. Antarmuka diwakili oleh panah putus-putus dan pewarisan oleh panah lurus ( ).

Dengan perangkat pemrograman. Jawa adalah:

OOP, Menerapkan Antarmuka Menggunakan Kata Kunci Bagaimana Mengimplementasikan Antarmuka Secara Terprogram

```
class Nama_Kelas implements Nama_Interface

{

 //isi kelas

}
```

BAB 4

PENGENALAN OOP

4.1 Pengenalan OOP

Pengenalan objek oriented programing merupakan metode pendekatan pemrograman atau paradigma dalam mengembangkan perangkat software computer dimana dalam struktur perangkat nya tersebut didasarkan pada.memiliki interaksi antar objek objek sehinga menjadi sebuah class.

Real word data > oop> aplikasi konsep pengembangan oop :

Clas

merupakan terdirinya beberapa objek yang saling berinteraksi menjadi objek-objek menjadi sebuah class dalam suatu pemrograman.

Property

deskirpsi atau atribut dari suatu objek dan menjadi identitas suatu class tertentu

Method

adalah aksi-aksi dalam melakukan terhadap sebuah objek .

pemahaman dasar program objek oriented murni :

di dalam struktur program tersebut memiliki class dan di dalam class tersebut memiliki atribbut, varibael, fungsi atau procedur. Dalam fungsi atau dan procedure tidak bisa berdiri dengan sendiri, keduanya harus ada di dalam classs tertentu, contoh case :

pada dasarnya pemahanan orang memandang manusia dengan berdasarkan kelas contoh semsal kelas manusia, kelas hewan dan kelas tanaman dan lain sebagainya.

Dalam kelas manusia memiliki contoh

Atribut : telinga, tangan, mata dan rambutdll.

Procedure /fungsi : berfikir,

minum, makan dan mendengar

Definisi Objek merupakan bagian dari individu dari sebuah class tertentu.

Example:

Kelas manusia

Objek semisal pak sidik, semua atribu fungsi dan prosedur semua yang melekat pada pak sidik merupakan bagian dari kelas manusia

Fungsi dan atau procedure yang harus bersama sama dalam kelas tidak bisa berdiri sendiri dan tidak ada attribute yang tidak mimiliki kelas, hal yang membedakan antar antribut adalah nilai atau value atribut objek tersebut kecuali kemungkinan dua objek betul-betul sama.

Type code java memiliki case sensitive yang artinya bahasa yang digunakan harus benar-benar benar tidak ada titik atau koma yang salah atau tertingal.

Syarat dalam variabel terdiri dari abjad, ankga da underscore. Karakter nomor awal wajib abjad dan tidak ada spasi diantara variabel tersebut.

Tipe data di bahasa pemrograman java memiliki beberapa kategori yaitu kategori sederhana atau primitif dan komposit aau refernsi.

Data Tipe Primitive

Data Tipe Ini Merupakan Tipe Yang Tidak Di Turunkan Dari Tipe Lain. Ada Delapan Data Tipe sederhana Primitive dan type komposit Pada Java Sebagai Berikut ;

ada 4 model bilangan yaitu ;

tipe integer (bulat) : byte, short, int, long

ada dua tipe data pecahan yaitu (floating point): float, double

tipe data huruf atau karakter dalah char

Boolean adalah satu tipe data yang berisi nilai logika : benar (true) atau salah (false)

Salah satu tipe data sederhana adalah Tipe (55ttribute)

Tipe data angka atau bilangan angka adalah tipe int atau Integer:

Tipe Data	Panjang	Rentang Nilai	Contoh Nilai
byte	8 bit	-2^7 sampai $2^7 - 1$ (-128 sampai 127) (256 kemungkinan nilai)	5 -126
short	16 bit	-2^{15} sampai $2^{15} - 1$ (-32.768 sampai 32.767) (65.535 kemungkinan nilai)	9 -23659
int	32 bit	-2^{31} sampai $2^{31} - 1$ (-2.147.483.648 sampai 2.147.483.647) (4.294.967.296 kemungkinan nilai)	2067456397 -1456398567
long	64 bit	-2^{-63} sampai $2^{63} - 1$ (-9.223.372.036.854.775.808 sampai 9.223.372.036.854.775.807) (18.446.744.073.709.551.616 kemungkinan nilai)	3L -2147483648L 67L

Point floating:

Data tipe floating adalah type data yang digunakan untuk menentukan type model data bilangan real (dapat memiliki nilai pecahan decimal).

Tipe Data	Panjang	Contoh Penulisan Nilai yang Diperbolehkan
float	32 bit	78F -34736.86F 6.4E4F (sama dengan 6,4 x 10^4)
double	64 bit	-2356 3.5E7 67564788965.567

Char:

Data Tipe Text Yaitu file type data yang berguna untuk Varibel Yang Memiliki Nilai Nilai Karakter Tunggal Dalam Nilainya, Char Yang Memiliki Jumlah 16 Bit. Nilai Char Variable Di Tulis dengan cara menggunakan tanda tunggal “ ” Dibawah Ini Contoh Penulisan type data penggunaan Textual :

Public char alphabet = ‘A’;

Public char ascii = ‘111’ //jika di print, akan mendapatkan outpu

//hufur ‘I’;

Logika (Boolean):

Type data logica merupakan data tipe memiliki dua type hasil nilai yang dapat di keluarkan yaitu benar (true) atau salah(false). Hanya memiliki nilai satu logika pada perkembangan teknologi bahasa pemrograman java yaitu Boolean , dibawah ini merupakan contoh pengunaan type data tersebut Boolean ;

```
public boolean status = true;

public boolean check = 10 < 5 ; // nilai check
        menjadi        // false
```

Komposit data memiliki beberapa tipe

Data komposit adalah tipe yang dirangkai dari tipe data sederhana atau dari komposit lainnya. Tipe-tipe ini antara lain adalah array, class, array dan interface. Khusus pada tipe string java dikenal sebagai class, bukan array of character, type data string pada bahasa java menggunakan simbol petik tambahan yaitu (“”)

Semisal ;

String s = “saya memakan nasi”

Operator di java

Operator unary

Operator	Art operator	pemakaian
++operand	Sebelum increment	Int I = 7 Int j = ++i I bernila 8, j bernilai 8
Operand++	Sesudah increment	Int I = 7 Int j = ++i I memiliki nilai 8, j bernilai 7
--operand	sebelum decrement	Int I = 7 Int j = --i I memiliki nilai 6, j bernilai 6
Operand--	Sesudah increment	Int I = 7 Int j = --i I memiliki nilai 6, j bernilai 7

Binary Operator

Arti Operator	Operator	Contoh Pemakaian	Keterangan
Penjumlahan	+	sum=num1 + num2	
Pengurangan	-	diff=num1 - num2	
Perkalian	*	prod=num1 * num2	
Pembagian	/	quot=num1 / num2	jika num1 dan num2 adalah integer, pembagian akan menghasilkan nilai integer tanpa mengikutsertakan sisa, jika terdapat sisa.
Sisa (modulus)	%	mod=num1 % num2	Hasil operasi modulus adalah sisa dari operasi num1 / num2. Hasil operasi modulus memiliki tanda (+/-) yang sama dengan operand pertama

Console atau kesimpulan :

Untuk menampilkan hasil tulisan pada layar:

System.out.println(“…..”)

Example :

int c = 11 * 2 +4 / (7 – 2) ;

urutan dalam pengoprasianya adalah dibawah ini :

int c = 11 * 2 + 4 / 4 ; int c = (hasil) + 5 / 5; int c = (hasil) + 1 ; int c = (hasil);

Operasi antar hubungan

Condition	Operator	Example
Is equal to (atau "is the same as")	==	`int i = 1;` `System.out.println(i==1); // (output : true)`
Is not equal to (atau "is not the same as")	!=	`int i = 1;` `System.out.println(i!=1); // (output : false)`
Is less than	<	`int i = 1;` `System.out.println(i<1); // (output : false)`
Is less than or equal to	<=	`int i = 1;` `System.out.println(i<=1); //(output : true)`
Is greater than	>	`int i = 1;` `System.out.println(i>1); //(output : false)`
Is greater than or equal to	>=	`int i = 1;` `System.out.println(i>=1); // (output : true)`

Operator kondisional

Condition	Operator	Example

If one condition AND another condition	&&	`int i = 1;`
		`int j = 2;`
		`System.out.println((i<1)&&(j>0));`
		`// (output : false)`

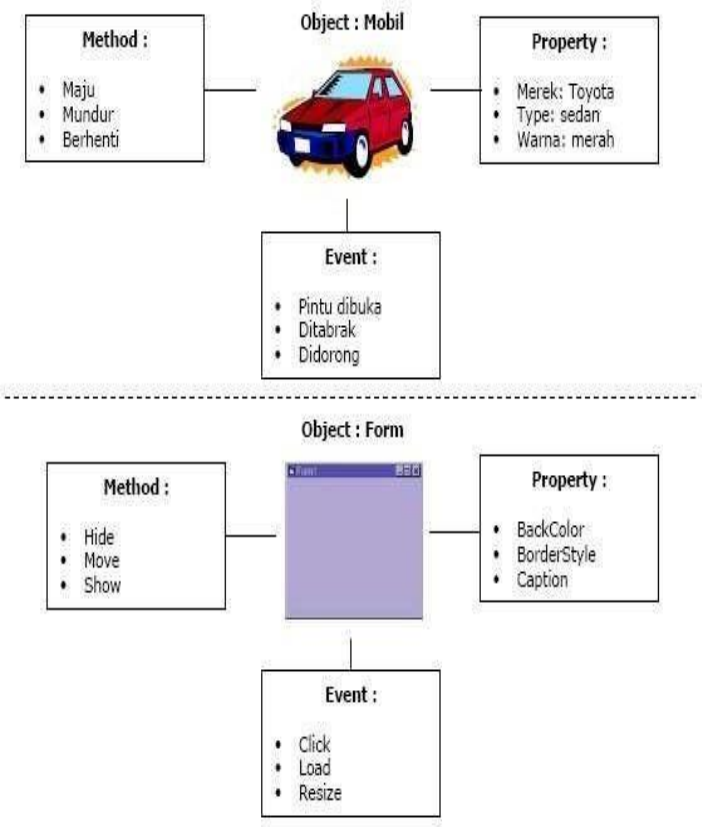
If either condition OR another condition	||	`int i = 1;`
		`int j = 2;`
		`System.out.println((i<1)||(j>0));`
		`// (output : true)`

NOT	!	`int i = 1;`
		`System.out.println(!(i<3));`
		`// (output : false)`

Konsep – konsep pemrograman objek orientasi Utama nya adalah :

Encapsulation (Enkapsulasi)

Pengkapsulan memiliki artik bahwa setiap case dalam representasi objek yang dibungkus dengan sebuah rutin biasa atau sederhana .



Gambar 1.11 Arsitektur Teknologi Java

Pholimorphism atau banyak bentuk

Objek mempunyai kemampuan yang sama dalam melakukan sub rutin dengan cara yang berbeda

Inheritance atau yang di kenal pewarisan suatu mengembangkan sub rutin tanpa melakukan perulangan code sub nilai tersebut.

4.2 Memahami Istilah Property, Method , Object, Dan Event

Karakteristik java

Sederhana

Pemrograman bahasa java menggunakan bahasa yang hamper mirip c=+ dan semakin kesini semakin banyka perbaikan dibagian- bagian yang multiple case inheritance dengan memakain auto memory allocation and garbage colletiaon .

Pemrograman berorientasi objek

Bahasa java berorientasi objek yang create programya dapat di buat modul- modul yang mudah digunakan dan sederhana

Bahasa pemrograman java awal mulanya di kembangkan untuk membuat aplikasi yang lebih mudah dalam pendistribusian dan memiliki libraries terhubung yang terintegrasi dengan java.

Program bahasa Java I

Berikut Listing sederhana Program java menampilkan kata atau kalimat “Apa Kabar Dunia…?” sebagai contoh :

Public class kenalan {

/**

*program java primitive atau sederhana

*/ public static void main (String args[]) {

//view kata pada layar (keterangan)

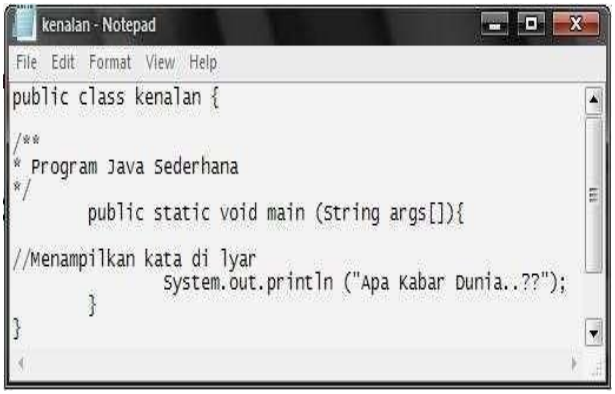
System.out.printn(“apa kabar dunia?”;}

}

Menggunakan Text tools Editor

Langkah selanjutnya sebagai contoh :

Ketik listing code java berikut ke tols Notepad plus atau editor lain oleh karena bahasa Java merupakan *case 60ttribute atau sangat sensitive dalam penulisan* maka dalam pengetikan atau entri code maka haruslah sangat jeli dan teliti.



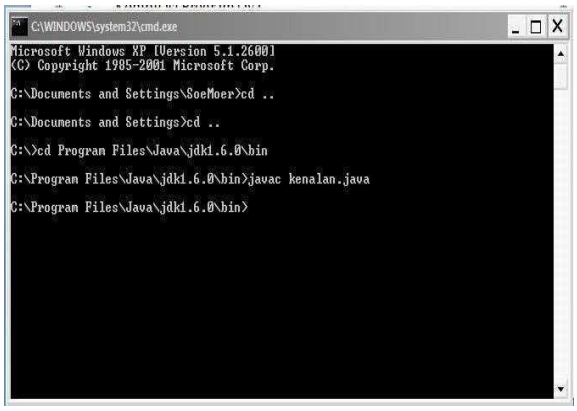
Gambar 1.11 Arsitektur Teknologi Java

Menyimpan project java

Simpan project java yang telah di buat dengan rename menjadi kenalan.java pada direktori yang di kehendaki atau direktori c;/programfiles/java/jdk1.6.0/bin

Compiler project java terse ut

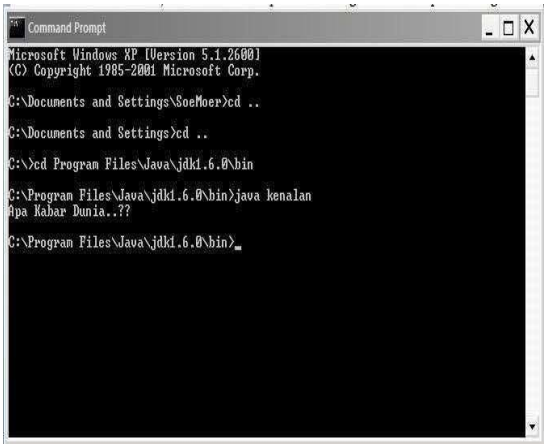
Untuk mencomplierkan dengan editor text digunakan java compiler.program javac akan menjalankan tugas compiler project yang kita buat dalam bytecode langkah-langkahnya seperti melalui command prompt berikut ini ;



Gambar 1.11 Arsitektur Teknologi Java

Menjalankan aplikasi Program

Untuk melihat hasil compilasi maka digunakan fungsi *interpreter* sesuai perintah code sebagai contoh:



Gambar 1.11 Arsitektur Teknologi Java

4.3 Package

Virtual Method Invocation (VMI) bisa terjadi jika terjadi polimorfisme dan overriding. Pada saat obyek yang sudah di buat tersebut memanggil overridden method pada parent class, compiler Java akan melakukan invocation (pemanggilan) terhadap overriding method pada subclass , dimana yang seharusnya di panggil adalah overridden method. Berikut contoh terjadinya VMI:

Listing Program

classParent{ intx=5; publicvoidInfo(){

System.out.println(“IniclassParent”);

}

```
}

classChildextendsParent{ intx=10; publicvoidInfo(){

System.out.println(“IniclassChild”);

}

}

publicclassTes{

publicstaticvoidmain(Stringargs[]){Parenttes=newChild(); System.out.println(“Nilaix=“+tes.x); tes.Info();

}

}
```

Hasil dari running program diatas adalah sebagai berikut:

```
Nilaix=5 IniclassChild
```

Polymorphic arguments adalah tipe suatu parameter yang menerima suatu Nilai yang bertipe subclass-nya. Berikut contoh dari polymorphics arguments:

Listing Program

```
Class Pegawai{

}

Class Manajer extends Pegawai{

…

}

Public class Tes{

Public static void Proses(Pegawaipeg){

…

}

Public static void main(Stringargs[]){Manajerman=newManajer(); Proses(man);

}

}
```

Pernyataan instance of sangat berguna untuk mengetahui tipe asal dari suatu Polymorphic arguments. Untuk lebih jelasnya, misalnya dari contoh program sebelumnya, kita sedikit membuat modifikasi pada class Tes dan ditambah sebuah class baru Kurir, seperti yang tampak dibawah ini:

Listing Program

```
Class Kurir extends Pegawai

}

Public classTes{ publicstaticvoidProses(Pegawaipeg){ if(peginstanceofManajer){

…lakukantugas-tugasmanajer…

}elseif(peginstanceofKurir){

  …lakukantugas-tugaskurir…


  }else{

  …lakukantugas-tugaslainnya…

  }


  }

  publicstaticvoidmain(Stringargs[]){ Manajerman=newManajer();

  Kurirkur=newKurir();

  Proses(man);Proses(kur);

  }
```

Seringkali pemakaian instance of diikuti dengan casting object dari tipe parameter ke tipe asal. Misalkan saja program kita sebelumnya. Pada saat kita sudah melakukan instance of dari tipe manajer, kita dapat melakukan casting object ke tipe asalnya, yaitu manajer. Caranya adalah seperti berikut:

Listing Program

```
if(peginstanceofManajer)

{

Manajerman=(Manajer)peg;

…lakukantugas-tugasmanajer…

}
```

Package

Package adalah sebuah sarana untuk mengelompokkan atau mengorganisasikan kelas dan *interface* yang sama atau sekelompok menjadi satu unit tunggal dalam *library*. Package mempengaruhi mekanisme hak akses ke kelas didalamnya. Hal terpenting yang diperhatikan pada saat mendeklarasikan package, bahwa *class* tersebut harus disimpan pada suatu *directory* yang sama dengan nama packagenya. Alasan menggunakan package pada java ialah untuk menghindari tabrakan nama kelas yang akan dibuat dengan nama kelas yang sudah ada. Selain itu, salah satu yangmenjadi keuntungan menggunakan package adalah untuk mudahnya developer dalam hal mencari dan me-

manage akses yang diberikan. Mengerti akan konsep dari package akan membantu mengelola dan menggunakan file yang disimpan didalam JAR (Java Archive).

Package juga mempengaruhi mekanisme hak akses ke kelas-kelas di dalamnya.

Pengaruh Package terhadap Method main()

Kelas yang mengandung method main() memilki syarat tidak berada dalam suatu package, dan hirarki posisi foldernya di atas package yang diimport.

Membuat Package

Ada tiga langkah untuk membuat package :

Mendeklarasikan dan memberi nama package.

Membuat struktur dan nama direktori yang sesuai dengan struktur dannama package.

Mengkompilasi kelas-kelas sesuai dengan packagenya masing-masing.

Mendeklarasikan dan Memberi Nama Package

Deklarasi package harus diletakkan pada bagian paling awal (sebelum deklarasi import) dari *source code* setiap kelas yang dibungkus package tersebut.

Bentuk umum deklarasi package :

package namaPackage;

Deklarasi tersebut akan memberitahukan kompilator, ke *library* manakahsuatu kelas dikompilasi dan dirujuk.

Syarat nama package :

Diawali huruf kecil,

Menggambarkan kelas-kelas yang dibungkusnya,

Harus unik (berbeda dengan nama package standard),

Merepresentasikan path dari package tersebut

Harus sama dengan nama direktorinya.

Contoh package standard :

java.lang (berisi kelas-kelas fundamental yang sering digunakan).

java.awt dan *javax.swing* (berisi kelas-kelas untuk membangun aplikasi GUI)

java.io (berisi kelas-kelas untuk proses input output)

Membuat Struktur Direktori

Pada langkah ini, buatlah direktori menggunakan file manager (di windows menggunakan explorer) sesuai struktur package dari langkah sebelumnya. Kemudian tempatkan kelas-kelas tersebut ke direktori yang bersesuaian (mirip seperti menyimpan file-file ke dalam folder).

Package dapat bersarang di package lain, sehingga dapat dibuat hirarkipackage.

Bentuk umum pernyataan package multilevel :

package namaPackage1[.namaPackage2[.namaPackage3]];

Contoh hirarki package di JDK :

package java.awt.image;

Compile dan Run Kelas dari suatu Package

Selanjutnya masing-masing kelas tersebut dalam package tersebut dikompilasi menjadi byte code (*.class). Artinya package tersebut siap digunakan.

Menggunakan Package

Ada dua cara menggunakan suatu package yaitu :

Kelas yang menggunakan berada dalam direktori (package) yang sama dengan kelas-kelas yang digunakan. Maka tidak diperlukan import.

Kelas yang menggunakan berada dalam direktori (package) yang berbeda dengan kelas-kelas yang digunakan. Maka pada awal source code di kelas pengguna harus mencantumkan :

import namaPackage.NamaKelas; atau

import namaPackage.;*

Contoh :

import java.text.DecimalFormat;import javax.swing.;*

Setting Classpath

Path hirarki package, didaftarkan sebagai salah satu nilai variabel lingkungan yang bernama Classpath.Classpath diset dengan aturan : berawal dari drive (C:\ atau D:\) sampai dengan satu tingkat sebelum kita mendeklarasikan package.

4.4 ARRAY

Array atau disebut juga larik merupakan struktur kumpulan variabel secara berurutan yang menyimpan data/nilai (isi variabel) bertipe sama. Satu variabel dalam array disebut sebagai elemen dan urutan akses elemen disebut indeks. Indeks elemen dalam array selalu dimulai dari angka 0. Dalam pemrograman Java, deklarasi arrayditandai dengan kurung siku .

Array Satu Dimensi

Array satu dimensi adalah array atau larik dengan satu ruang penyimpanan. Untuk lebih jelas mengenai perbedaan variabel biasa dengan array satu dimensi dapat dilihat pada gambar di bawah ini.

```
double nilai0 = 80.5;
double nilai1 = 65;
double nilai2 = 50;
double nilai3 = 78.9;
double nilai4 = 92.5;
```

0 1 2 3 4

Untuk menyimpan 5 nilai yang berbeda dibutuhkan 5 variabel biasa untuk masing- masing nilai tersebut. Dengan

array, kelima nilai tersebut dapat disimpan dalam 1 variabel dengan ruang penyimpanan yang berbeda sehingga seolah-olah hanya menggunakan 1 variabel saja. Untuk mengakses ruang penyimpanannya, serta urutan akses (indeks) setelah nama variabel.

Nilai[0] =nilai[1] =nilai[2] =nilai[3] =nilai[4] =

Berikut bentuk deklarasi array satu dimensi.

deklarasiin isialisasi

Bentuk 1 : *tipeData[] namaArray = {data-1, data-2, data-n};*

Jumlah dimensi elemen ke-0 elemen ke-1 elemen ke-(n-1)

Bentuk 2 :

Jumlah dimensi keyword

tipeData[] namaArray= newtipeData[n];] deklarasi

namaArray[0] = data-1;] elemen ke-0

inisialisasi *namaArray[1] = data-2;*] elemen ke-1

namaArray[n-1] = data-n;] elemen ke-(n-1)

Keterangan :

tipeData disesuaikan dengan jenis data/nilai yang akan disimpan dalamarray.

*namaArray*adalah nama variabel.

n adalah jumlah/banyaknya elemen array.

Pada bentuk 1, data/nilai ditentukan secara langsung (inisialisasi) setelah deklarasiarray. Data/nilai satu dengan yang lainnya dipisahkan dengan simbol koma (,). Banyaknya data/nilai yang di-inisialisasi-kan sama dengan banyaknya elemen arrayyang akan terbentuk. Indeks elemennya dimulai dari angka 0 sampai dengan n-1

Pada bentuk 2, inisialisasi dilakukan setelah proses deklarasi array. Banyaknya data/nilai yang di-inisialisasikan tergantung kepada nilai *n* pada deklarasi array dengan indeks elemennya dimulai dari angka 0 sampai dengan n-1.

Contoh 1:

Berikut adalah contoh array dimensi satu (bentuk 1) :

```
public static void main(String[] args){

    double[] nilai = {80.5, 65, 50, 78.9, 92.5};

    System.out.println("elemen ke-0 = " + nilai[0]);
    System.out.println("elemen ke-1 = " + nilai[1]);
    System.out.println("elemen ke-2 = " + nilai[2]);
    System.out.println("elemen ke-3 = " + nilai[3]);
    System.out.println("elemen ke-4 = " + nilai[4]);
}
```

Pembahasan :

```
run:
elemen ke-0 = 80.5
elemen ke-1 = 65.0
elemen ke-2 = 50.0
elemen ke-3 = 78.9
elemen ke-4 = 92.5
```

Banyaknya data yang di-inisialisasi-kan berjumlah 5 (*n=5*) yaitu {80.5, 65, 50, 78.9,92.5} dengan urutan akses elemen dari 0 sampai dengan n-1.

Array Multi Dimensi

Array multi dimensi memiliki lebih dari satu dimensi atau ruang penyimpanandata/nilai. Pada array dua dimensi, ruang penyimpanan berupa matriks atau tabel. Berikut bentuk deklarasi array multi dimensi.

Bentuk 1 :

deklarasiinisialisasi

tipeData[][] namaArray = { {data-1, data-n}, elemen x0

{data-1, data-n}, elemen x1

{data-1, data-n} elemen x2

};

jumlah dimensi elemen y0 elemen y1

Bentuk 2 :

jumlah dimensi keyword ukuran dimensi

tipeData[][] namaArray = new tipeData[x][y];

deklarasi *namaArray[0][0] = data-1;* elemen 0,0 *namaArray[0][y-1] = data-n;*

elemen 0,y-1

inisialisasi

Keterangan :

namaArray[x-1][0] = data-1;

elemen x-1,0

namaArray[x-1][y-1] = data-n;

elemen x-1, y-1

Jumlah kurung siku menyatakan jumlah dimensi yang digunakan dimana(*[][]*) berarti menggunakan array 2 dimensi.

Pada bentuk 2, *[x][y]* menyatakan ukuran dimensi yang digunakan.

Contoh 1:

Berikut adalah contoh array multi dimensi (bentuk 1) :

```
public static void main(String[] args){

    String[][] mhs = { {"1700001", "Tora"},
                       {"1700002", "Tompel"},
                       {"1700003", "Toyo"}
                     };

    System.out.println("NPM \t\t Nama Mahasiswa");

    System.out.println(mhs[0][0] + "\t\t" + mhs[0][1]);
    System.out.println(mhs[1][0] + "\t\t" + mhs[1][1]);
    System.out.println(mhs[2][0] + "\t\t" + mhs[2][1]);
}
```

```
run:
NPM             Nama Mahasiswa
1700001         Tora
1700002         Tompel
1700003         Toyo
```

Pembahasan :

Contoh di atas menggunakan array 2 dimensi dengan ukuran 3x2 yaitu 3 baris (0-2)dan 2 kolom (0-1). Array 2 dimensi ditandai dengan kurung siku sebanyak 2 kali. Karakter '\t' akan mencetak 'tab' sebanyak 1x.

Perulangan Array

Perulangan pada array menggunakan pernyataan *for* dan hanya berlaku untuk array satu dimensi. Salah satu pernyataan *for* yang dapat digunakan pada array adalah *for each* dimana akan mengiterasi dan menyimpan nilai setiap elemen array dalam variabel tunggal. Berikut sintaks atau bentuk penulisan pernyataan *for each* pada array.

for(tipeData namaVariabel : namaArray){kode_program1;

kode_program2; blok kode program

…;

}

Keterangan :

*tipeData*yang digunakan harus sama dengan tipe pada array.

Contoh 1 :

Pada contoh array satu dimensi di atas,

```
public static void main(String[] args){

    double[] nilai = {80.5, 65, 50, 78.9, 92.5};

    System.out.println("elemen ke-0 = " + nilai[0]);
    System.out.println("elemen ke-1 = " + nilai[1]);
    System.out.println("elemen ke-2 = " + nilai[2]);
    System.out.println("elemen ke-3 = " + nilai[3]);
    System.out.println("elemen ke-4 = " + nilai[4]);
}
```

dapat diubah menggunakan perulangan array menjadi :

```
public static void main(String[] args){

    short m = 0;
    double[] nilai = {80.5, 65, 50, 78.9, 92.5};

    for (double n : nilai){
        System.out.println("elemen ke-" + m + " = " + n);
        m++;
    }
}
```

tetap dengan hasil eksekusi (*run*) yang sama.

Run :

Elemen ke 0 = 80

Elemen ke 1 = 65

Elemen ke 2 = 50

Elemen ke 3 = 78

Elemen ke 4 = 92

Contoh 2 :

Berikut adalah contoh array dimensi satu (bentuk 2) :

```
public static void main(String[] args){
    short i;
    String[] terbilang = new String[10];
    terbilang[0] = "nol";
    terbilang[1] = "satu";
    terbilang[2] = "dua";
    terbilang[3] = "tiga";
    terbilang[4] = "empat";
    terbilang[5] = "lima";
    terbilang[6] = "enam";
    terbilang[7] = "tujuh";
    terbilang[8] = "delapan";
    terbilang[9] = "sembilan";
```

```
    for(i=0; i<terbilang.length; i++){
        System.out.println(i + " = " + terbilang[i]);
    }
}
```

```
run:
0 = nol
1 = satu
2 = dua
3 = tiga
4 = empat
5 = lima
6 = enam
7 = tujuh
8 = delapan
9 = sembilan
```

Pembahasan :

Fungsi *length* pada array menghasilkan nilai banyaknya array yang terbentuk sehingga dapat digunakan pada perulangan *for* sebagai batas akhir perulangan untuk mengakses setiap elemen pada array.

Contoh 3:

Berikut adalah contoh array multi dimensi (bentuk 2) dengan contoh kasus jadwal railink Medan – Kualanamu sesuai tabel di bawah ini.

MEDAN - KUALANAMU AIRPORT		
NO KERETA	KEBERANGKATAN	KETIBAAN
U2	03:30	04:01
U4	05:15	05:46
U6	06:00	06:31

```
public static void main(String[] args){
    short bar, kol;
    String[][] mdn_kno = new String[3][3];

    mdn_kno[0][0]="U2";
    mdn_kno[0][1]="03:30";
    mdn_kno[0][2]="04:01";
```

```
run:
Jadwal Railink Bandara (WIB)
No Kereta        Berangkat        Tiba
U2               03:30            04:01
U4               05:15            05:46
U6               06:00            06:31
```

```
    mdn_kno[1][0]="04";
    mdn_kno[1][1]="05:15";
    mdn_kno[1][2]="05:46";

    mdn_kno[2][0]="06";
    mdn_kno[2][1]="06:00";
    mdn_kno[2][2]="06:31";

    System.out.println("Jadwal Railink Bandara (WIB)");
    System.out.println("No Kereta \t Berangkat \t Tiba");

    for (bar=0; bar < mdn_kno.length; bar++){
        for (kol=0; kol < mdn_kno[bar].length; kol++){
            System.out.print(mdn_kno[bar][kol] + "\t\t");
        }
        System.out.println("");
    }
}
```

Pembahasan :

Contoh di atas menggunakan array 2 dimensi dengan ukuran 3x3 yaitu 3 baris (0-2)dan 3 kolom (0-2). Array 2 dimensi ditandai dengan kurung siku sebanyak 2 kali dan ukuran array ditandai dengan . Karakter '\t' akan mencetak 'tab'sebanyak 1x.

Perintah menghasilkan nilai banyaknya array pada ruang dimensix dan perintah menghasilkan nilai banyaknya array pada ruang dimensi y.

Class Array Pada Java

Untuk melakukan operasi lebih lanjut pada array, pemrograman Java telah menyediakan class untuk membantu programer mengolah data/nilai pada array.

Class Arrays

Class Arrays merupakan turunan dari java.util.Arrays. Class Arrays melakukan operasi terhadap variabel array, yang berarti class ini tidak dapat digunakan tanpa adanya variabel array (baik array satu dimensi maupun array multidimensi). Dengan Class Arrays, programer dapat melakukan operasi terhadap variabel arrayseperti menyalin (*copying*), mengurutkan (*sorting*), mencari (*searching*) dan lain-lain. Sebelum menggunakan class arrays, terlebih dahulu programer harus menambahkan class tersebut melalui import (import java. Until.arrays ;). Berikut beberapaoperasi (*method*) Class Arrays yang akan dibahas.

copyOf(arraySumber, ukuranArray) adalah fungsi untuk menyalin array sumber (baik struktur maupun isi) secara keseluruhan ke array tujuan.Contoh :

```
public static void main(String[] args){
    double[] lama = {80.5, 65, 50, 78.9, 92.5};
    double[] baru = Arrays.copyOf(lama, lama.length);

    for (double n : baru){
        System.out.print(n + " ");
    }
}
```

```
run:
80.5 65.0 50.0 78.9 92.5
```

copyOfRange(arraySumber, indeksAwal, indeksAkhir) sama seperti *copyOf*, namun dapat ditentukan/dikostumasi ukuran 73a nisi array sumber ke array tujuan. Contoh :

```
public static void main(String[] args){
    double[] lama = {80.5, 65, 50, 78.9, 92.5};
    double[] baru = Arrays.copyOfRange(lama, 2, lama.length);

    for (double n : baru){
        System.out.print(n + " ");
    }
}
```

```
run:
50.0 78.9 92.5
```

sort(arraySumber, indeksAwal, indeksAkhir) adalah metodeuntuk melakukan pengurutan pada seluruh elemen array. Parameter *indeksAwal*dan *indeksAkhir*bersifat opsional. Contoh :

```
public static void main(String[] args){
    double[] nilai = {80.5, 65, 50, 78.9, 92.5};
    Arrays.sort(nilai);

    for (double n : nilai){
        System.out.print(n + " ");
    }
}
```

```
run:
50.0 65.0 78.9 80.5 92.5
```

untuk melakukan pengurutan pada elemen tertentu pada array, dapatmenambahkan indeks awal dan indeks akhir dari elemen array yang ingin diurutkan pada pada parameter kedua dan ketiga. Contoh :

```
double[] nilai = {80.5, 65, 50, 78.9, 92.5};
Arrays.sort(nilai,2,4);
```

binarySearch(arraySumber, nilaiCari) adalah fungsi untuk melakukan pencarian biner berdasarkan nilai yang ada didalam array. Fungsi ini menghasilkan nilai bertipe *int* yang menyatakan indeks elemen. Apabila nilaiyang dicari tidak ditemukan, fungsi akan menghasilkan nilai lebih kecil 0. Contoh :

```
public static void main(String[] args){
    int indeks;
    double cari = 78.9;
    double[] nilai = {80.5, 65, 50, 78.9, 92.5};
    indeks = Arrays.binarySearch(nilai, cari);

    if (indeks < 0){
        System.out.println(cari + " tidak ditemukan");
    }else{
        System.out.println(cari + " ditemukan pada indeks ke-" + indeks);
    }
}
```

Run :

78.9 ditemukan pada indeks ke -3

equals(array1, array2) adalah fungsi untuk melakukan perbandingan terhadap seluruh nilai elemen array. Fungsi ini menghasilkan nilai bertipe *boolean* yang menyatakan *true* apabila seluruh nilai elemen sama dan *false*apabila salah satu nilai elemen tidak sama. Contoh :

```
public static void main(String[] args){
    boolean result;

    char[] huruf1 = {'m', 's', 'n'};
    char[] huruf2 = {'m', 's', 'n'};
    char[] huruf3 = {'n', 'm', 's', 'i'};

    result = Arrays.equals(huruf1, huruf2);
    System.out.println("nilai array huruf1 & huruf2 sama?"+result);

    result = Arrays.equals(huruf1, huruf3);
    System.out.println("nilai array huruf1 & huruf3 sama?"+result);
}
```

```
run:
nilai array huruf1 & huruf2 sama?true
nilai array huruf1 & huruf3 sama?false
```

fill(arraySumber, nilaiBaru) merupakan metode (*method*) untukmengisi nilai kedalam elemen array.
Contoh 1:

```
public static void main(String[] args){
    double[] hasil = {80.5, 65, 50, 78.9, 92.5};
    Arrays.fill(hasil, 55);

    System.out.println("nilai elemen array:");
    for (double n : hasil){
        System.out.print(n + " ");
    }
}
```

```
run:
nilai elemen array:
55.0 55.0 55.0 55.0 55.0
```

Terdapat 5 elemen dalam array 'hasil' yang telah memiliki nilai awal. Dengan *method fill(hasil, 55)* pada class Arrays, seluruh nilai pada setiap elemen array 'hasil' diubah menjadi '55'.

Contoh 2:

```
public static void main(String[] args){
    double[] hasil = {80.5, 65, 50, 78.9, 92.5};
    Arrays.fill(hasil, 2, 4, 55);

    System.out.println("nilai elemen array:");
    for (double n : hasil){
        System.out.print(n + " ");
    }
}
```

Run :

nilai elemn array ;

80.5 65.0 55.0 92.5

Untuk mengubah sebagian nilai elemen dalam array, tentukan parameter ke-2 yaitu indeks awal dan parameter ke3 yaitu banyaknya elemen array.

BAB 5

MENGGUNAKAN IDE NETBEANS

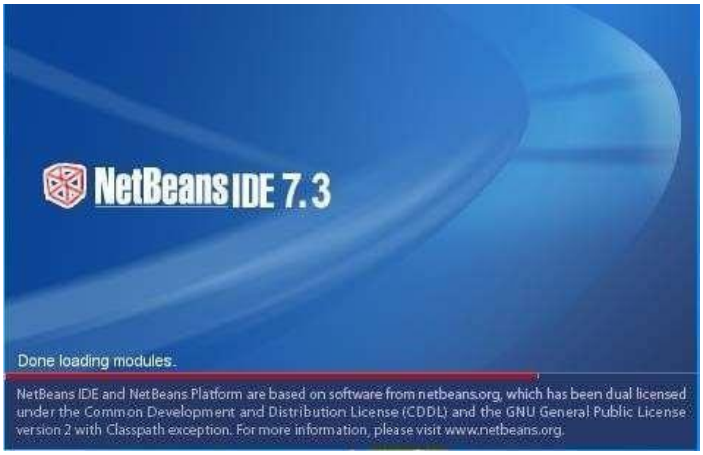
5.1 Lingkungan Kerja Ide Netbeans

Tools IDE merupakan lingkup pemrograman java yang diintegrasikan atau di hubungkan kedalam software untuk menyediakan pengembangan tampilan display tampilan, tools kode editor atau suatu text, compiler (run) atau suatu debugger dan interpreter (code).

Langkah menjalankan sebagai contoh :

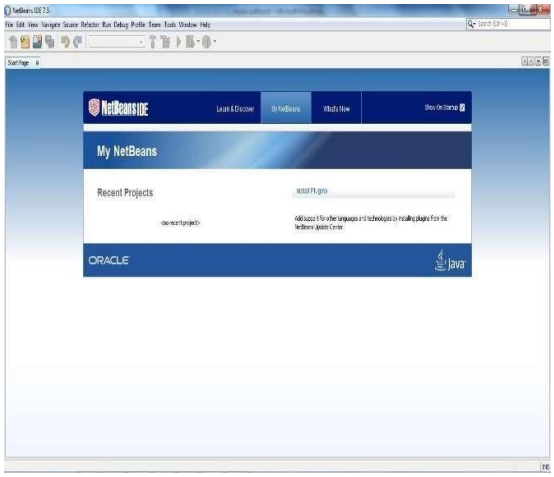
Aktifkan aplikasi tools NetBeans

Tekan Tombol Start > Pilih All Programs > Pilih NetBeans > Pilih NetBeans IDE logo NetBeans di computer desktop masing-masing.



Gambar 1.11 netbeans ide 7.3

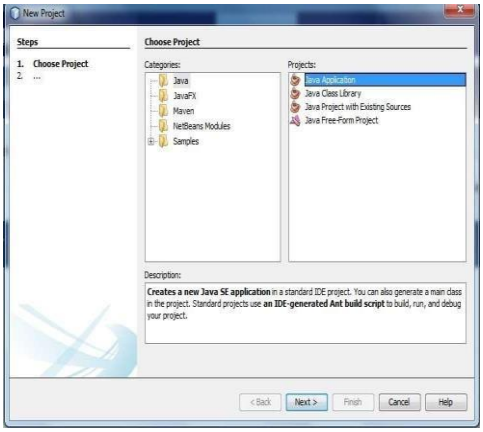
Selanjutnya akan muncul tampilan GUI seperti contoh :



Gambar 1.11 Tampilan Gui Netbeans

create Project new

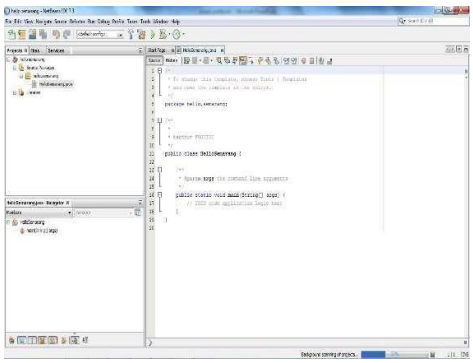
Klick selanjutnya File selanjutnya New Project atau bisa juga dengan klick icon New Project Selanjutnya akan tampilan *box* new. Selanjutnya Klick Next dan ikuti langkah selanjutnya sampai selesai dan selanjutnya project baru akan tampil di GUI.



Gambar 1.11 New Project

Menulis Program

Selanjutnya Setelah membangun new project maka langkah selanjutnya menulis program pada tab editor aplikasi NetBeans.

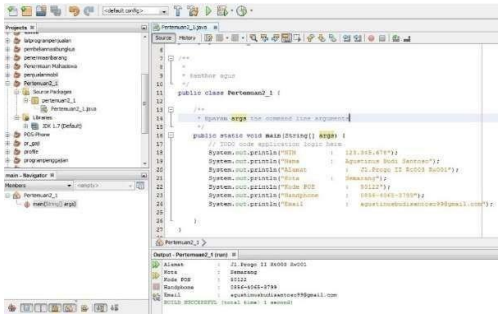


Gambar 1.11 editor netbeans

Pada Source

/*

```
* To change this template, choose Tools | Templates □□* and open the template in the
editor.
*/
package hello.semarang;
/**
*
* @author FUJITSU
*/
public class HelloSemarang {
/**
* @param args the command line arguments
*/
public static void main(String[] args) {
// TODO code application logic here
}

}
```



Gambar 1.11 editor aplikasi netbeans

Perhatikan, tanda /* * */ serta /** * */ merupakan penanda untuk komentar. Jadi tulisan apapun jika berada diantara tanda itu tadi tidak akan dipedulikan oleh program alias tidak berpengaruh terhadap hasil program yang sudah kita tulis. Tanda // juga komentar, bedanya dengan sebelumnya, kalo yang sebelumnya bisa berbaris baris, yang ini cuman berlaku untuk satu baris saja, dari // sampe barisnya selesai.

Public static void main(String[] args) adalah penanda bagian start program, jadi 78ttribut akan baca program yang sudah kita tulis mulai dari situ kebawah, dan kita bisa nulis kode kode pemrograman dibawahnya atau diantara tanda kurung kurawal ({…})

di Java kalo penulisan program/baris perintah harus hati hati ,karena case sensitive, jadi besar kecilnya huruf mempengaruhi.

Nah, sekarang kita coba buat Hello Semarang. Coba ketikkan ini diantara “// TODO code application here” dan “}”

System.out.println(“Hello Semarang”);

Compile dan Running aplikasi Program

untuk mengkompilasi program dengan cara dilakukan dengan cara

Klick Run selanjutnya Build selanjutnya Project

Langkah selanjutnya klik program run Run Project

Latihan membuat profile

Run :
NIM : 089671418611
nama: Muhammad sidik
alamat : jl kidalem kudu baru rt 1 rw 6 genuk semarang
kota : semarang
kode pos:50116
hp:089671418611

Email:mgcn.sidik@gmail.com
build succesfull

```
/*
* To change this template, choose Tools | Templates □        * and open the template
in the editor.
*/
package pertemuan2_1;
/**
*|
* @author sidik
*/
public class Pertemuan2_1 {
/**
* @param args the command line arguments
*/
public static void main(String[] args) {
// TODO code application logic here
        // TODO code application logic here
        System.out.println("NIM          :   08967148611");
System.out.println("Nama         :   Muhammad Sidik");
System.out.println("Alamat       :   Jl.kudu baru genuk semarang");
System.out.println("Kota         :   Semarang");
System.out.println("Kode POS     :   50116");
System.out.println("Handphone    :   089671418611");
System.out.println("Email        :   mgcn.sidik@gmail.com");
```

}

}Menghitung luas PersegiPanjang

Run

Menghitung luas persegi panjang

Masukan panjang =4

Masukan lebar=8

Luas =32

Ini merupakan uji coba pertamaku

Build succesfull

```
/*
* Untuk mengubah template ini, pilih Alat | Template
* dan buka template di editor.


*/
package pertemuan2_persegi;
import java.util.Scanner;
/**
*
* @author
*/
public class Pertemuan2_Persegi {
/**
    * @param args the command line arguments
    */
    public static void main(String[] args) {
    // TODO code application logic here
    Scanner input = new Scanner (System.in);
    int p, l, luas;
System.out.println("Menghitung Luas Persegi Panjang");
System.out.println("-----------------------------");
System.out.print("Masukkan Panjang  = "); p=input.nextInt();
System.out.print("Masukkan Lebar = "); l=input.nextInt();
luas=p*l;
```

```
System.out.println("----------");
System.out.println("Luas          = "+luas);
System.out.println("Ini merupakan uji coba  pertama ku");
}
}
```

5.1 Lingkungan Kerja Ide Netbeans

Lingkungan kerja dari IDE NetBeans secara umum dapat dilihat pada gambar

Adapun keterangan mengenai lingkungan kerja IDE NetBeans adalah sebagaiberikut :

Menu bar

Merupakan kumpulan menu yang menyediakan fitur atau fungsi, dan fasilitastertentu.

Toolbar

Merupakan kumpulan tombol pintas untuk menjalankan fungsi atau fitur tertentu berupa gambar icon.

Jendela Projects

Merupakan area yang menampilkan seluruh daftar projek yang berada dalam IDE NetBeans. Untuk menampilkan jendela 'Projects' (Gambar 1.3), klik menu 'Window'

- ✓ 'Projects'.

Jendela Navigator

Merupakan area navigasi baik susunan kode program maupun susunan komponen pada lembar perancangan. Untuk menampilkan jendela 'Navigator' (Gambar 1.4), klik menu 'Window' 'Navigator'.

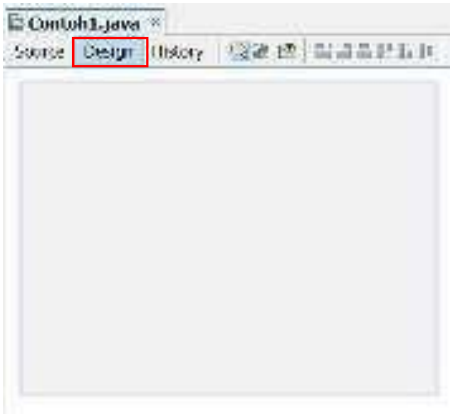
Lembar Perancangan

Merupakan tempat perancangan antar muka (Gambar 1.5), perancangan kode program (Gambar 1.6), dan riwayat pekerjaan.

Jendela Palette

Merupakan area yang menyediakan komponen untuk merancang antar muka berbasis *Graphic User Interface* (GUI) pada lembar perancangan. Untuk menampilkan jendela 'Palette' (Gambar 1.7),

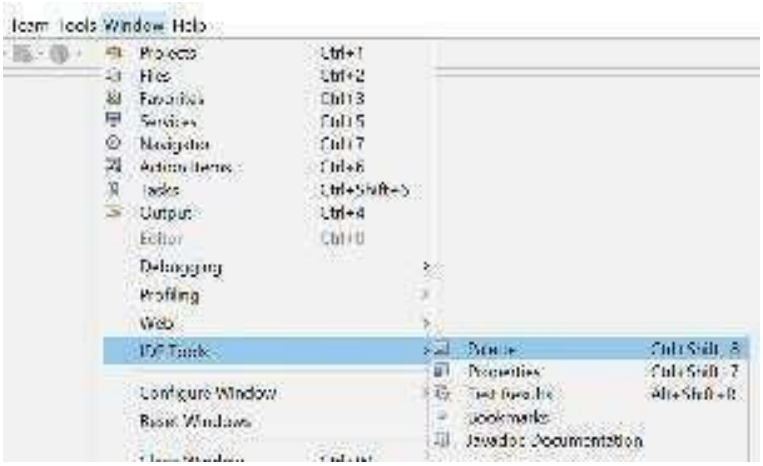
- ✓ Klik menu 'Windows'
- ✓ 'IDE Tools'
- ✓ 'Palette'.



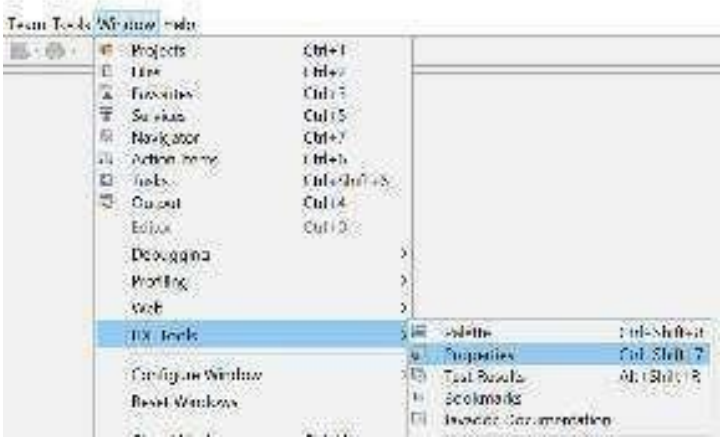
Gambar 1.5 Tampilan Perancangan Antar Muka

Jendela Properties

Merupakan area yang berisi item-item properti dari suatu komponen yang digunakan pada lembar perancangan. Pada area ini pula dapat ditentukan nilai atau data suatu komponen agar pengolahan komponen atau nilai menjadi lebihmudah. Untuk menampilkan jendela 'Properties' (Gambar 1.8), Klik menu 'Windows' IDE Tools' 'Properties'.



Gambar 1.11 Arsitektur Teknologi Java



Gambar 1.8 Cara Menampilkan Jendela Properties

Percobaan

Instalasi Java + Netbeans

Dalam percobaan berikut ini, akan dijelaskan langkah-langkah dalam melakukan instalasi Java dan Netbeans. File-file yang dibutuhkan yaitu Java Development Kit (JDK) dan Netbeand IDE yang dapat diunduh secara gratis di https://www.oracle.com/technetwork/java/index.html untuk JDK dan https://netbeans.org/downloads/index.html untuk Netbeans IDE.

Menginstall software Java Development Kit (pada contoh di bawah menggunakan jdk-8) di windows komputer dengan langkah-langkahnya sebagai berikut :

Setelah JDK berhasil diunduh, double click pada 'jdk installer'.

Java Development Kit Installer

Pada halaman 'Welcome', klik tombol 'Next >' untuk melanjutkan ke proses

berikutnya.

Gambar 1.11 Arsitektur Teknologi Java

Halaman Welcome

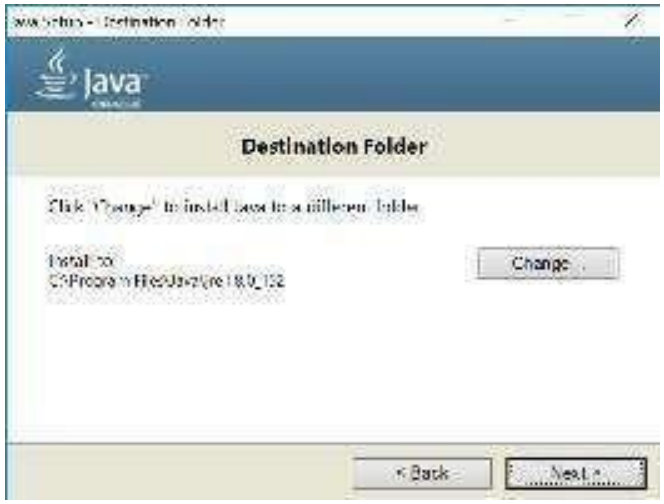
Pada halaman 'Custom Setup', lokasi instalasi dapat diubah dengan kliktombol 'Change'. Klik tombol 'Next >' untuk melanjutkan ke proses instalasi JDK. Pada halaman tersebut menampilkan informasi progres instalasi JDK.

Halaman Custom Setup

Gambar 1.11 Arsitektur Teknologi Java

Halaman Proses Instalasi JDK

Setelah selesai proses instalasi JDK, proses berikutnya adalah instalasiJava Runtime Environment (JRE). Halaman instalasi JRE, lokasi instalasi dapat diubah dengan klik tombol 'Change...'. Klik tombol 'Next >' untuk melanjutkan ke proses instalasi. Pada halaman tersebut menampilkan informasi progres instalasi JRE. Setelah selesai proses instalasi maka akan tampil halaman hasil yang berisi informasi status instalasi.



Gambar 1.11 Arsitektur Teknologi Java

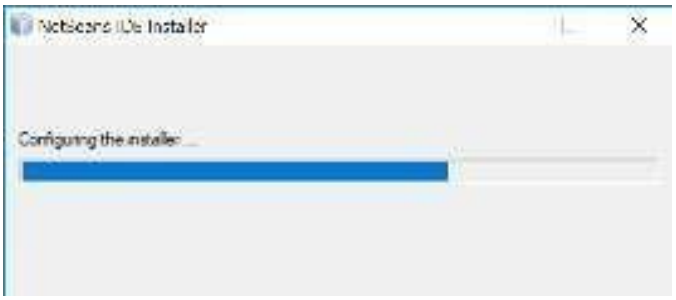
Halaman Instalasi JRE

Gambar 1.11 Arsitektur Teknologi Java

Halaman Proses Instalasi JRE

Menginstall text editor dan console NetBeans IDE dengan langkah-langkahadalah sebagai berikut :

- ✓ Setelah NetBeans berhasil diunduh, double click pada 'netbeans'.
- ✓ NetBeans Installer



Gambar 1.11 Arsitektur Teknologi Java

Configuring the Installer

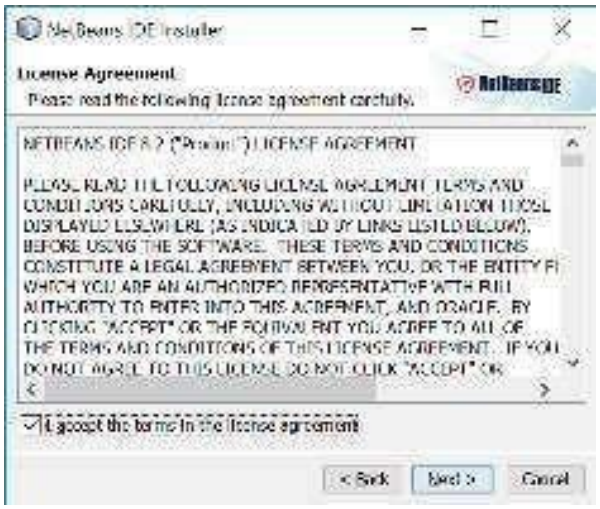
Halaman pertama yang tampil adalah halaman 'Welcome'. Pada halaman ini, terdapat pilihan untuk instalasi 'application servers'. Klik tombol 'Next >' untuk melanjutkan ke proses berikutnya.

Gambar 1.11 Arsitektur Teknologi Java

Halaman Welcome

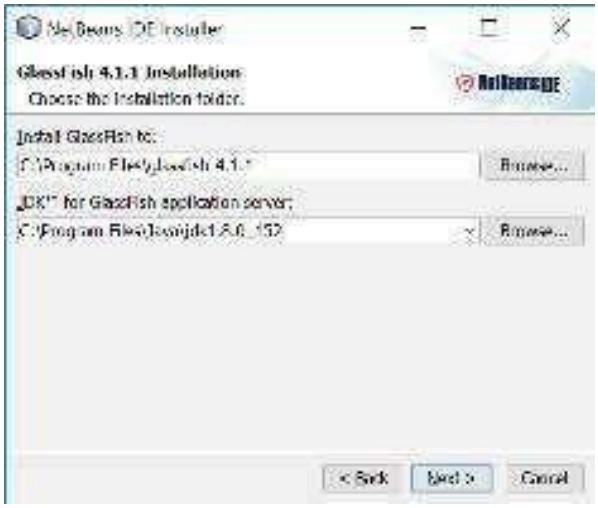
Halaman selanjutnya adalah halaman 'License Agreement'. Centang pada bagian "I accept the terms in the license agreement", kemudian klik tombol 'Next >' untuk melanjutkan ke proses berikutnya.

Apabila pada halaman 'Welcome' memilih *application server*, tahap berikutnya adalah menentukan lokasi instalasi dari *application server*. Untuk merubah lokasi instalasi yang telah ada, klik tombol 'Browse…'. Klik tombol 'Next >' untuk melanjutkan ke proses berikutnya.



Gambar 1.11 Arsitektur Teknologi Java

Halaman License Agreement



Gambar 1.11 Arsitektur Teknologi Java

Halaman Application Server Installation Folder

Halaman berikutnya yang tampil adalah halaman 'Summary'. Klik tombol

'Install' untuk melanjutkan ke proses installasi.

Gambar 1.11 Arsitektur Teknologi Java

Halaman Summary

Halaman Proses Instalasi NetBeans IDE

Gambar 1.11 Arsitektur Teknologi Java

Setelah proses instalasi selesai, akan tampil halaman hasil yang berisistatus hasil instalasi. Klik tombol 'Finish'.

Gambar 1.11 Arsitektur Teknologi Java

Halaman Hasil

Menambah Projek Baru (New Project)

Untuk membuat aplikasi menggunakan Bahasa Pemrograman Java, terlebih dahulu membuat projek baru (*new project*). Berikut langkah-langkah membuat projek baru :

Pada layar dekstop, double click NetBeans IDE shortcut untuk menjalankan software NetBeans IDE. Selain itu, dapat pula melalui menu windows :

NetBeans IDE shortcut

Pada tampilan utama software NetBeans, pilih menu 'File' → 'New Project'.

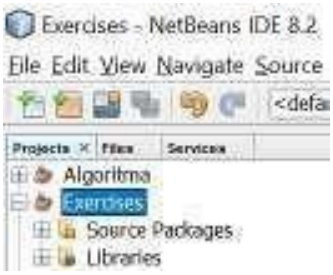
Menu ‘New Project’

Pada halaman ‘New Project’, pilih ‘Java’ dibagian Categories dan pilih ‘JavaApplications’ dibagian Projects kemudian klik tombol ‘Next >’.

Pada Project Name Isikan nama projek akan dibuat contoh ‘Exercises’.Hilangkan centang pada ‘Create Main Class’, kemudian klik ‘Finish’.

Catatan : nama projek tidak boleh menggunakan spasi

Projek baru akan muncul pada jendela ‘Projects’ (umumnya disisi sebelah kiri).



Gambar 1.11 Arsitektur Teknologi Java

Apabila jendela ‘Projects’ tidak muncul/tampil pada IDE Netbeans, jendela

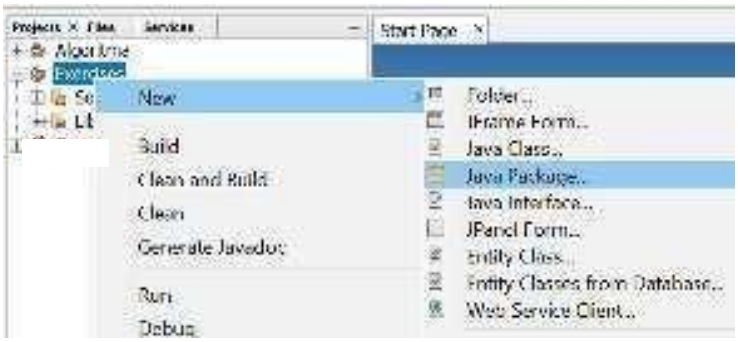
‘Projects’ dapat ditampilkan dengan cara : klik menu ‘Window’ ▯ ‘Projects’

Gambar 1.11 Arsitektur Teknologi Java

Menambah Paket Baru (*New Package*)

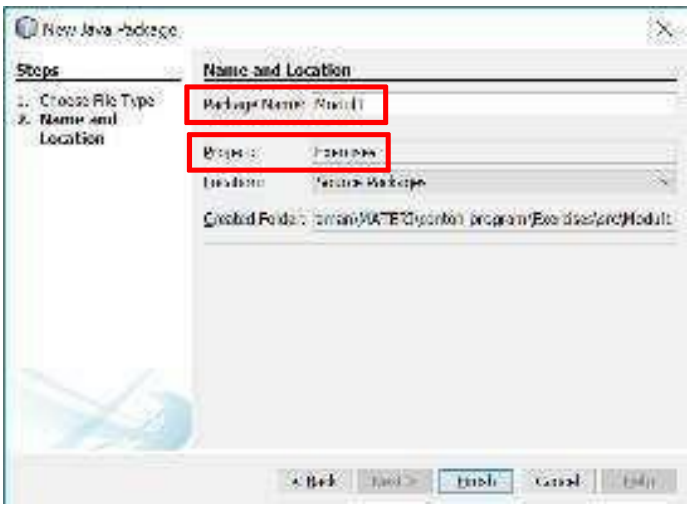
Paket (*Package*) digunakan untuk membuat folder yang dapat mempaketkanfile-file yang sejenis. Berikut langkah-langkah untuk menambah paket baru :

Klik kanan pada projek ‘Exercises’ ▯ New ▯‘Java Package’.



Gambar 1.11 Arsitektur Teknologi Java

Tentukan Package Name, sebagai contoh ‘Modul1’. Projek ‘Exercises’ yang tampil untuk memastikan bahwa paket terdapat dalam projek tersebut. Klik tombol ‘Finish’.



Gambar 1.11 Arsitektur Teknologi Java

Catatan : nama paket tidak boleh menggunakan spasi.

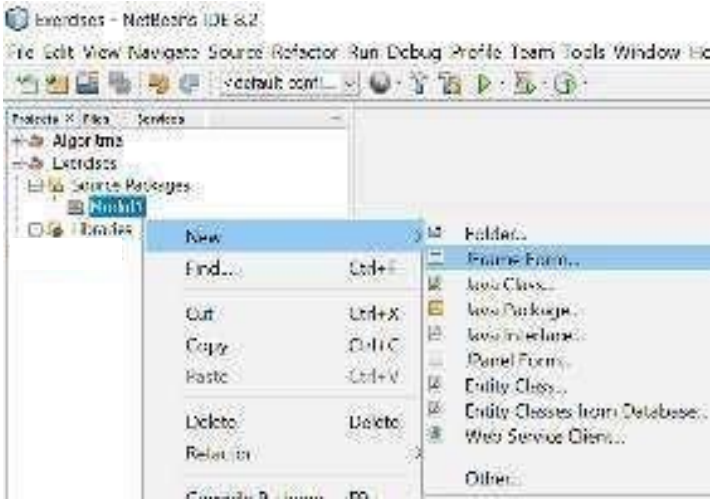
Paket ‘Modul1’ akan muncul didalam projek ‘Exercises’.

Gambar 1.11 Arsitektur Teknologi Java

Menambah Frame Form

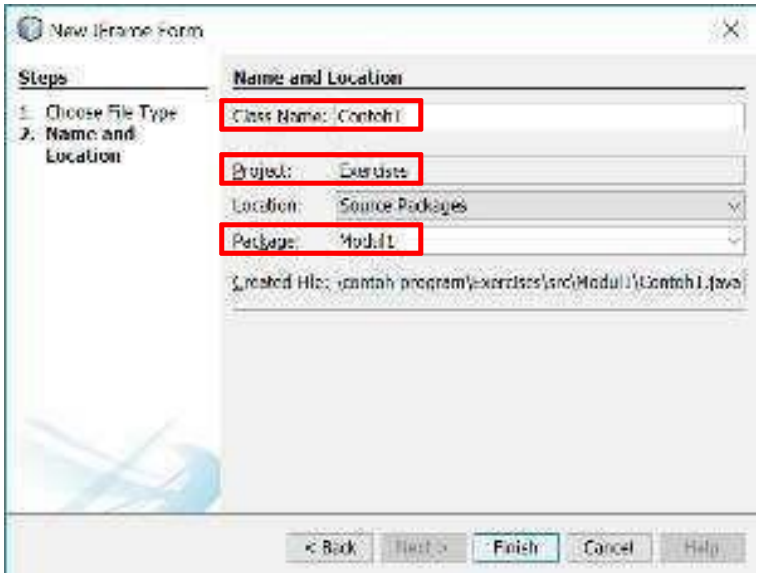
Menambah Frame Form dapat dilakukan pada paket (package) tertentu atau*default package*. Pada contoh di bawah ini, Frame Form akan ditambahkan pada paket “Modul1”. Langkah-langkah untuk menambah Frame Form adalah sebagai berikut :

Klik kanan pada package “Modul1” → New → Jframe Form…



Gambar 1.11 Arsitektur Teknologi Java

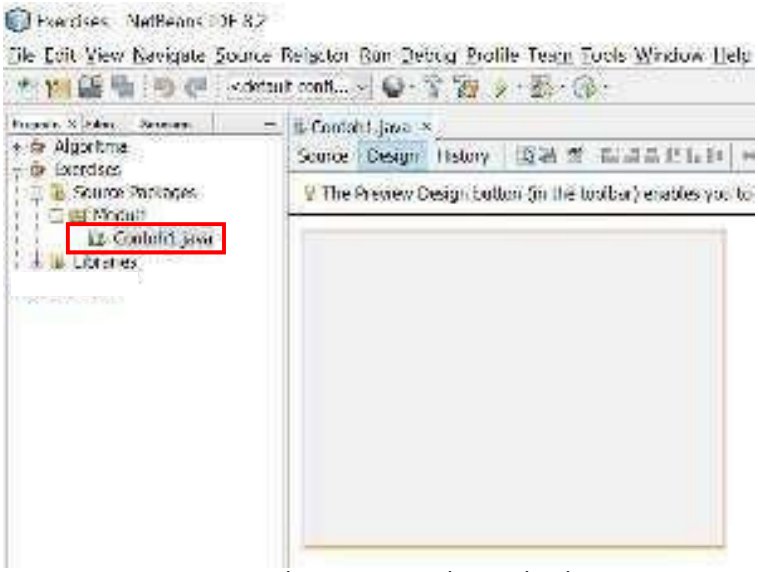
Tentukan Class Name, sebagai contoh ‘Contoh1’. Project ‘Exercises’ dan package ‘Modul1’ yang tampil untuk memastikan bahwa Frame Form terdapat dalam projek dan paket tersebut. Klik tombol ‘Finish’. *Catatan : nama class tidak boleh menggunakan spasi.*



Gambar 1.11 Arsitektur Teknologi Java

Form ‘Contoh1’ akan muncul didalam paket ‘Modul1’ dengan nama

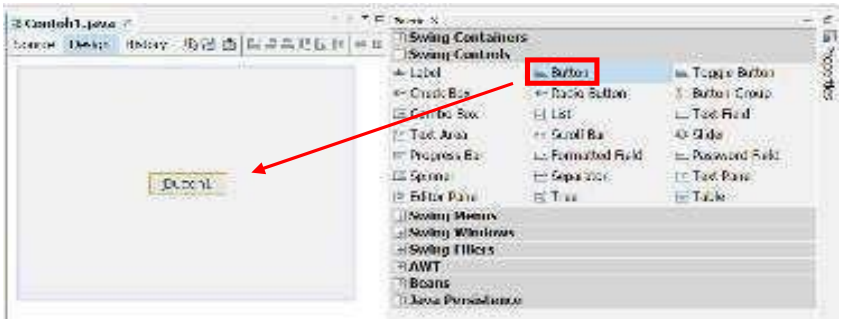
'Contoh1.java'.



Gambar 1.11 Arsitektur Teknologi Java

Merancang Aplikasi

Masukkan komponen 'Button' kedalam form. Pada jendela 'Palette' pilih kategori 'Swing Controls', kemudian klik kiri 'Button'. Arahkan pointer mouse kedalamform untuk menentukan posisi 'Button', kemudian klik kiri sekali lagi.



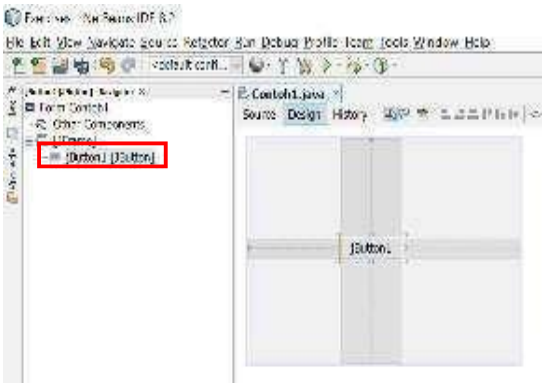
Gambar 1.11 Arsitektur Teknologi Java

Catatan!

Komponen 'Button' (tombol) merupakan komponen eksekutor yang digunakan untuk menjalankan suatu proses tertentu pada aplikasi yang dibangun dengan cara klik kiri.

Pada jendela 'Navigator', muncul komponen 'Button' yang dimasukkan kedalam

form dengan nama standar 'jButton1'.

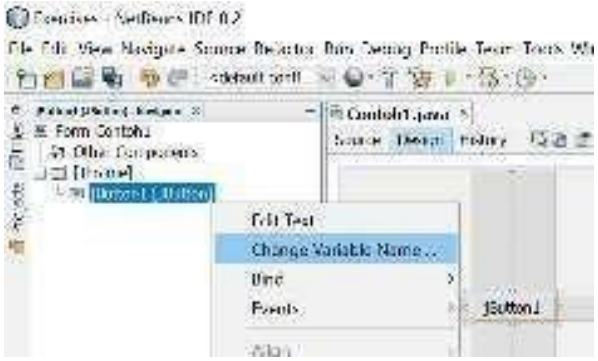


Gambar 1.11 Arsitektur Teknologi Java

Setiap komponen didalam form memiliki nama (*variable name*) untuk membedakan komponen satu dengan komponen lainnya dan memudahkan programer untuk mengoperasikan komponen melalui kode program.

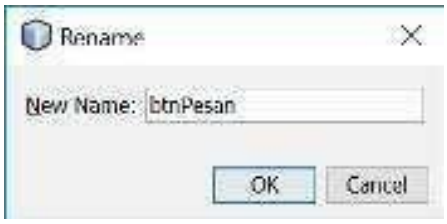
Untuk mengubah nama variabel dari komponen, klik kanan 'jButton1' pada

jendela navigator kemudian klik 'Change Variable Name…'



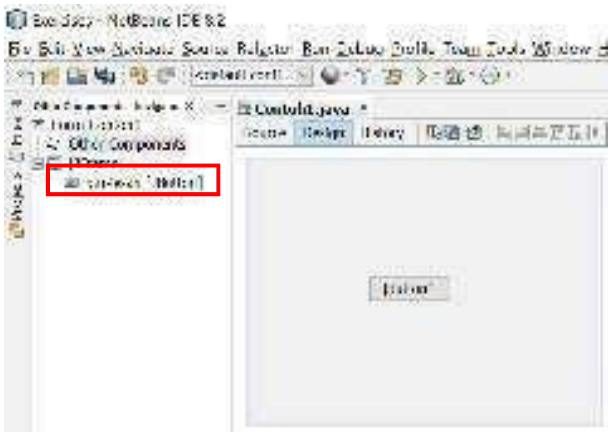
Gambar 1.11 Arsitektur Teknologi Java

Ubah menjadi 'btnPesan'. Klik tombol OK.



Gambar 1.11 Arsitektur Teknologi Java

Nama variabel komponen akan berubah pada jendela navigator



Gambar 1.11 Arsitektur Teknologi Java

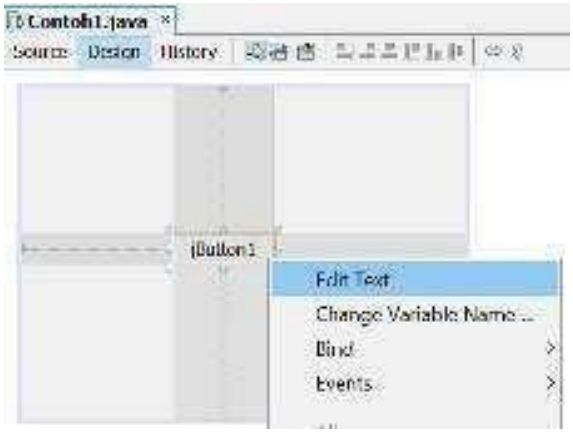
Pemberian nama variabel pada komponen bermanfaat untuk memudahkanprogramer dalam :

Mengenali dan membedakan setiap komponen yang digunakan.

Menulis kode program didalam suatu komponen.

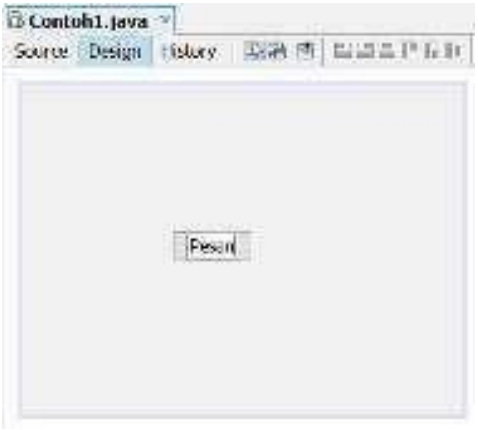
Memanipulasi data yang dimasukkan dari komponen atau untuk ditampilkanke komponen melalui kode program.

Untuk mengubah teks button, klik kanan ‘btnPesan’ pada jendela navigator atauklik kanan komponen pada form, kemudian klik ‘Edit Text’.



Gambar 1.11 Arsitektur Teknologi Java

Ubah menjadi ‘Pesan’ kemudian tekan enter.



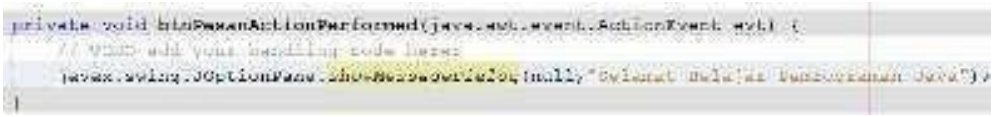
Gambar 1.11 Arsitektur Teknologi Java

Double click button 'Pesan' pada form sehingga muncul tampilan kode program

untuk menulis kode program pada button 'btnPesan'

Gambar 1.11 Arsitektur Teknologi Java

Tuliskan kode program berikut:



Gambar 1.11 Arsitektur Teknologi Java

Untuk menjalankan program, tekan tombol 'shift + F6' pada keyboard atau

tombol run project pada toolbar.

Pada program yang berjalan (runtime) klik tombol 'Pesan', kemudian munculkotak pesan yang berisi 'Selamat Belajar Pemrograman Java'.

Gambar 1.11 Arsitektur Teknologi Java

BAB 6

JAVA SWING SEDERHANA

6.1 Struktur Kontrol

Tujuan struktur control pada bab ini adalah bagaimana kita membuat program yang mengunakan model struktur control bagaimana cara menggunakan percabangan dan pernyataan-pernyataan.

Uji coba Select if.java

```
public class seleksi_if {
public static void main (String | args) {
int jmlhmhs   105;
int dayatampungkelas=100;
if (jmlhmhs>100) {
System.out.println ( "Kelas melebihi kapasitas" );
}
}
```

Yang di dapat :

kelas melebihi kapasitas

keterangan :

Fungsi struktur control di digunakan untuk pernyataand dalam dua kondisi yang berbeda jika mengunakan statemen tambahan else if maka itu bukan if lagi.

if

Else

If

If (kondisi) {

//bagan pernyataan jika jalan,dalam kondisi true}

Ketidak samaan pada konsidi if yang ada pada java script dibandingkan suatu bahasa pemrograman yang lainnya contoh kondisional ba ha s a Java hanya menghasilkan s ua t u nilai Boolean

p i l i han (benar atau salah) .

bahasa code c dan c++, nilai balik berupa type int

Uji coba contoh 2 (seleksi_if_else.java)

```
public class seleksi_if_else {
public static void main (String[] args) {
int totalbelanja = 100000;
int jmlh_brg_dibeli = 2;
double diskon = 0.25;
double hrg_brg_stlh_diskon;
if (totalbelanja>50000 && jmlh_brg_dibeli>1)
{hrg_brg_stlh_diskon = totalbelanja-(totalbelanja*diskon);
System.out.println ("Harga barang setelah diskon :"+hrg_brg_stlh_diskon);
}
else { System.out.println( "Maaf anda belum dapat diskon"); }
}
}
```

Hasil :

Harga barang setelah diskon : 75.000.0

keterangan :

pembahasan if -else berfungsi mengatur saat kondisi di jalanakn dan memiiki nilai benar atau salah

pembahasan statement :

if (kondisi) {

// blok keterangan keteranga di jalankan, kondisi true}

Else {

//blok keterangan di jalankan, bila kondisi false }

Uji coba 3 (seleksi_if_bersarang.java)

```
                public class seleksi_if_bersarang {
public static void main (String[] args) {

                int nilai_angka = 75;
                if (nilai_angka>=85 && nilai_angka <=100) {
                System.out.println ("Nilai Huruf A"); }
                else if (nilai_angka >=70 && nilai_angka <85) {
                System.out.println ("Nilai Huruf B"); }
                else if (nilai_angka >=55 && nilai_angka <70) {
                System.out.println ("Nilai Huruf C"); }
                else if (nilai_angka ==50 && nilai_angka <55) {
                System.out.println ("Nilai Huruf D"); }
                else {
                System.out.println("Nilai Huruf E"); }
                }
                }
```

Hasil :

Nilai Huruf B

Keterangan :

keterangan if-else if datap melakukan pengaturan statement yang running ketika kondisi berupa suatu pilihan bentuk pernyataan lain.

```
if (kondisi x) {
// pernyataan yang dijalankan, apabila kondisi x benar }
else if (kondisi y) {
// pernyataan yang dijalankan, apabila kondisi y benar }
else if (kondisi z) {
// pernyataan yang dijalankan, apabila kondisi z benar }
else {
// pernyataan yang dijalankan untuk kondisi selain itu }
```

Uji coba 4 (seleksi_switch.java)

```
import java.io.BufferedReader;
import java.io.InputStreamReader;
import java.io.IOException;

public class seleksi_switch {
public static void main (String[] args){
BufferedReader InputData = new BufferedReader (new InputStreamReader
(System.in));
String angkaInput= null;
System.out.print ("Masukkan kode hari [1.  7] : " );
try {
angkaInput = InputData.readLine();
}
catch (IOException e ){
System.out.println("Error");
}
int Hari=Integer.valueOf(angkaInput).intValue();
switch (Hari){
case 1:System.out.println("Minggu"); break;
case 2:System.out.println("Senin"); break;
case 3:System.out.println("Selasa"); break;
case 4:System.out.println("Rabu"); break;
case 5:System.out.println("Kamis"); break;
case 6:System.out.println("Jum'at"); break;
case 7:System.out.println("Sabtu"); break;
default :System.out.println("Kode hari yang anda masukkan salah"); }
}
}
```

Hasil :

Masukan kode hari [1,7] : 4

Rabu

Keterangan :

Penulisan code switch membuat jalur eksekusi, control switch bekerja dengan type data , byte, short, char dan int dari semua itu merupakan type data jenis primitive.

Bentuk statement switch :

Switc (variabel) {

Case : <pernyataan>

Default :

}

```
switch(variabel) {
case [data] : <pernyataan> // dieksekusi apabila seleksi pada variabel sesuai
dengan data (bernilai true)
default : <pernyataan> // dieksekusi apabila seleksi pada variabel sesuai
dengan data (bernilai false)
}
```

Pembahasan if, ada beberapa pembahasan code struktur switch control akan di eksekusi tanpa perlu symbol kurawal ({}).

Suatu program di atas switch control akan mendapatkan hasil analisis dari masukan yang awalnya angka akan berubah menjadi sebuah type data string, seperti pada contoh int hari = integer.valueof(angkainput).intvalue().

Suatu variabel bernilai angka inputanya berupa type data string di edit kedalam type data integer yang di tambung pada variabel, pernyataan tersebu dalam switch control nantinya .

pengujian 5 (loop_for.java)

```
public class loop_for {
public static void main (String[] args){
int i;
for(i=1;i<5;i++)
{
  if(i%2==0)
       {System.out.println(i+" adalah Genap");}
    else
       {System.out.println(i+" adalah Ganjil");}
}
}
}
```

Hasil :

Sama dengan ganjil

Sama dengan genap

Sama dengan ganjil

Sama dengan genap

keterangan :

For disebut juga dengan for loop, because bermanfaat untuk proses looping atau suatu pengulangan Bentuknya sebagai berikut :

For (inisialisasi ; kondisi ;step_ekspresi) {

Pernyataan;

}

dimana,

inisialisasi variable loop, kondisi membandingakan loop variabel dengan nilai batas.

setiap ekspresi dari loop melakukan pembaruan variabel loop tersebut, biasanya yang digunakan tambahan pengurangan.

untuk example pada pembahasan :

for (i=1;i<5+=)

Uji coba 6 (loop_while.java)

```
Public class loop_while{

Public static void main(String[]args){

Int x=20;

While (x>10){

System.out.println("Maka x : " + x) ;

x--;}

}

}
```

Hasil :

Maka nya x = 20

Maka nya x = 19

Maka nya x = 18

Maka nya x = 17

Maka nya x = 16

Maka nya x = 15

Maka nya x = 14

Maka nya x = 13

Maka nya x = 12

Maka nya x = 11

Bentuk kondisi:

```
While (kondisi ) {
Pernyataan ;}
```

Uji coba 7 (loop_do_while.java)

```
Public class loop_do_while {
Public static void main (String[] args){
Int i=1;
Do
{
System.out.println("jumlah 11;"+i);
I++;
}
While (i<20);
}
}
```

Hasil :

jumlah I = 11

jumlah I = 12

jumlah I = 13

jumlah I = 14

jumlah I = 15

jumlah I = 16

jumlah I =17

jumlah I = 18

jumlah I = 19

Keterangan :

Pembahasan do while loop sama denga pambahasan while loop. Akan mengeksekusi statemen do while di dalam loop, beberapa kali selama kondisi mempunyai nilai benar. Perbedaan statemen antara while do dan while loop adalah pernyataan statemen while loop akan di eksekusi utama.

```
Do {

Statemen1 ;

Statemen2;

}

While (Boolean-expression) ;
```

Uji coba 8 (pernyataanBreak.java)

```
Public class pernyataanBreak{

Public static void main ( String[]arg){

For(int i=1;<5;i++){

If(i%2==0) break;

System.out.println("Nilai I;'+i);

}

}

}
```

Hasilnya :

Nilai nya I =1

keterangan :

Keterangan code break dimanfaatkan untuk mepause fungsi loop, dimaan pernyattan statement for (int i-1< i++) memilik fungus loop akan berhenti sendirinya apabila statement I bernilai akhiran 5 karena memiliki nilai Boolean salah, namun statemen pernyataan if (i%2==break; , statement tersebut jika I bernilai genap maka loop nya stop.

Uji coba 9 Countineus

```
public class pernyataanContinue {
public static void main(String[] args){
    int i=1;
    while (i<11){
        if(i==7);{
        i++;
        continue;       }
    }
    System.out.println("Nilai i "+ i);
    i++;  }
}
```

Hasil :

Nilai I = 11

Keteragan :

Statement continue atau selanjutnya dimanfaatkan untuk eksekusi pengulalngan atau loop dan hal yang sama dengan code statement brak ; pengunan pengulanan seterusnya.

ujicoba 10 (pernyataan Return.java)

```
public class pernyataanReturn {
   void perkalian(int i,int j){
       int hasilkali=i*j;
       System.out.println("Hasil perkalian :" +hasilkali);
       return;
    }

public static void main(String[] args){

       pernyataanReturn a = new pernyataanReturn();
       a.perkalian(5, 6);
       }
}
```

Hasil :

Hasil perkalian : 30

keterangan :

Keterangan return dapat digunakan sebagai output dari method. Keterangan retun memiiki bentuk memberi suatu nilai dan tidak memberi tambahan nilai. Untuk menghasilkan suatu nilai tambah ikspresi penghasil nilai sesuah menuliskan code return pada program

example, Return

++count ;Atau

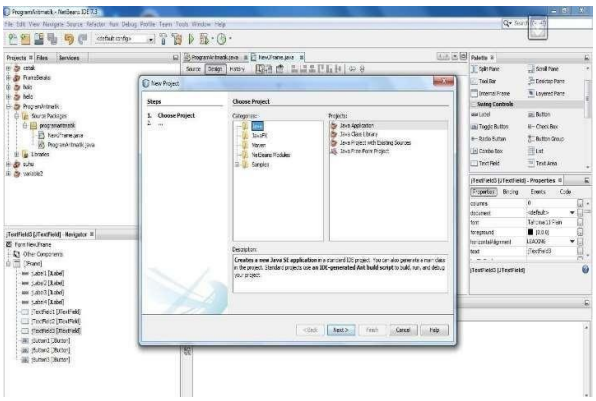
Return “hello” ;

Suatu tipe data yang nilai diberi, typenya harus sama dengan method yang dibuat. Contoh semisal suatu void dideklarasikan, bentuk return tidak bisa memberi nilai.

Return ;

6.2 Java Swing Sederhana

Mula-mula kita buka program Netbeanslalu pilih menufile / baru Project maka akan muncul GUI tampil seperti ini.



Gambar 1.1 Compiler

Pilih pada Java pada Categories serta pada Java Apllication pada Projects. Klik selanjutnya muncul tampilanForm seperti contoh ini

Pada Project Name isi Pembelian Nasi selanjutnya hapus tanda (V) centang di Create Main Class serta Set as Main Projectlalu klik selesai maka akan muncuk tampilan Form seperti contoh :

di isian Class Name isi serta PembelianNasiBungkus lalu tekan selesaimaka akanmuncul view form work

Netbean. Klik p a d a kanan pada bagian data

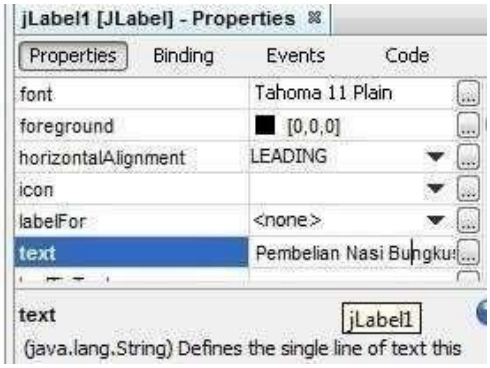
Packages selanjutnya JFrame Form muncul tampilan seperti ini :

Buatlah 5 Label, 4 TextField, dan 3 tombol seting template layaoute seperti image di bawah :

Selanjutnya ganti nama attribute pada label keterangan, TextField tempat masukan, dan tombol untuk contoh gambar di bawah ini :

Gambar 1.1 Compiler

Cara mengubah nya dengan cara di kotakclick sasaran dan kita ubah di bagian propertis nya seperti di bawah ini atau click kanan pada media yg ada di ganti nama nya.



Gambar 1.1 Compiler

Untuk mempermudahkan menginputcoding tiap elemen diubah name nya dengan menekan tomobol kanan untuk action rubah kemudian klik Change Variable Name.

Gambar 1.1 Compiler

Selanjutnya, Langkah selanjutnya adalah melakukan pengcodingan di case ini kita memiliki 3 tujuan tombil button memiliki action sebagai berikut :

Klik tombol jumlah perhitungan terhadap (dua) komponen textfield yaituTextField Jumlah Uang serta TextField Harga Nasi Bungkus serta hasilnya ditampilkan pada TextField Sisa dan Text Field

Kesimpulan.

tombol tombol Hitung berfungsi menjalankan operasinya (

TextFieldHargaNasiBungkus-TextFieldJumlahUang),kemudian

ij jumlahnya = 0 maka lunas,

if jumlahnya < 0, maka pinjam

jika jumlahnya > 0 maka menabung.

Untuk Pengcodingannya klik 2 kali tombol Hitung selanjutnya ketikkan kodeseperti ini :

```
private void btnhitungActionPerformed(java.awt.event.ActionEvent evt)
{     // TODO add your handling code here: int jumlahuang =
Integer.parseInt(txtjumlahuang.getText()); int harganasibungkus =
Integer.parseInt(txtharganasibungkus.getText(
));
int sisauang = jumlahuang -harganasibungkus;
String hasil = String.valueOf(sisauang);txtsisauang.setText(hasil);
if (sisauang==0) txtkesimpulan.setText("LUNAS");
else if (sisauang<0) txtkesimpulan.setText("NGUTANG");
else
txtkesimpulan.setText("NABUNG");
```

Else

Txtkesimpulan.settext(“NABUNG”);

}

Tombol delete berfungsi untuk action hapus text pada semua file textfield sehingga menjadikan semua file textfield akan terisi kosong lagi. Untuk pengkodinganya tekan 2 tombol hapus selanjutnya masukan kode seperti pada di bawah ini ;

```
private void btnhapusActionPerformed(java.awt.event.ActionEvent evt) {


// ketik program di bawah todo add ini:
txtjumlahuang.setText(""); txtharganasibungkus.setText("");
txtsisauang.setText(""); txtkesimpulan.setText("");
}
```

tombol action keluar.Untuk Pengcodingannyaklik

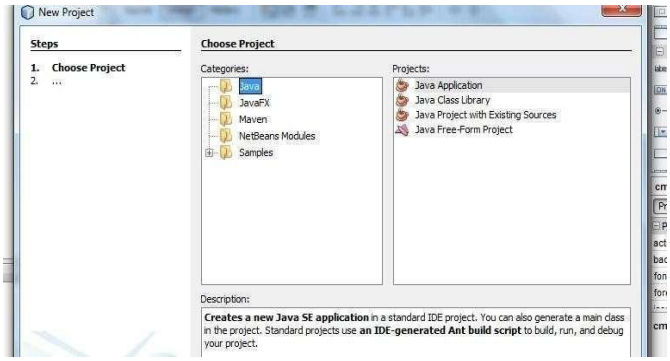
2 kali tombol Keluar lalu ketikkan kodeseperti ini:

```
private void btnKeluarActionPerformed(java.awt.event.Acti
onEvent evt) {
// ketik program code dibawah tuisan todo add dibawah ini :
System.exit(0);
}
```

Jika Coding selesai di masukan maka tahap terakhir adalah m e l a k u k a n running. Untuk Running biasanyatekan F6. makahasilnyabisadapatdijalankan

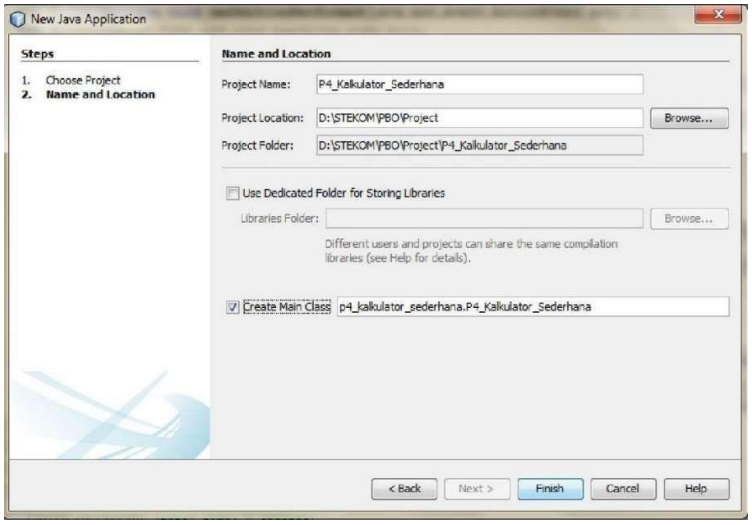
Gambar 1.1 Compiler

6.3 Aplikasi Sederhana Kalkulator



Gambar 1.1 Compiler

Jalankan NetBean lalu New Project



Gambar 1.1 Compiler

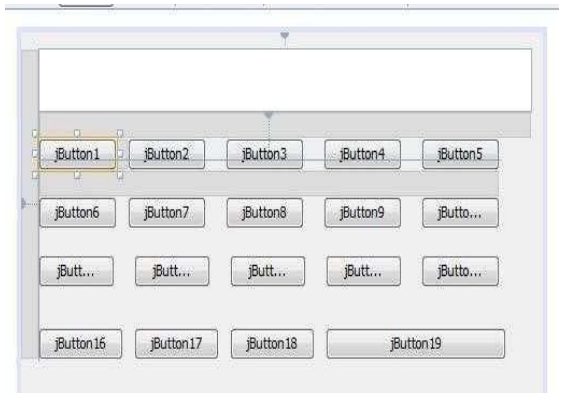
Lalu buatlah Project Name Kalkulator Sederhana lalu Finish

Lalu pada netbean pilin New-> Jframe From pada Source Packages lalu beri nama class yang berbeda dengan nama project

Gambar 1.1 Compiler

Desain Form Kalkulator Sebagai berikut

Gambar 1.1 Compiler

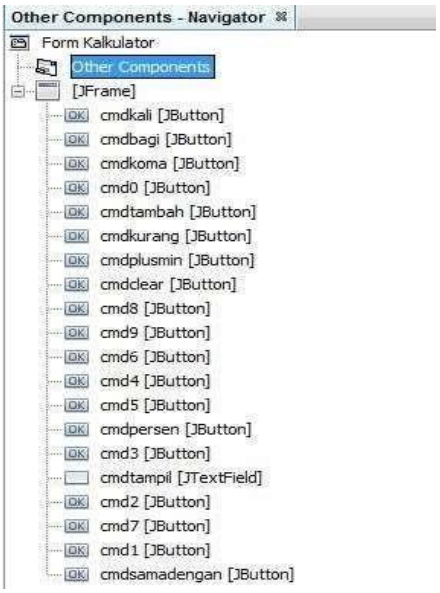


Gambar 1.1 Compiler

Gambar 1.1 Compiler

Kemudian clik kanan pada tombol yg di pilih -> Change Variable name untuk mempermudah pengkodingan

Untuk Text Field =cmdtampil (+)=cmdtambah (-)=cmdkurang (/)=cmdbagi (*)=cmdkali (%)=cmdpersen (+/-)=cmdplusminus(=)=cmdsamadengan (C)=cmdclear (,)=cmdkoma



Kemudian deklarasikan jenis data yg di pakai

```
public class kalkulator_sederhana extends javax.swing.JFrame {
String angka;
Double total, angka1, angka2;
int pilih;

    /**
     * Creates new form kalkulator_sederhana
     */
    public kalkulator_sederhana() {
        initComponents();
        angka="";
    }
```

Ketikan Listing kode perintah pada tombol 0-9Click 2x pada tombol 0 kemudian ketik -> angka+="0";

cmdtampil.setText("0");

Hasil :

```
    private void cmd0ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        angka+="0";
        cmdtampil.setText("0");
    }
```

Untuk tombol 1

```
    private void cmd1ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        angka+="1";
        cmdtampil.setText("1");
    }
```

Setelah tombol 0-9 di input perintah coba jalan kan Run dah cobalah menekan tombol 0-9 jikakeluar pada Text Field maka perintah anda benar

Pada Tombol +

```
    private void cmdtambahActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        angka1=Double.parseDouble(angka);
        cmdtampil.setText("+");
        angka="";
        pilih=1;
    }
```

Pada tombol –

```
    private void cmdkurangActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        angka1=Double.parseDouble(angka);
        cmdtampil.setText("-");
        angka="";
        pilih=2;
    }
```

Tombol C

```
    private void cmdclearActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        angka1=0.0;
        angka2=0.0;
        total=0.0;
        angka="";
        cmdtampil.setText("");
    }
```

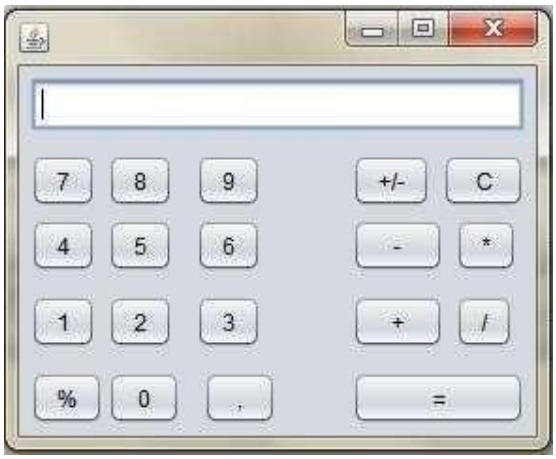
Untuk Tombol =

Switch (pilih)

{

```
case 1: angka2=Double.parseDouble(angka); total=angka1+angka2;
angka=Double.toString(total); cmdtampil.setText(angka);

break; defau

lt: break;

case 2: angka2=Double.parseDouble(angka); total=angka1-angka2;
angka=Double.toString(total); cmdtampil.setText(angka);

break;
```

}



Gambar 1.1 Compiler

BAB 7
PENGAKSESAN BASIS DATA

Pada pembahasan di bab sebelumnya, kita telah berhasil membuat aplikasi berbasis GUI yang dapat menyimpan dan menampilkan data ke dalam tabel. Akan tetapi, masih ada kekurangan pada aplikasi tersebut, yakni data yang disimpan dalam aplikasi tersebut masih bersifat volatile, artinya ketika aplikasi ditutup, maka data yang tadinya disimpan pun **hilang**.

Pada praktiknya, sebuah aplikasi seringkali membutuhkan sistem penyimpanan data yang rapi dan terorganisasi dengan baik. Oleh karena itu, penggunaan basis data sangat diperlukan untuk dapat mengatasi hal tersebut. Integrasi antara aplikasi berbasis Java dengan **DBMS** (Database Management System) merupakan hal yang sudah lazim, dan dapat di lakukan menggunakan library **jdbc** yang biasanya sudah tersedia dalam IDE Netbeans. Pada bab ini, kita akan membahas mengenai proses integrasi antara aplikasi Java dengan DBMS MySQL.

7.1 Persiapan Basis Data

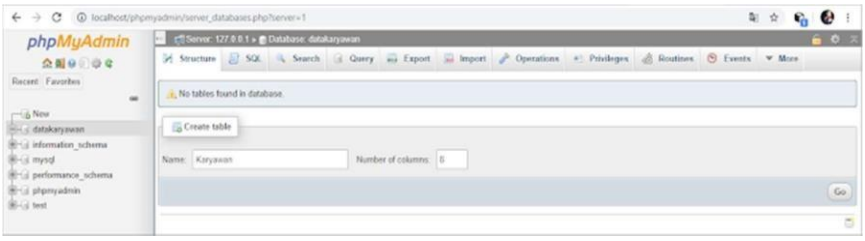
Untuk menggunakan basis data MySQL, salah satu alternatif yang mudah adalah dengan menginstall aplikasi XAMPP yang di dalam nya telah terintegrasi fitur phpMyAdmin yang merupakan platform MySQL berbasis GUI, sehingga memudahkan kita untuk merancang dan membuat basis data relasional. Anda dapat mengunduh aplikasi tersebut pada link: https://www.apachefriends.org /download.html.

Setelah Anda berhasil mengunduh dan melakukan instalasi XAMPP pada komputer Anda, maka buka XAMPP-Control Panel, lalu tekan tombol "**Start**" pada fitur "**Apache**" dan "**MySQL**""seperti yang terlihat pada gambar di bawah.

Gambar 7.1 XAMP

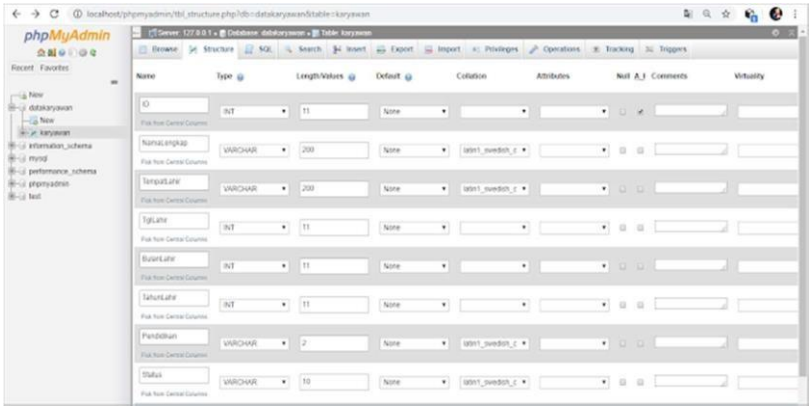
Setelah itu, buka aplikasi web browser pada komputer Anda, lalu ketik URL: **localhost/phpmyadmin**. Apabila Anda masuk ke halaman mesin pencari Google, coba nonaktifkan dahulu sementara jaringan **internet** Anda. Jika sudah, Anda bisa mulai membuat basis data baru dan membuat tabel baru seperti di bawah ini. Pada contoh ini, kita membuat basis data dengan nama

"**dataKaryawan**" dan membuat sebuah tabel bemama "**Karyawan**" yang terdiri dari delapan buah kolom. Perhatikan gambar dibawah.



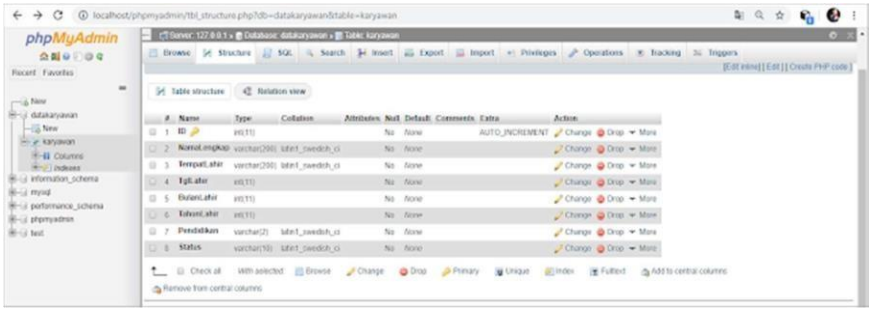
Gambar 7.2 MYSQL

Kemudian buatlah kedelapan kolom pada tabel tersebut yang terdiri dari: ID, NamaLengkap , TempatLahir, TglLahir, BulanLahir, TahunLahir, Pendidikan, dan Status. Jadikan kolom/ atribut ID sebagai primary key dan set Auto increment.



Gambar 7.3 Dasbor Mysql

Tekan tombol "**Go**" akan tampil seperti gambar berikut

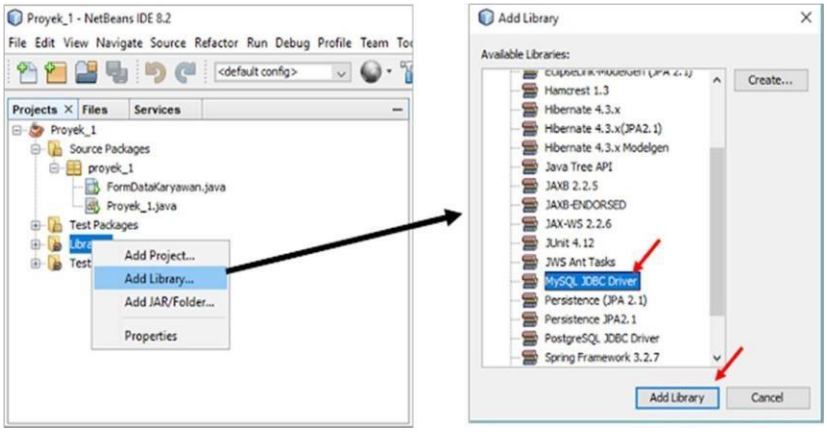


Gambar 7.4 Tampilan Database

Jika sudah, maka selanjutnya kita lakukan pengetikan kode di dalam Java.

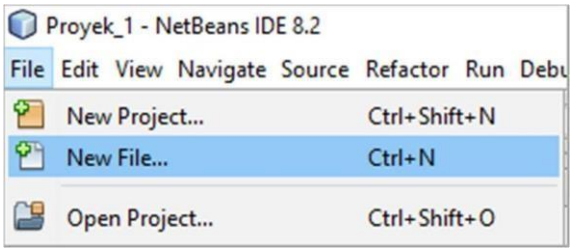
7.2 Membangun Koneksi dengan JDBC

Pada package Proyek_ 1 yang telah kita buat pada bab sebelumnya, Lihat tab "**Projects**" pada bagian kiri Netbeans, lalu klik kanan pada "**Libraries**'', kemudian pilih "**Add Library**". Setelah itu, pilih "**MySQL JDBC Driver**" pada kolom "**Available Libraries**".



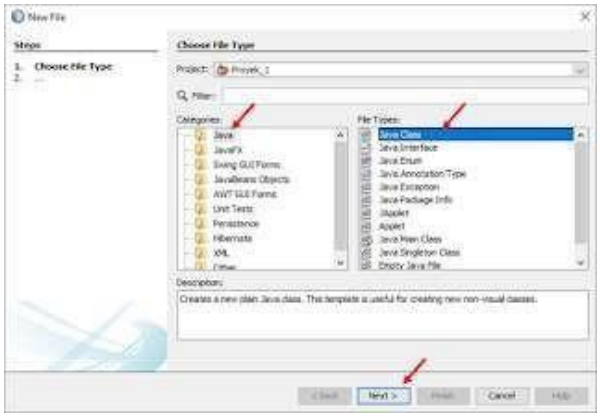
Gambar 7.5 Add Library Mysql

Berikutnya, buka menu "**File → New File**",



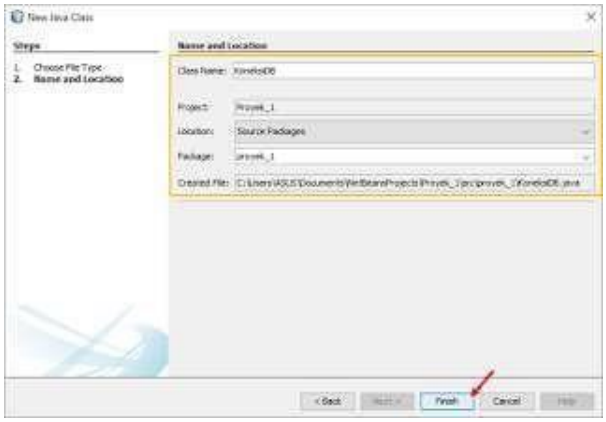
Gambar 7.6 Compiler

lalu pilih "**Java**" pada bagian Categories dan pilih "**Java Class**" pada bagian File Types,



Gambar 7.8 Compiler

lalu klik tombol "**Next**". Setelah itu, pada bagian Class Name, isi dengan "**KoneksiDB**".



Gambar 7.9 Compiler

Pada kode program dari kelas **KoneksiDB**, kita tambahkan kode program seperti di bawah ini:

```
package_proyek_1;


import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;


public class KoneksiDB{
 //Membuat variabel bertipe connection
 private static Connection koneksi;


 public static Connection getKoneksi(){
  //method ini berfungsi untuk membuat koneksi ke MYSQL
  if(koneksi == null){
   try{
    String url = "jdbc:mysql://localhost:3306/datakaryawan";
    String username = "root";
```

```
  String password = "";


  DriverManager.registerDriver(new com.mysql.jdbc.Driver());


  koneksi = DriverManager.getConnection(url,username,password);
 }catch (SQLException e){
  System.out.println("Gagal membuat koneksi");
 }
}
 return koneksi;
}
}
```

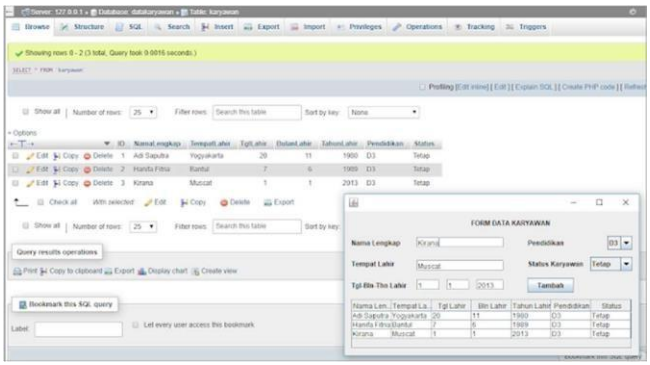
Pada kode program di atas, kita deklarasikan sebuah variabel bertipe Connection. Variabel ini kita beri nama "koneksi". Kemudian kita buat sebuah method bemama get Koneksi ( ) yang berfungsi untuk membuat koneksi antara aplikasi Java yang kita buat dengan DBMS MySQL pada phpMyAdmin. Pada method tersebut, kita menginisialisasi tiga buah variabel, yakni url, username, dan password. Variabel url berfungsi menyimpan lokasi penyimpanan database di dalam local host. Variabel username digunakan untuk menyimpan usemame dari DBMS yang kita gunakan, dan variabel password digunakan untuk menyimpan password dari DBMS yang kita gunakan. Apabila tidak ada password yang diset, maka variabel password cukup diisi dengan karakter kosong (" ").

Selanjutnya, kita tambahkan baris kode program di bawah ini untuk menyimpan data ke database pada method jButtonlActionPer formed ( ). Letakkan kode di bawah ini di bawah kode program yang sudah ada pada method jButtonlAction Performed ( ) .

```
try{
 Connection conn = KoneksiDB.getKoneksi();
 String sql = "INSERT INTO KARYAWAN(NamaLengkap,TempatLahir,Tgllahir,"
 + "BulanLahir,TahunLahir,Pendidikan,Status) VALUES (?,?,?,?,?,?,?)";


 PreparedStatement p = conn.prepareStatement(sql);
 p.setString(1, NamaLengkap); //kolom.ke-1 diset auto increment
 p.setString(2, TempatLahir);
 p.setInt(3, tgl);
 p.setInt(4, bln);
```

```
p.setInt(5, thn);
p.setString(6, Pendidikan);
p.setString(7, Status);

p.executeUpdate();
p.close();
}catch(SQLException e){
System.out.println("Terjadi error");
}
```

Apabila koneksi database berhasil dibangun dan proses INSERT query berhasil dilakukan, maka seharusnya data yang di inputkan melalui aplikasi Java yang kita buat juga tersimpan di dalam database MySQL. Perhati kan gambar di bawah ini.



Gambar 7.10 Compiler

Namun , jika Anda cermati, ternyata ada sedikit masalah pada aplikasi yang telah kita bangun sejauh ini. Ketika Anda menutup aplikasi tersebut, lalu membukanya lagi, ternyata data karyawan yang telah kita simpan sebelumnya tidak tampil pada tabel. Kenapa hal ini bisa terjadi? Karena kita belum melakukan load data dari database setiap kali aplikasi dibuka.

Oleh karena itu, kita perlu membuat sebuah method baru lagi di dalam kelas **FormDataKaryawan** yang berfungsi untuk melakukan load data dari database ke dalam tabel.

```
public void loadData(){
 try{
  //Membuat koneksi
  Connection conn = KoneksiDB.getKoneksi();
  Statement s = conn.createStatement();
```

```
   //Membuat query SELECT
  String sql = "SELEC * FROM KARYAWAN";
  ResultSet r = s.executeQuery(sql);


  //Membaca data di dalam database baris per baris
  while(r.next()){
   //Membuat objek 'obj' untuk menampung data yang dibaca dari DB
   Object[] obj = new Object[7];
   obj[0]  =  r.getString("NamaLengkap");
   obj[1]  =  r.getString("NamaLengkap");
   obj[2] = r.getInt("TglLahir");
   obj[3] = r.getInt("BulanLahir");
   obj[4] = r.getInt("TahunLahir");
   obj[5] = r.getString("Pendidikan");
   obj[6] = r.getString("Status");
     //tampilkan satu baris data ke dalam tabel
   model.addRow(obj);
     }
    //menutup hasil penelusuran dan koneksi
  r.close();
  s.close();
 }catch(SQLException e){
  System.out.println("Terjadi error");
 }
}
```

keterangan program :

Setelah itu, kita panggil method loadData ( ) tersebut di dalam constructor public FormDataKaryawan ().

```
public FormDataKaryawan(){
 initComponents();
```

```
//Membuat Table Model
model = (DefaultTableModel) jTable1.getModel1();
//Menambahkan TableModel ke jTable1
jTable1.setModel(model);
//Melakukan load data dari database
loadData();
}
```

Maka, kini setiap kali kita membuka aplikasi FormDataKaryawan, secara otomatis semua data yang tersimpan di dalam database akan dibaca dan ditampilkan di dalam tabel.

7.3 Langkah Membuat Aplikasi Sederhana Database

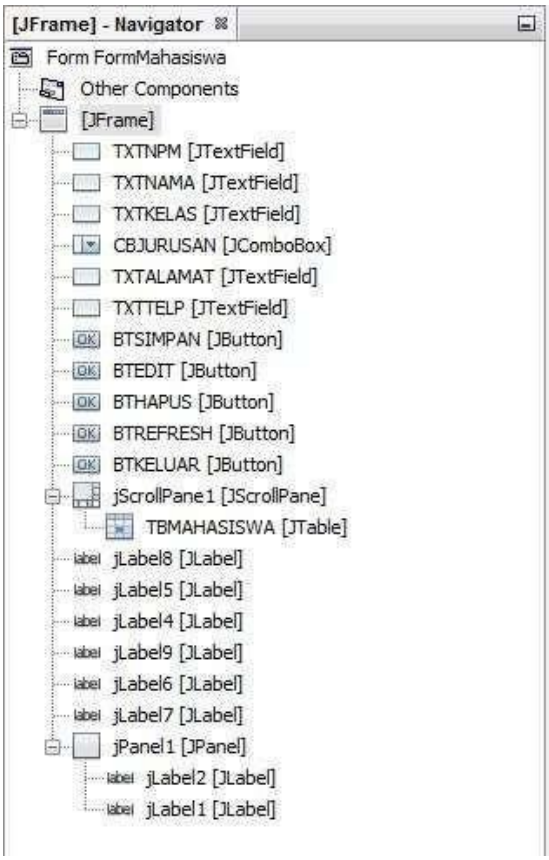
Buat Project Baru, beri nama dengan Mahasiswa

Buat New JFrame From dan beri nama FormMahsiswa

Gambar 7.11 Compiler

Desain di New JFrame From sebagai berikut

Komponen yang Digunakan

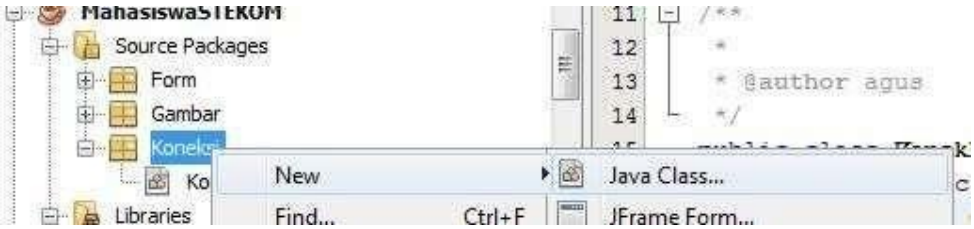


Buat database db_mahasiswa_stekom

#	Name	Datatype	Length/Set	Unsign...	Allow N...	Zerofill	Default
1	**NPM**	**VARCHAR**	**15**	☐	☐	☐	No default
2	NAMA	VARCHAR	50	☐	☑	☐	NULL
3	KELAS	VARCHAR	7	☐	☑	☐	NULL
4	JURUSAN	VARCHAR	25	☐	☑	☐	NULL
5	ALAMAT	VARCHAR	50	☐	☑	☐	NULL
6	TELP	VARCHAR	12	☐	☑	☐	NULL

Tambahkan library mysql

Ketik Kode Program Sebagai berikutBuat koneksi dari Package koneksi



Buat nama file java (KonekDB.java)

```
public class KonekDB {
    private static Connection koneksi;
    public static Connection getkoneksi() {
        if (koneksi==null) {
            try {
                String url=new String();
                String user=new String();
                String password=new String();
                url="jdbc:mysql://localhost:3306/db_mahasiswa_stekom";
                user="root";
                password="";
                DriverManager.registerDriver(new com.mysql.jdbc.Driver());
                koneksi=DriverManager.getConnection(url,user,password);
            }catch (SQLException t) {
                System.out.println("Database tidak tersambung");
            }
        }
     return koneksi;
    }
```

```
public class FormMahasiswa extends javax.swing.JFrame {

    /**
     * Creates new form FormMahasiswa
     */
    private DefaultTableModel model;

    public FormMahasiswa() {
    initComponents();
    model=new DefaultTableModel();
    TBMAHASISWA.setModel(model);
    model.addColumn("NPM");
    model.addColumn("NAMA");
    model.addColumn("KELAS");
    model.addColumn("JURUSAN");
    model.addColumn("ALAMAT");
    model.addColumn("TELP");
    loadData();
    }
```

```
    public void loadData() {
    model.getDataVector().removeAllElements();
    model.fireTableDataChanged();
    try {
        Connection c=KonekDB.getkoneksi();
        Statement s= c.createStatement();
        String sql="SELECT * FROM MAHASISWA";
        ResultSet r=s.executeQuery(sql);

        while (r.next()) {
            Object[] o=new Object[6];
            o[0]=r.getString("NPM");
            o[1]=r.getString("NAMA");
            o[2]=r.getString("KELAS");
            o[3]=r.getString("JURUSAN");
            o[4]=r.getString("ALAMAT");
            o[5]=r.getString("TELP");

            model.addRow(o);
        }
        r.close();
        s.close();
    }catch(SQLException e) {
        System.out.println("GAGAL Load");
    }
}
```

```
public void HAPUSMAHASISWA() {
    String NPM= TXTNPM.getText().trim();

    try {
        Connection c=KonekDB.getkoneksi();
        String sql = "DELETE From MAHASISWA WHERE NPM=?";
        PreparedStatement p=(PreparedStatement) c.prepareStatement(sql);
        p.setString(1, NPM);
        p.executeUpdate();
        p.close();
        System.out.println("SUKSES HAPUS");
        JOptionPane.showMessageDialog(null, "Sukses Hapus " + NPM + " !");
    }catch(SQLException e){
        System.out.println("Terjadi kesalahan");
    }finally{
        loadData();
    }
}
```

```
public void BLANKFORM(){
    TXTNPM.setText(null);
    TXTNAMA.setText(null);
    TXTKELAS.setText(null);
    //CBJURUSAN.;
    TXTALAMAT.setText(null);
    TXTTELP.setText(null);
}
```

```
    private void BTHAPUSActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String NPM=this.TXTNPM.getText();

        if ("".equals(NPM.trim()) || NPM.trim()==null)
        {
            return;
        }
        else{
            HAPUSMAHASISWA();
            BLANKFORM();
        }
    }
```

```
    private void BTKELUARActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        System.exit(0);
    }
```

```
    private void BTSIMPANActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String NPM=TXTNPM.getText();
        String NAMA=TXTNAMA.getText();
        String KELAS=TXTKELAS.getText();
        String JURUSAN=(String)CBJURUSAN.getSelectedItem();
        String ALAMAT=TXTALAMAT.getText();
        String TELP=TXTTELP.getText();

        try {
            Connection c=KonekDB.getkonseksi();
            String sql = "INSERT INTO MAHASISWA VALUES (?,?,?,?,?,?)";
            PreparedStatement p=(PreparedStatement) c.prepareStatement(sql);
            p.setString(1, NPM);
```

```
            p.setString(2, NAMA);
            p.setString(3, KELAS);
            p.setString(4, JURUSAN);
            p.setString(5, ALAMAT);
            p.setString(6, TELP);
            p.executeUpdate();
            p.close();
            BLANKFORM();
            JOptionPane.showMessageDialog(null, "Sukses Simpan NPM =" + NPM + " !");
        }catch(SQLException e){
            System.out.println("Tidak bisa Isi");
            JOptionPane.showMessageDialog(null, "Gagal Simpan NPM =" + NPM + " !");
        }finally{
            loadData();
        }
    }

    private void BTREFRESHActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        BLANKFORM();
    }

    private void BTEDITActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String NPM=TXTNPM.getText();
        String NAMA=TXTNAMA.getText();
        String KELAS=TXTKELAS.getText();
        String JURUSAN=(String)CBJURUSAN.getSelectedItem();
        String ALAMAT=TXTALAMAT.getText();
        String TELP=TXTTELP.getText();

        try {
            Connection c=KonekDB.getkoneksi();
            String sql = "UPDATE MAHASISWA Set NAMA=?,KELAS=?,JURUSAN=?,ALAMAT=?,TELP=? "
                    + "WHERE NPM=?";
            PreparedStatement p=(PreparedStatement) c.prepareStatement(sql);
            p.setString(1, NAMA);
            p.setString(2, KELAS);
            p.setString(3, JURUSAN);
            p.setString(4, ALAMAT);
            p.setString(5, TELP);
            p.setString(6, NPM);
            p.executeUpdate();
            p.close();
            BLANKFORM();
            JOptionPane.showMessageDialog(null, "Sukses update NPM =" + NPM + " !");
        }catch(SQLException e){
            System.out.println("Tidak bisa Isi");
            JOptionPane.showMessageDialog(null, "Gagal update NPM =" + NPM + " !");
        }finally{
            loadData();
        }
    }
```

```
    private void TBMAHASISWAMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        int i=TBMAHASISWA.getSelectedRow();
        if(i==-1)
        {
            return;
        }
        String NPM=(String) model.getValueAt(i, 0);
            TXTNPM.setText(NPM);
        String NAMA=(String) model.getValueAt(i, 1);
            TXTNAMA.setText(NAMA);
        String KELAS=(String) model.getValueAt(i, 2);
            TXTKELAS.setText(KELAS);
        String JURUSAN=(String) model.getValueAt(i, 3);
            //CBJURUSAN.getSelectedItem().equals(JURUSAN);
        String ALAMAT=(String) model.getValueAt(i, 4);
            TXTALAMAT.setText(ALAMAT);
        String TELP=(String) model.getValueAt(i, 5);
            TXTTELP.setText(TELP);
    }
```

BAB 8

LATIHAN - LATIHAN

8.1 Latihan-Latihan

a. Aplikasi MDI

Sesuai namanya, aplikasi MDI (Multiple Document Interface) merupakan aplikasi yang bisa menampilkan beberapa dokumen sekaligus. Contoh aplikasi MDIadalah program aplikasi Micrososft word, Microsoft Excel dan program aplikasi lainnya. Di dalam aplikasi MDI, ada 2 pembagian Form.

- ✓ **Form Induk** : Form terluar yang dijadikan tempat (wadah) untuk menampilkan form yang lain.
- ✓ **Form Anak** : Form yang ditampilkan di dalam form Induk. Form ini terpasang seolah-olah menempel di dalam Form induk dan tidak dapat keluar dari tampilan Form induk.

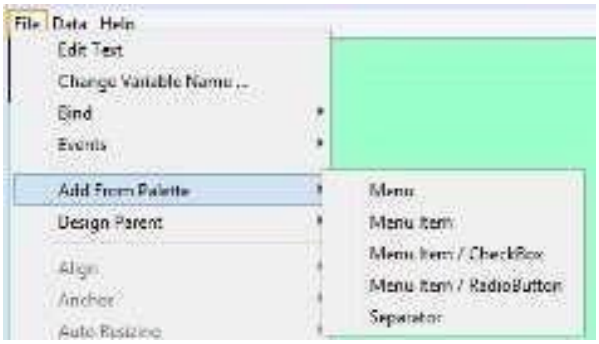
Berikut contoh projek menggunakan MDI :

Tambahkan paket (*Java Package*) pada projek 'Exercises'. Beri nama paket dengan 'Modul7.MDI'.

Tambahkan 'JFrame Form' dengan langkah-langkah seperti pada percobaanmodul 1 (menambah frame form). Beri nama JFrame Form dengan 'FormUtama'.

Tambahkan komponen Desktop Pane dan Menu Bar pada 'FormUtama' sepertipada gambar di bawah ini.

Tambahkan JMenuItem di Menu File dengan Cara klik kanan Menu File



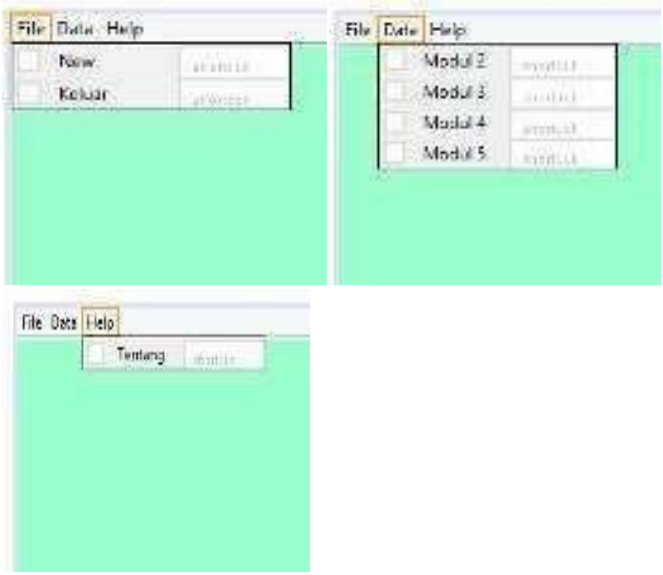
- ✓ AddFrom Palette
- ✓ Menu Item

Gambar 1.1 Compiler

Ganti Text JMenuItem dengan 'New' caranya klik kanan JMenuItem1

EditText

Ulangi Langkah 4 dan 5, buat tampilannya seperti berikut :



Gambar 1.1 Compiler

Buat Variable Global untuk dapat diakses semua method yang ada didalamclass FormUtama.

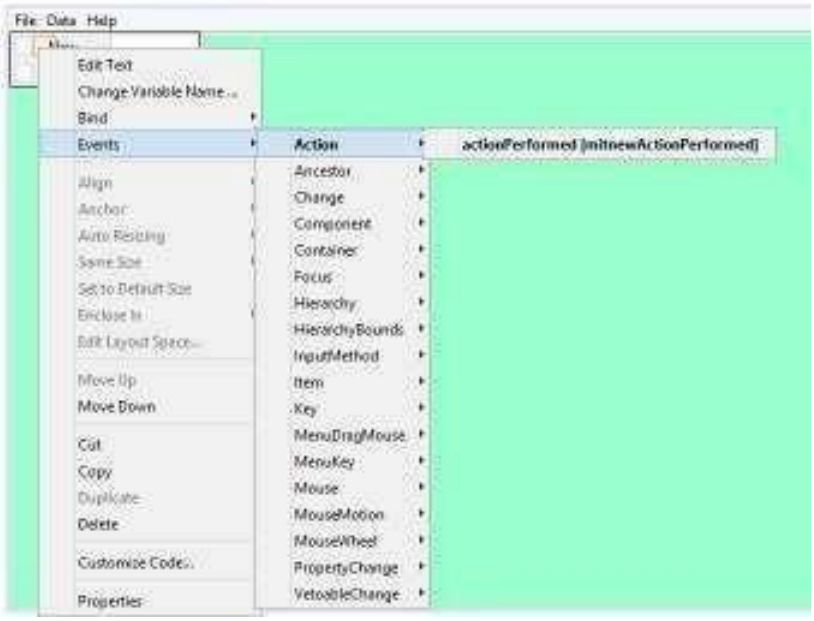
Tambahkan perintah berikut di dalam *constructor* FormUtama :

Buat Method baru dengan nama "FormBaru" dengan isi berikut :

```
17  public class FormUtama extends javax.swing.JFrame {
18  private int jml;
19  private String judul;
20
21      /**
22       * Creates new form FormUtama
23       */
24      public FormUtama() {
25      setTitle("Mdi Form");
26      setExtendedState(MAXIMIZED_BOTH);
27          initComponents();
28      }
29
30      private void FormBaru(){
31       try{
32         JInternalFrame jin = new JInternalFrame(judul,false, true,true);
33          jin.setDefaultCloseOperation(DISPOSE_ON_CLOSE);
34          JPanel pp = new JPanel();
35          jin.setBounds(10, 10, 500, 500);
36          jDesktopPane1.add(jin);
37          jin.setVisible(true);
38       }catch (Exception e){
39          JOptionPane.showMessageDialog(null, e);
40       }
41      }
```

Isi Perintah pada Menu New (JMenuItem) dengan cara Klik kanan pada menuNew

- ✓ Events
- ✓ Action
- ✓ actionPerformed.



Gambar 1.1 Compiler

Ketikkan kode program seperti berikut :

```
165     private void mitnewActionPerformed(java.awt.event.ActionEvent
166         // TODO add your handling code here:
167         jml = jml + 1;
168         judul = "Form Baru "+jml;
169         FormBaru();
```

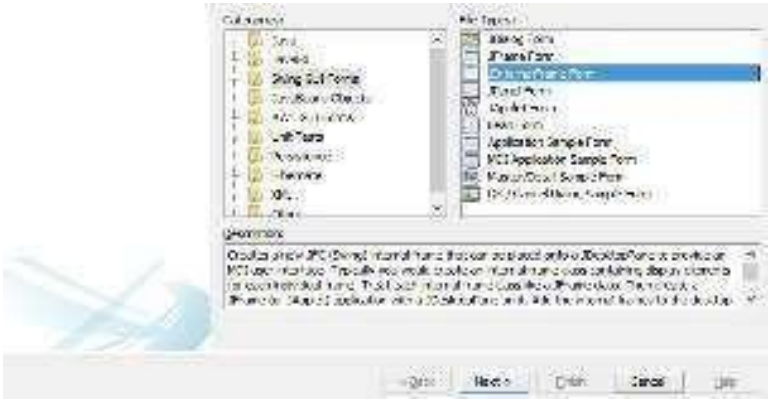
Isi Perintah pada Menu Keluar (JMenuItem) dengan cara Klik kanan pada menu New

- ✓ Events
- ✓ Action
- ✓ actionPerformed.

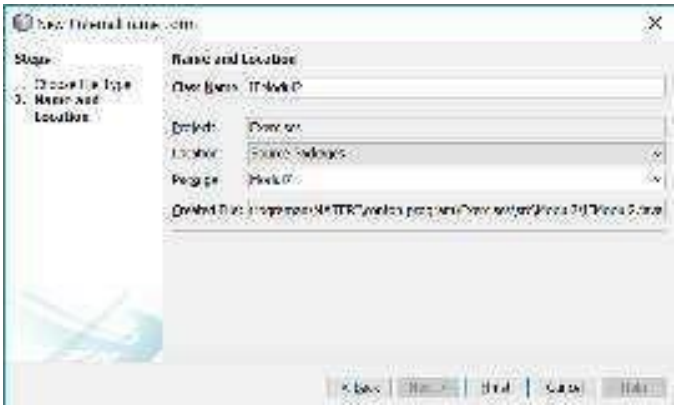
```
194     private void mitkeluarActionPerformed(java.awt.event.ActionEvent
195         // TODO add your handling code here:
196         System.exit(0);
197     }
```

Tambahkan JInternalFrame Form pada project pilih menu File

New File pilih Project anda, kemudian pada Categories pilih Swing GUI Forms dan pada File Types pilih JInternalFrame Form kemudian klik Next.



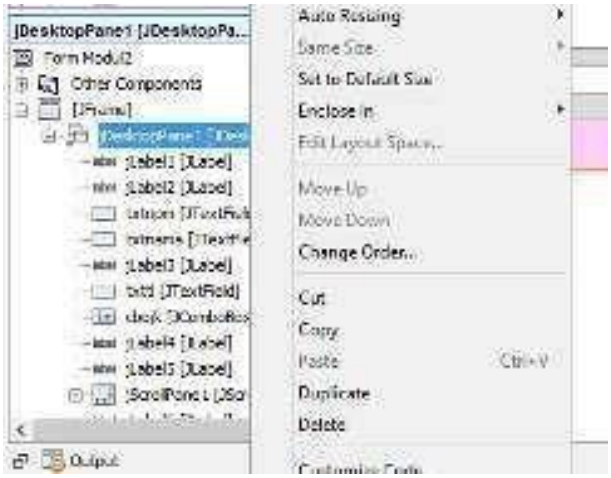
Isikan Class name “IFModul2” dan Pakage “Modul7” kemudian klik Finish.



Kemudian desain sama seperti program pada modul2, atau copy paste dengancara :

Buka Modul2 yang sebelumnya kemudian pilih navigator item jDesktopPane1

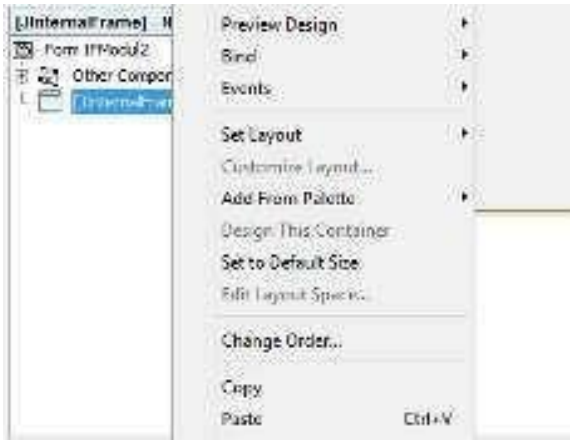
klik kanan



Copy.

Buka IFModul2 yang sebelumnya kemudian pilih navigator item jInternalFrame

- ✓ klik kanan
- ✓ Paste.



Lakukan hal yang sama (ikuti langkah-langkah 11 s/d 12) untuk membuat file jInternalFrame Form untuk 'IFModul3', 'IFModul4', 'IFModul5'.

Gambar 1.1 Compiler

Ubah nilai properti pada masing-masing InternalFrame, pilih navigator item jInternalFrame

- ✓ klik kanan Properties. Centang Closable sehingga bernilai true.

Kembali pada FormUtama kemudian berikan perintah-perintah untuk memanggil form-form yang telah dibuat diatas.

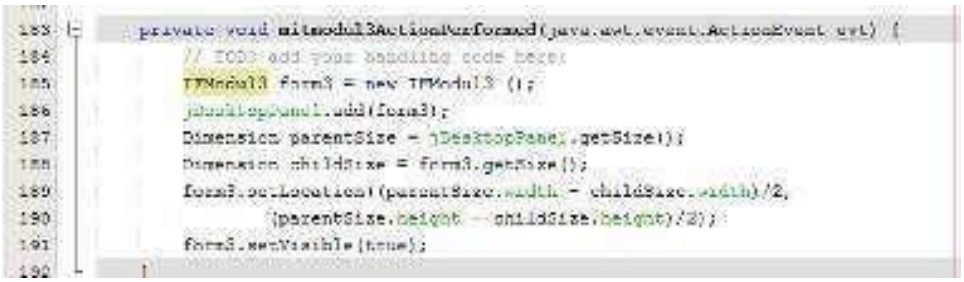
Isi Perintah pada Menu Modul2 (JMenuItem) dengan cara Klik kanan pada menuModul 2

- ✓ Events
- ✓ Action

actionPerformed.

```
171
172     private void mimodul2ActionPerformed(java.awt.event.ActionEvent evt) {
173         // TODO add your handling code here:
174         IFModul2 form2 = new IFModul2();
175         jDesktopPane1.add(form2);
176         Dimension parentSize = jDesktopPane1.getSize();
177         Dimension childSize = form2.getSize();
178         form2.setLocation((parentSize.width - childSize.width)/2,
179                 (parentSize.height - childSize.height)/2);
180         form2.setVisible(true);
181     }
```

Isi Perintah pada Menu Modul3 (JMenuItem) dengan cara Klik kanan pada menuModul 3 ⇨ Events ⇨ Action ⇨ actionPerformed.



```
    private void mitmodul3ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        IFModul3 form3 = new IFModul3 ();
        jDesktopPanel.add(form3);
        Dimension parentSize = jDesktopPanel.getSize();
        Dimension childSize = form3.getSize();
        form3.setLocation((parentSize.width - childSize.width)/2,
                (parentSize.height - childSize.height)/2);
        form3.setVisible(true);
    }
```

Isi Perintah pada Menu Modul4 (JMenuItem) dengan cara Klik kanan pada menuModul 4

- ✓ Events
- ✓ Action
- ✓ actionPerformed.

```
    private void mitmodul4ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        IFModul4 form4 = new IFModul4 ();
        jDesktopPanel.add(form4);
        Dimension parentSize = jDesktopPanel.getSize();
        Dimension childSize = form4.getSize();
        form4.setLocation((parentSize.width - childSize.width)/2,
                (parentSize.height - childSize.height)/2);
        form4.setVisible(true);
    }
```

Isi Perintah pada Menu Modul5 (JMenuItem) dengan cara Klik kanan pada menuModul 5 ⇨ Events ⇨ Action ⇨ actionPerformed.

```
    private void mitmodul5ActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        IFModul5 form5 = new IFModul5 ();
        jDesktopPanel.add(form5);
        Dimension parentSize = jDesktopPanel.getSize();
        Dimension childSize = form5.getSize();
        form5.setLocation((parentSize.width - childSize.width)/2,
                (parentSize.height - childSize.height)/2);
        form5.setVisible(true);
    }
```

Run Program melalui FormUtama menu Run Run File (Shift + F6).

Gambar 1.1 Compiler

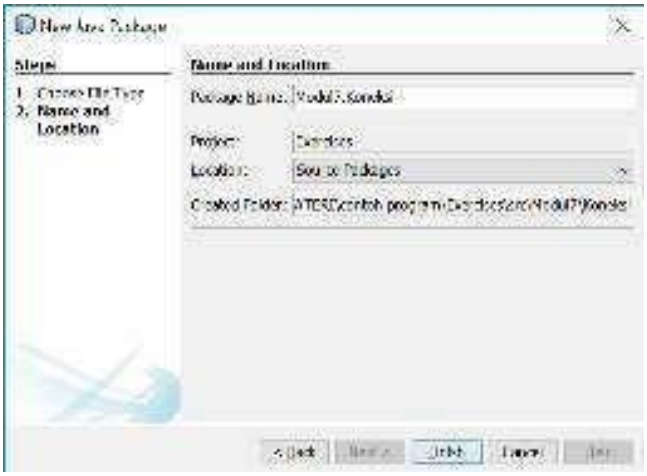
b. Aplikasi Input-Data1

Berikut ini adalah cara untuk membuat program input data customer dari Javadan Netbeans dan menggunakan database MySQL

Buatlah database dan tabel terlebih dahulu :

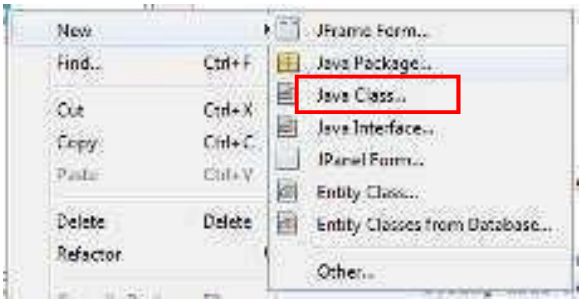
- ✓ CREATE DATABASE **dbjava**;
- ✓ USE **dbjava**;
- ✓ CREATE TABLE **tcustomer** (id VARCHAR(6) PRIMARY KEY,nama VARCHAR(25),
- ✓ nohp VARCHAR(14), alamat VARCHAR(100));

Tambahkan paket (*Java Package*) pada projek 'Exercises'. Beri nama paketdengan 'Modul7.Koneksi'.

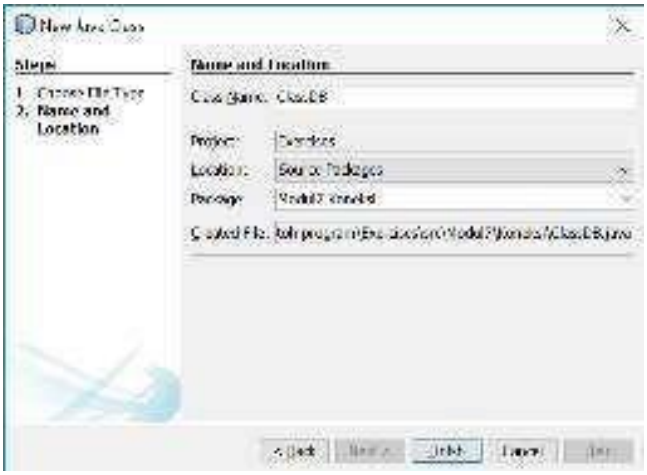


Gambar 1.1 Compiler

Buatlah JClass caranya, klik kanan pakage Modul7.Koneksi, New Java Class.Beri nama 'ClassDB', kemudian klik Finish.



Gambar 1.1 Compiler



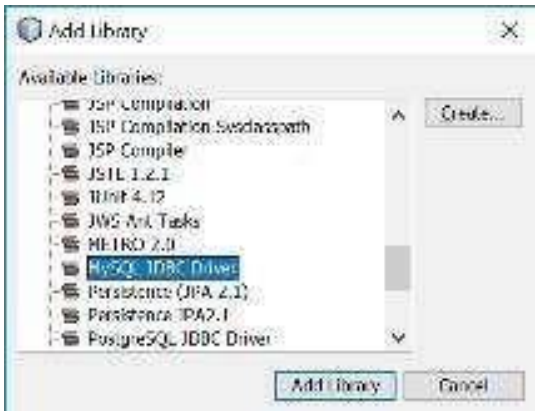
Gambar 1.1 Compiler

Tambahkan Library konektor MySQL pada Libraries Project 'Exercises' dengan

- ✓ cara klik kanan pada folder Libraries

- ✓ Add Library

Pilih MySQL JDBC Driver ⇾ klik tombol 'Add Library'

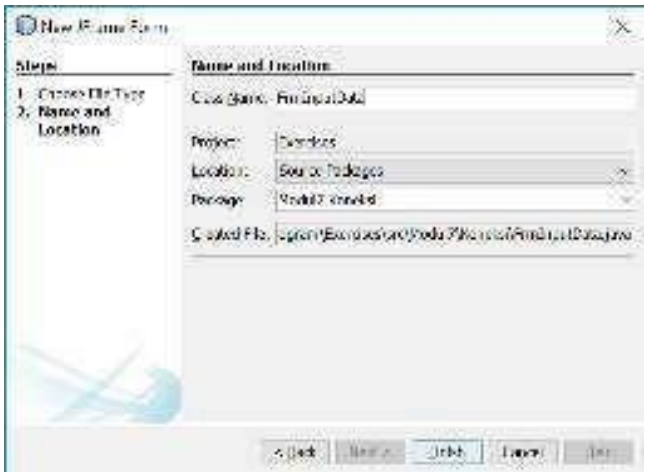


Gambar 1.1 Compiler

Ketikkan kode progam berikut ini didalam ‘ClassDB’.

```
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;



public class ClassDB {
    private static Connection koneksi;
    public static Connection getKoneksi() {
        if (koneksi==null) {
            try {
                String url=new String();
                String user=new String();
                String password=new String();
                url="jdbc:mysql://localhost:3306/dbjaya";
                user="root";
                password="1234";
                DriverManager.registerDriver(new com.mysql.jdbc.Driver());
                koneksi=DriverManager.getConnection(url,user,password);
            }catch (SQLException x) {
                System.out.println("Error membuat koneksi");
            }
        }
      return koneksi;
    }
}
```

Tambahkan JFrame Form pada paket ‘Modul7.Koneksi’ dan beri nama ‘FrmInputData’.



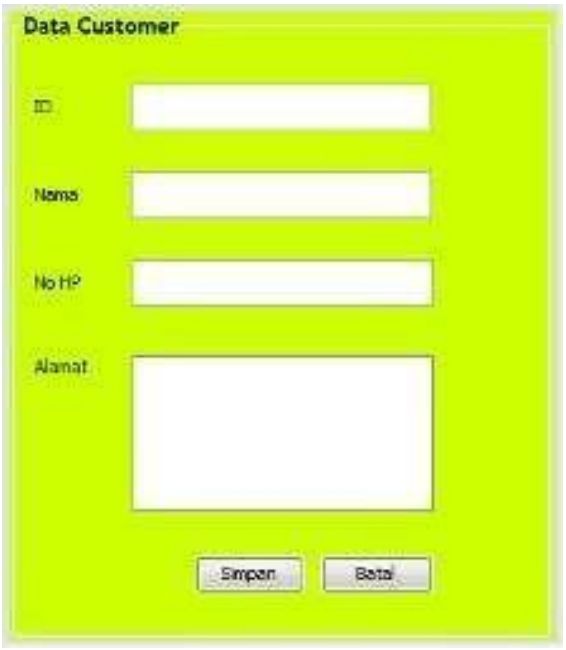
Gambar 1.1 Compiler

Tambahkan beberapa komponen seperti yang ada di dalam tabel berikut ini:

Objek	**Properties**	**Nilai**
jDesktopPane1	Background	Purple [204,0,255]
	Border	Title Border = Data Customer

JLabel1	text	ID
JLabel2	text	Nama
JLabel3	text	No Hp
JLabel4	text	Alamat
JTextField1	text Variable Name	Txtid
JTextField2	text Variable Name	Txtnama
JTextField3	text Variable Name	Txthp
JTextArea	text Variable Name	Txtalamat
JButton1	text Variable Name	Simpan Btnsimpan
JButton2	text Variable Name	Batal Btnbatal

Susunan posisi dan ukuran komponen seperti pada gambar dibawah ini:



Gambar 1.1 Compiler

Deklarasikan class import berikut :

```
6   package Modul7.Koneksi;
7
8   import java.sql.*;
9   import javax.swing.JOptionPane;
10  import Modul7.Koneksi.ClassDB;
11
```

Selanjutnya Ketikkan kode berikut ini:

```
37      public FrmInputData() {
38          initComponents();
39          bacaId();
40          txtid.setEnabled(false);
41          btnsave.setEnabled(false);
42      }
43
44      private void bacaId() {
45          int kode = 0;
46          try {
47              Class.forName("com.mysql.jdbc.Driver");
48              try (Connection cn = DriverManager.getConnection("jdbc:mysql://localhost/dbjava", "root", "");
49                      Statement smt = cn.createStatement()) {
50                  String sql = "SELECT max(Id) as id FROM tcustomer";
51                  ResultSet rs = smt.executeQuery(sql);
52                  if (rs.next()) {
53                      kode = rs.getInt("id");
54                  } else {
55                  }
56                  txtid.setText("C00" + (kode + 1));
57              }
58          } catch (ClassNotFoundException | SQLException e) {
59          }
60      }
```

Keterangan Kode:

Method dengan nama ‘bacaId’, digunakan untuk membaca field id pada tabeltcustomer.

```
private void simpan() {
    String id = this.txtid.getText();
    String nama = this.txtnama.getText();
    String nohp = this.txthp.getText();
    String alamat = this.txtalamat.getText();
    String st = "Free";
```

```
if ("".equals(this.txtid.getText()) || ("".equals(this.txtnama.getText()))
        ||                    ("".equals(this.txthp.getText()))          ||
("".equals(this.txtalamat.getText()))) {
    JOptionPane.showMessageDialog(this, "Lengkapi
    data");
} else {
    try {
        Connection c =
        ClassDB.getkoneksi();Statement s
        = c.createStatement();
        String sql = "Insert into tcustomer values
        (?,?,?,?)";try
        (com.mysql.jdbc.PreparedStatement
            p=  (com.mysql.jdbc.PreparedStatement)
            c.prepareStatement(sql))
    {
        p.setString(1, id);
        p.setString(2, nama);
        p.setString(3, nohp);
        p.setString(4, alamat);
        p.executeUpdate();
```

Keterangan Kode:

Method dengan nama simpan, digunakan untuk menyimpan data pada tabel tcustomer.

Selanjutnya di bagian bawah kode perintah:

```
private void bersih() {
    bacaId();
    txtnama.setText(null);
    txthp.setText(null);
    txtalamat.setText(null);
    btnsimpan.setEnabled(fals
    e);txtid.setEnabled(false);
```

Keterangan kode:

Method dengan nama bersih, maksudnya yaitu untuk membersihkan/ mengosongkannilai pada tiap-tiap JTextField dengan tujuan untuk memulai proses baru.

Klik kanan pada btnsimpan setelah itu pilih Events

- ✓ Action

actionPerformed. Ketikkan kode perintah:

simpan ():

Klik kanan pada btnclear setelah itu pilih Events

- ✓ Action
- ✓ actionPerformed.Ketikkan kode perintah:

Bersih () ;

Klik kanan pada txtnama setelah itu pilih Events

- ✓ Key
- ✓ KeyTyped. Ketikkankode perintah:

Btnsimpan.setEnabled(true);

Kompile dan jalankan dari menu Run

- ✓ Run File atau tekan Shift + F6.

Id akan otomatis terisi. Isikan Nama, No HP, Alamat kemudian pilih tombolSimpan.

Klik Tombol Clear untuk mengosongkan TextField.

DAFTAR PUSTAKA

Sakur, S. B.2011. Pemrograman Berorientasi Objek-Konsep & Implementasi. Yogyakarta: Andi.

Mohan, P., Fundamentals of Object Oriented Programming in Java, he University of the West lndie

Khan nedy, E. K., 2011. Belajar Java Dasar. Bandung. Available

Rahardjo, B., Heryanto, I., Haryono, I. 2012. Mudah Belajar Java: Revisi Kedua. Bandung: Penerbit Informatika.

Koentjoro, E.Y ., 201 6, Modul Praktikum Pemrograman Berorientasi Objek, STIKOM Surabaya

David Etheridge, "Java : Graphical User Interfaces" Bookbon.com

Justin Simmons, "Java Programming " Global Media

James Levenick "Simply Java An Introduction toJava Programming" Willamette University

Aloysius Sigit W, "7Proyek Aplikas Dengan Java," Elexmedia Komputindo

Y. Daniel Liang, "Java Programming Comprehensive Version 10th", Pearson

R.H. Sianipar, "Java : Teori, Algoritma, dan Aplikasi", Andi Publisher

Kishori Sharan, "Beginning Java 9 Fundamentals: Arrays, Objects, Modules,

JShell, and Regular Expressions 2nd Edition", Apress Publisher

Adam Drozdek, "Data Structure and Algorithms in C++ 4th edition", Cengage Learning

BELAJAR DENGAN MUDAH
PEMROGRAMAN
BERORIENTASI OBJEK
BAGI PEMULA

Oleh
Muhamad Sidik, S.Kom., M.Kom

BIODATA PENULIS

BIODATA PENULIS Penulis Memiliki Berbagai Disiplin Ilmu Yang Diperoleh Dari Universitas Kristen Satya Wacana (UKSW) Disiplin Ilmu Itu Antara Lain Pemrograman Mobile, Pemrograman Berorientasi Objek. Penulis Memiliki Pengalaman Kerja Di Dunia Akademisi Dari Seorang Guru Hingga Menjadi Seorang Dosen.Penulis Adalah Seorang Dosen Pada Universitas Sains Dan Teknologi Computer (STEKOM) Dan Seorang Yang Memiliki Jabatan Fungsional Akademik Asistes Ahli Dan Menulis Beberapajurnal Terakhreditas Nasional, Beberapa Karya Cipta IOT Hasil Penelitian Yang Di Danai Oleh LLDIKTI. Penulis Juga Terlibat Dalam Organisasi

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
JL. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-623-8120-03-1 (PDF)
9 786238 120031