MODUL PEMROGRAMAN BERORIENTASI OBJEK (PBO)

AKBAR JULIANSYAH, ST.,M.MT

SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER
(STMIK) BUMIGORA MATARAM
2017

DAFTAR ISI

MODUL 1 KONSEP DASAR JAVA

[Pert. 1]

TUGAS PENDAHULUAN

Buatlah algoritma dari program yang ada dalam kegiatan praktikum ini !

1. TUJUAN

a). Mahasiswa mengetahui dasar dan elemen-elemen pembentuk bahasa Java

b). Mahasiswa dapat menggunakan dasar dan elemen-elemen pembentuk bahasa Java

2. DASAR TEORI

Java adalah bahasa pemrograman seperti halnya dengan bahasa pemrograman lain seperti Pascal, Basic, dan C. Java juga dapat digunakan untuk membuat aplikasi web yang saat ini terus berkembang. Java dikembangkan oleh Sun Microsystem pada Agustus 1991 dengan nama semula Oak dan merupakan hasil perpaduan dari berbagai bahasa pemrograman seperti C, C++, Objective-C, Smalltalk dan CLISP. Kelebihan dari Java adalah program java dapat berjalan pada berbagai platform (multi platform).

Kompilasi code Java menggunakan JVM (Java Virtual Machine) yang nantinya JVM tersebut akan merubah intermediate code menjadi bytecode (machine code). Pada Java 2, interpreter Java terkemas dalam sebuah paket yang disebut JRE (Java Runtime Environment).

Java Technology

Java salah satu bahasa pemrograman baru menjanjikan banyak kemudahan bagi programer junior maupun senior (selama menggunakan IDE handal bayaknya NetBeans). Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh Sun Microsystems. Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan platform independent, aitu dapat dijalankan di berbagai jenis sistem operasi dan arsitektur komputer. Bahasa ini juga dirancang untuk pemrograman di Internet sehingga dirancang agar aman dan portabel.

Java telah digunakan dalam banyak hal dan telah membuktikan keberadaannya pada abad ke 21. Saat ini, Java digunakan pada bermacam jenis aplikasi seperti aplikasi mobile phone (J2ME), aplikasi embedded, aplikasi keuangan, desktop, simulasi pesawat,

pemrosesan citra, game, aplikasi perusahaan terdistribusi yang disebut J2EE dan masih banyak lagi.

Keyword

Berikut adalah kata-kata kunci yang ada pada Java. Kata kunci tidak dapat digunakan sebagai pengenal (identifier) ataupun sebagai nilai variable dalam pemrograman Java.

Abstract	else	interface	super
Boolean	extends	long	switch
break	final	native	strictfp
byte	finally	new	synchroniz
case	float	null	this
catch	for	package	throw
char	goto	private	throws
class	if	protected	transient
const	implement	public	try
continue	Import	return	void
do	instanceof	short	volatile
double	Int	static	While

Comment

Comment merupakan bagian dari source code yang tidak dijalankan tetapi hanya digunakan untuk dokumentasi. Sama seperti pada bahasa C, comment pada bahasa Java dimulai dengan "//" diakhiri dengan enter atau dimulai dengan "/*" dan diakhiri oleh "*/".

Variable

Variabel digunakan untuk menampung nilaipada pemrograman. Kita tidak dapat Menyimpan tipe bernilai angka ke variabel bertipe nilai huruf ataupun sebaliknya, sehingga tipe dan nilai variable harus sesuai. Variabel merupakan container yang digunakan untuk menyimpan suatu nilai pada sebuah program dengan tipe tertentu. Untuk mendefinisikan variabel, sama seperti pada bahasa pemrograman keluarga C lainnya, kita dapat menuliskan tipe data dan identifier untuk menamai variabel tersebut.

Data Type

Pada Java memiliki 8 jenis tipe data primitif, antara lain

- Tipe bilangan bulat : long, int. short, byte
- Tipe bilangan riil : float, double
- Tipe data karakter : char
- Tipe data Boolean : boolean

Untuk range atau rentang nilai tipe data numerik sebagai berikut:

Nama	Lebar (bit)	Rentang Nilai
Long	64	-9223372036854775808 s/d 9223372036854775807
Double	64	1.7E-308 s/d 1.7E+308
Int	32	-2147483648 s/d 2147483647
Float	32	3.4E-038 s/d 3.4E+038
Short	16	-32768 s/d 32767
Byte	8	-128 s/d 127

- **Operator**

a. Operator Aritmatika

Operator	Hasil
+	Penjumlahan
-	Pengurangan
*	Perkalian
/	Pembagian

%	Modulus
++	Increment
--	Decrement
+=	Persamaan penjumlahan
-=	Persamaan pengurangan

b. Operator Logika

Operator	Hasil
&&	AND
||	OR
!	NOT

c. Operator Relasi

Operator	Hasil
==	Sama dengan
!=	Tidak sama dengan
>	Lebih besar dari
<	Lebih kecil dari
>=	Lebih besar dari atau sama dengan
<=	Lebih kecil dari atau sama dengan

Percabangan

- IF-ELSE

```
if (kondisi_1) {
  // instruksi jika kondisi_1 true
  if (kondisi_2) ;// instruksi
  jika kondisi_2 true else ;
  // instruksi jika kondisi_2
  false
} else {
  // instruksi jika kondisi_1 false
  if (kondisi_3) ; // instruksi jika kondisi_3 true
  else if(kondisi_4) ;// instruksi jika
kondisi_3 false dan kondisi_4 true
}
```

- SWITCH-CASE

```
switch(ekspresi) {case nilai_1:
  // instruksi bila ekspresi
  bernilai nilai_1 break;
  case nilai_2:
  // instruksi bila ekspresi
bernilai nilai_2 case
nilai_3:
  // instruksi bila ekspresi bernilai
  nilai_2 atau nilai_3 break;
default:
  // instruksi bila ekspresi tidak selain nilai di atas
   }
```

Perulangan

- FOR

```
for ( /* 1 */ ; /* 2 */ ; /* 3 */ )
{
  /* 4 */
  if( /* 5 */ ) continu
  e ; if( /* 6 */ ) break;
  /* 7 */
}
/* 8 */
```

Bagian 1 akan dijalankan 1x, lalu dilanjutkan bagian 2, apabila bagian 2 bernilai salah maka akan dilanjutkan bagian 8, apabila benar akan dilanjutkan pada semua bagian di dalam kurung, bagian 3 dan kembali lagi pada bagian 2. Apabila bagian 5 bernilai true, maka program akan berlanjut ke bagian 3. Apabila bagian bagian 6 bernilai true, maka looping akan berhenti dan dilanjutkan ke bagian 8.

- WHILE

```
while (kondisi) {
  // blok_pernyataan;
}
```

- DO-WHILE

```
do{
  // pernyataan_pernyataan;
} while(kondisi);
```

Array

- Array 1 Dimensi

```
int[] angka=new int[5];

int[] angka={5,3,23,99,2};
```

- Array 2 Dimensi

```
tipeData[][] nama_variabel = new tipeData[jumlah_baris][jumlah_kolom];
```

- Array Dinamis

Untuk dinamis kita dapat menggunakan class Vector atau class StringTokenizer yang terdapat pada class Java.util. Penggunaan keduanya dapat kita lihat pada link http://java.sun.com/j2se/1.4.2/docs/api/java/util/Vector.html, http://java.sun.com/javase/7/docs/api/java/util/StringTokenizer.html

Contoh potongan program, mengisi variabel StringTokenizer dari string dan menampilkan isinya.

```
String kata = "Belajar Pemrograman Java"; String
Tokenizer st = new StringTokenizer(kata); while(st.
hasMoreTokens())
{
  // System.out.println(st.nextToken()+ " ");
}
```

Contoh hasil outputnya:

B e l a jar Pe mrog raman J a va

Untuk lebih memahami penggunaan array 2 dimensi, berikut kita akan mencoba membuat program yang menghasilkan nilai acak dari 0-9 dan ditampung pada array dua dimensi, kemudian ditampilkan pada layar.

```
public class LatArray {
  public static void main(String[] args) {
  int [][] angka = new int[5][5];
  System.out.println("==Data Array 2 dimensi==");
  for (int i=0; i < angka.length; i++) {
      for(int j=0; j<angka[i].length; j++) {
        angka[i][j]= (int) (Math.random() * 10);
      }
    }
    for (int baris=0; baris<angka.length; baris++) {
      for(int   kolom=0;kolom<angka[baris].length;kolom++)
      {
       System.out.print(angka[baris][kolom] + " ");
      }
      System.out.println();
    }
  }
}
```

3. KEGIATAN PRAKTIKUM

- Input dari Keyboard

Pada setiap aplikasi rata-rata telah menggunakan inputan dari keyboard, misalnya jika kita ingin membuat menu pilihan di mana user dapat memilih pada menu mana dia akan masuk. Oleh karena itu, kita harus menggunakan sebuah class yang khusus bertugas menangani pembacaan inputan keyboard.

Berikut adalah contoh penulisan program pada java secara lengkap dan cara menangkap inputan keyboard:

```
import java.io
.*; public class
Main {
public static void main(String[ ]args) throws IOException{
 BufferedReader br= new BufferedReader(new InputStreamR
eader(Sytem.in)); int input=0;
System.out.print(" ==Menu Program= = \n"
+ "1. Aplikasi Kotak \n"
+ "2. Aplikasi Segitiga \n"
+"3 Exit \n"          ); System.out.pr
int(" Masukkan pilihan: "); int pilih
an= Integer.parseInt(br.readLin
e()); switch(pilihan) { case 1:
System.out.println(" \n= = Aplikasi
Kotak= = "); System.out.print(" Mas
ukkan angka:"); input= Integer.pars
eInt(br.readLine()); cetak(1, input);
break; case 2:
System.out.println(" \n= = Aplikasi S
egitiga= = "); System.out.print(" Mas
ukkan angka:"); input= Integer.parseI
nt(br.readLine()); cetak(2, input); bre
ak; case 3:
System.out.println(" \nKeluar
Aplikasi..."); System.exit(0); bre
ak;
```

```
}
}
public static void cetak (int jenis, int inputan) { S
ystem.out.println("");
                if(jenis==1){for(int baris=0; baris<inp
                  utan; baris++){for(int    kolom=0;kol
                  om<inputan;kolom++){S
                     ystem.out.print("*");
                    }System.out.print
                    ln("*");
                  }
                }else if(jenis==2){
                  for(int baris=0; baris<inputan; b
                  aris++){for(intkolom=0;kolom<b
                  aris;kolom++){System.out.print
                  ("*");
                    }System.out.println("*");
                  }
                } else { System.out.println("Unknown");
                }
              }
            }
```

Contoh hasil output:

```
==Menu Program==

1. AplikasiKotak

2. AplikasiSegitiga

3. Exit

Masukkan pilihan:1


==AplikasIKotak==
Masukkan angka:4

* * * * * * * * * * * * * * * * * * * *
```

4. TUGAS

1. Buat program kalkulator.java sederhana, tetapi menggunakan menu.

Contoh:

==========

Kalkulator Sederhana

==========

Menu:

1. Penambahan (+)

2. Pengurangan (-)

3. Perkalian (*)

4. Pembagian (/)

5. Modulus (%)

6. Exit

Masukkan pilihan anda : 1

Masukkan Bilangan1 : 5

Masukkan Bilangan2 : 4

Hasil Penjumlahan : 9

(Bila ditekan Enter akan kembali ke menu awal)

MODUL 2 CLASS, OBJECT, METHOD DAN CONSTRUCTOR

[Pert. 2]

TUGAS PENDAHULUAN

Jelaskan tentang Kelas, Objek, Konstruktor! Dan berilah contohnya!

1. TUJUAN

Praktikan mampu mengetahui dan menggunakan fitur-fitur object-oriented dasar bahasa Java.

2. DASAR TEORI

OOP (Object Oriented Programming)

Pemrograman Berorientasi Object atau sering kita kenal dengan OOP merupakan metode pemrograman yang masih popular pada saat ini. Dalam OOP, object menjadi konsentrasi utama dalam pemrograman dibandingkan dengan teknik pemrograman sebelumnya yaitu pemrograman prosedural di mana fungsi-fungsinya menjadi konsentrasi utama. Pada pemrograman dengan prosedural seperti ini sering kita jumpai di mana satu file atau form memiliki jumlah baris source code sangat banyak karena semua dituliskan pada file atau form tersebut, selain itu sering dijumpai redundansi atau penulisan fungsi yang sama pada form yang lain.

Pada OOP memungkinkan kita untuk membuat beberapa object yang terdiri dari data dan fungsi kemudian dipanggil pada object yang lainnya. Dengan pendekatan ini, maka penulisan code menjadi lebih ringkas dengan mengurangi duplikasi fungsifungsinya.

Fungsi cukup dituliskan satu kali kemudian dapat digunakan pada object-object yang lain. Pengembangan program ke depan juga akan menjadi lebih mudah. Pengembangan dapat dilakukan pada object yang diperlukan saja tanpa banyak mempengaruhi code pada object yang lain.

Object

Elemen dasar dari konsep OOP adalah object. Object merupakan abstraksi sesuatu dalam dunia nyata, misalnya: manusia, binatang, aktivitas, bussiness system atau workflow. Sebuah object dalam OOP terdiri dari 2 elemen penyusun:

a) Attribut (state, data member), merupakan ciri-ciri yang melekat pada suatu object.

b) Method (behaviour, member function) merupakan fungsi-fungsi yang digunakan untuk memanipulasi nilai-nilai pada attribut atau untuk melakukan hal-hal yang dapat dilakukan pada suatu object.

Object sepeda motor:

Atribut:

- Merk
- Warna
- Ukuran sepeda
- Nomer kerangka
- Jumlah perseneling

Method:

- Nyalakan mesin
- Berjalan
- Belok
- Mengerem
- Menambah kecepatan

Object mahasiswa:

Attribut:

- NPM
- Nama mahasiswa
- Fakultas
- Jurusan

- Nilai ujian

Method:

- Terima materi
- Melihat nilai
- Melihat tunggakan spp
- serah tugas

Class

Setelah kita memahami tentang object yang masih memiliki bentuk yang abstrak, maka kita akan melanjutkan ke pembahasan class. Dikatakan abstrak karena masih belum spesifik untuk siapakah object tersebut ditujukan, misalnya object sepeda motor ditujukan untuk jenis sepeda motor apa (honda beat, Yamaha yupiter, kawasaki ninja). Ketiga jenis motor tersebut akan menjadi class untuk object sepeda motor.

Contoh berikutnya misalnya object mahasiswa yang ditujukan untuk seorang mahasiswa. Sebagai contoh mahasiswa dengan NPM 08120100015, maka dapat dibentuk class mahasiswa dengan NPM 08120100015 yang merupakan spesifikasi atau implementasi dari object mahasiswa.

Sebuah object yang sudah dibentuk atau digambarkan perlu didefinisikan menjadi sebuah class. ***Class merupakan cetak biru (blue print) dari object***. Berikut contoh sintaks dari class mahasiswa dan class Main-nya. Class main digunakan untuk melakukan instanisasi class Mahasiswa, sehingga attribut dan method pada class Mahasiswa dapat digunakan.

```
package modul2a;
import java.io.*;
class Mahasiswa{
 private String npm, nama, fakultas, jurusan;
 private int nTugas, nUjian;
 public void setNPM(String npm) {
  this.npm = npm;
 }
 public String getNPM() {
  return npm;
 }
```

```
}
public class Main {
 public static void main(String[] args) throws IOException {
  BufferedReader br = new BufferedReader(new
InputStreamReader(System.in));
  Mahasiswa mhs = new Mahasiswa();
  System.out.print("Masukkan NPM: ");
  mhs.setNPM(br.readLine());
  System.out.println(mhs.getNPM());
 }
}
```

Pada source code kita melihat baris berikut:

Mahasiswa mhs = new Mahasiswa();

adalah sintaks untuk melakukan instanisasi class sebelum dapat digunakan (diisi dan ditampilkan nilai- nilainya).

Method

Secara umum method terdiri dari 2 jenis:

a. Tidak mengembalikan nilai (prosedur), dengan melihat pada contoh source code class Mahasiswa, maka method yang tidak mengembalikan nilai adalah setNPM(String npm). Memiliki ciri dengan tipe data void.

b. Mengembalikan nilai (fungsi), contoh method yang mengembalikan nilai pada class Mahasiswa adalah getNPM().

Constructor

Constructor adalah method yang dapat digunakan untuk memberikan nilai awal saat object diciptakan. Constructor otomatis dipanggil/dijalankan pada saat di-instance pada sebuah class.

Sifat constructor:

a. Memiliki nama yang sama dengan nama class.
b. Tidak memiliki return value/nilai balik.
c. Merupakan method yang pertama kali dipanggil saat class tersebut dibuat.

Berikut contoh sebuah 2 constructor dalam sebuah class:

```
class Mahasiswa{
  private String npm, nama, fakultas, jurusan;
  private int nTugas, nUjian;
    public Mahasiswa() {
   this.npm = "";
   this.nama= "";
  }
  public Mahasiswa(String npm, String nama) {
    this.npm = npm;
    this.nama = nama;
  }
  @Override void finalize() { }
}
```

Pada contoh di atas terdapat 2 constructor, constructor yang pertama merupakan ***constructor tanpa parameter***, sedangkan constructor yang kedua adalah constructor dengan 2 parameter yaitu npm dan nama. Constructor yang kedua dinamakan ***overloading constructor.***

Overloading

Overloading adalah pembuatan metode-metode dengan nama yang sama tetapi jumlah parameter dari metode-metode tersebut berbeda.

3. KEGIATAN PRAKTIKUM

✓ **Pembuatan Kelas (Class)**

Kelas pada java didefinisikan dengan menggunakan kata kunci class.

Contoh sederhana penciptaan kelas:

a. Pembuatan Kelas “Halo” pada bahasa pemrograman java

Bahasa Algoritmik	Bahasa Java
class Halo	class Halo{
private kata:string procedure main()	public static void main (String[] args) { String kata;
kata ← “halo sahabat” output (kata) {end procedure} {end class}	kata=”halo sahabat”; System.out.println(kata); } }

Bagaimana keluaran program tersebut (no.1) ?

b. Contoh penciptaan kelas

```
class Siswa{
       //variable instan
    String nama;
      //metode
    void isiData(String namaku) {
             nama=namaku;
        }
     String ambilNama(){
         return nama;
        }
     }
```

Bagaimana keluaran dari program tersebut (no.2) ?

✓ **Konstruktor (Constructor)**

Kita sudah mengetahui bahwa kelas adalah alat untuk menciptakan objek. Sekarang yang menjadi pertanyaan adalah bagaimana cara menciptakan objek menggunakan sebuah kelas.

Jawabannya adalah dengan menggunakan sebuah konstruktor.

a. Contoh :

```
class MOTOR{
  String merk;
  int tahun;
  String noPolisi;
  String warna;

  public MOTOR(String merk,int tahun, String noPolisi,String
warna){
    this.merk=merk;
    this.tahun=tahun;
    this.noPolisi=noPolisi;
    this.warna=warna;
   }
 void showInfoMotor(){
   System.out.println("Merk : "+this.merk);
   System.out.println("Tahun: "+this.tahun);
   System.out.println("No Polisi: "+this.noPolisi);
   System.out.println("Warna : " + this.warna);
 }
}
public class KelasMotorku{
  public static void main(String [] args){
  MOTOR motorku = new MOTOR("Honda GL Pro",1997,"G 5879
BF","Hitam");
   motorku.showInfoMotor();
  }
}
```

Bagaimana keluaran dari program konstruktor diatas (no.1) ?

Potongan program manakah yang dimaksud dengan kelas dan objek/konstruktor ?

2. Contoh :

```
class tampilNilai{
 private String nilai;
 private String kategori;
public tampilNilai(char huruf){
switch (huruf){
  case 'A':
this.nilai="antara 80 sampai 100";
  this.kategori="Istimewa";
   break;
  case 'B':
  this.nilai="antara 65 sampai 79";
  this.kategori="Baik";
   break;
  case 'C':
  this.nilai="antara 56 sampai 64";
  this.kategori="Cukup";
  break;
  case 'D':
  this.nilai="antara 40 sampai 55";
  this.kategori="Kurang";
  break;
  case 'E':
  this.nilai="antara 0 sampai 39";
  this.kategori="Buruk";
  break;
  }
```

```
}
 public void info(){
   System.out.println("Nilaiku : "+this.nilai);
   System.out.println("Kategori: "+this.kategori);
 }
}
 class nilaiku {
  public static void main(String[]args){
 tampilNilai obj=new tampilNilai('A');
  obj.info();
 }
}
```

Bagaimana keluaran darai program konstruktor diatas (no.2)?. Modifikasilah program diatas (no.2) dimana terdapat input nilai dari luar! Potongan program manakah yang dimaksud dengan kelas dan objek/konstruktor?

✓ **Overloading**

```
Public class Hitung {
static int tambah(int x, int y){
return x+y;
}
static double tambah(double x, double y){
return x+y;
}
static int tambah(int x, int y, int z){
return x+y+z;
}
static void tambah2(int x, int y){
System.out.println("x"+x+" + y"+y+"="+(x+y));
}
```

```
        public static void main(String[] args){
        int x,y;
        x=2;
        y=x+3;
        x=tambah(2,3);
        System.out.println("1.    "+x);
        System.out.printf("2.   %.2f \n",tambah(2.3, 4.1));
        System.out.println("3.   "+tambah(5, 12, 3));
        System.out.println("4.   "+tambah(100, 0.5));
        System.out.println("5.   "+tambah(2.5, 4));
        System.out.println("6.    "+tambah(tambah(x*2,   (x*2-y)),
        tambah((y-x),
        tambah(7, 3), 2)));
        System.exit(0);
        }
}
```

4. TUGAS

1. Buatlah program untuk menghitung luas, keliling dari persegi panjang dengan memanfaatkan konsep OOP!
2. Tanyakan pada asisten praktikum!

MODUL 3 INHERITANCE

[Pert. 3]

Tugas Pendahuluan

1. Sebutkan syarat pemrograman yang mendukung pemrograman berorientasi objek!
2. Apakah yang dimaksud inheritance pada pemrograman berbasis objek?

1. TUJUAN

Mahasiswa mampu membuat Method Inheritance dengan hirarki penciptaan kelas untuk pembentukan kelas super dan sub kelas serta Overiding Method.

2. DASAR TEORI

❖ Inheritance

Inheritance adalah kemampuan suatu objek atau kelas untuk mewariskan sifat-sifat yang terdapat didalamnya ke kelas turunannya. Inheritance merupakan suatu mekanisme yang memungkinkan seorang pemrogram menciptakan kelas baru berdasarkan kelas yang sudah tersedia sehingga tidak perlu menuliskan kode dari nol. Kelas dasar/kelas induk mewarisi semua metode dan variable instant, namun kelas turunan dapat menambahkan metode baru atau variable instant baru tersendiri.

Class yang mewariskan disebut dengan superclass/parent class/baseclass, sedangkan yang mewarisi disebut subclass/child class/derived class. Keuntungan dari inheritance adalah mengurangi penggunaan code yang berulang (reuse), karena sub class dapat menggunakan attribute dan method yang ada pada superclass-nya.

a. Cara Pewarisan Kelas.

Untuk menerapkan inheritance pada java menggunakan statement extends. Kelas turunan secara prinsip dapat dibuat dengan menggunakan bentuk :

```
Class KelasTurunan extends KelasDasar {
        Tubuh kelas
}
```

b. Pemanggilan Konstruktor Super Kelas

Pada contoh sebelumnya, Superkelas tidak mengandung konstruktor. Bagaimana jika superkelas memiliki konstruktor. Bagaimana apabila subkelas ingin memanggil konstruktor. Dalam hal ini bisa menggunakan kata kunci super.

```
Super (nama , nim );
```

Pemanggilan konstruktor kelas dasar harus memenuhi persyaratan berikut :

- Pemanggilan dengan super seperti diatas hanya bisa dilakukan pada konstruktor.
- Pemanggilan konstruktor superkelas harus berkedudukan sebagai pernyataan pertama dalam konstruktor

❖ **Keyword "super" and "this"**

Keyword **super** digunakan oleh subclass untuk memanggil constructor dan method dari superclass-nya. Contoh sintaks untuk memanggil constructor dari superclass-nya:

```
super()
super(parameter)
```

Contoh sintaks untuk memanggil method dari superclass-nya:

```
super.namaMethod(parameter1, parameter2, ....)
```

Keyword **this** digunakan untuk membedakan antara variable pada level class dengan variabel pada method. Keyword this menandakan bahwa varibel tersebut merupakan variabel pada level class. Untuk memperjelas, kita lihat contoh berikut.

```
public class BangunDatar {
private String jenis;
private int panjang1;
private int panjang2;
public BangunDatar(String jenis, int panjang1, int panjang2) {
  this.jenis = jenis;
  this.panjang1 = panjang1;
  this.panjang2 = panjang2;
 }
}
```

3. KEGIATAN PRAKTIKUM

a. Pada class Salam

```
class Salam {
       String slm=" Hello !!!";
public void info1() {
System.out.println(slm);
     }
}
class PanggilSalam extends Salam{
    String salamku="selamat pagi";
  public void info2(){
    System.out.println(salamku);
   }
public static void main(String[] args){
PanggilSalam obj=new PanggilSalam();
        obj.info1();
        obj.info2();
  }
}
```

Bagaimana hasil keluarannya? Berilah penjelasan tentang program inheritance no.1 tersebut!

b. Pada class Dasar

```
class DASAR{
 private int x;
 public int GetX(){
  this.x=20;
  return this.x;
  }
}
//membuat kelas turunan
 class TURUNAN extends   A   {
   int y, hasil;
  DASAR A =    B    ();
  int X = A.GetX();
  public void Sety (int yy){
    this.y=yy;
    }
  public void Kalixy (){
    this.hasil = X * this.y;
   }
 public int getHasil(){
    return this.hasil;
  }
}
class Pewarisan {
 public static void main(String[] args){
 TURUNAN B=    C    ();
 B.Sety(5);
 B.Kalixy();
 System.out.println("Hasil x kali y : " + B.getHasil());
 }
}
```

Lengkapi program inheritance no.2 pada kolom merah A, B, C! Bagaimana hasil keluarannya?

c. Pada class Identitasku

```
class Identitasku{
private String universitas="Universitas Trunojoyo Madura";
 private String jurusan="Pendidikan Informatika";
 private String nama;
 private String nim;
//Konstruktor
public Identitasku(String nama,String nim){
  this.nama=nama;
  this.nim=nim;
}
//Metode
public void info(){
  System.out.println("Universitas :"+universitas);
  System.out.println("Jurusan     :"+jurusan);
  System.out.println("Nama        :"+this.nama);
  System.out.println("Nim         :"+this.nim);
  }
}
class Keterangan extends Identitasku{
 protected String angkatan;
 protected String alamat;
//konstruktor
public  Keterangan(String nama,String nim, String angkatan,String alamat){
 super(nama,nim);
 this.angkatan=angkatan;
         this.alamat=alamat;
 }
//Metode
public void info(){
    super.info();
 System.out.println("Angkatan    :"+this.angkatan);
 System.out.println("Alamat      :"+this.alamat); }
}
class KonstruktorSuperKelas{
  public static void main(String[]args){
    Keterangan mahasiswa=new Keterangan("Sholahuddin Ayyubi","0412585","2014","Surabaya");
    mahasiswa.info(); }
}
```

Bagaimana keluaran program inherintance no.3 diatas? Jelaskan potongan dari program :

1. class Keterangan extends Identitasku →baris ke-19
2. super (nama,nim) →baris ke 24
3. super.info() →baris ke-30
4. Keterangan mahasiswa=new Keterangan →baris ke-36

d. Konstructor dan inheritance pada class BangunDatar.

```
public class BangunDatar {
  private String jenis;
  private int panjang1;
  private int panjang2;
     public BangunDatar() {
    this.panjang1=0;
    this.panjang2=0;
  }
     public BangunDatar(String jenis, int panjang1, int panjang2) {
    this.jenis = jenis;
    this.panjang1 = panjang1;
this.panjang2 = panjang2;
  }
}
class PersegiPanjang extends BangunDatar{
  protected int panjang, lebar, luas;
  protected String jenis;
  public PersegiPanjang(String jenis, int panjang, int lebar) {
    super(jenis, panjang, lebar);
    this.jenis=jenis;
    this.panjang=panjang;
    this.lebar=lebar;
   }
   public int Luas() {
     this.luas = panjang * lebar;
     return luas;
   }
}
class Main{
  public static void main(String[] args) {
    PersegiPanjang pp = new PersegiPanjang("Square", 10, 8);
    System.out.println("Panjang : " + pp.panjang);
    System.out.println("Lebar   : " + pp.lebar);
    System.out.println("Luas    : " + pp.Luas());
  }
}
```

Pada contoh no.4 diatas terdapat 3 class. Class BangunDatar merupakan superClass, class PersegiPanjang merupakan subclass, sedangkan class Main digunakan untuk mencetak hasil dari subclass PersegiPanjang. Bagaimana hasil keluaran program no.4 tersebut? Modifikasilah program no.4 dengan memberi masukan dari luar (bukan didefinisikan dalam program)!

4. TUGAS

Buatlah sebuah program yang menerapkan constructor dan inheritance yang kita pelajari pada modul ini, dengan ketentuan sebagai berikut:

- ✓ super class dengan nama BangunRuang yang memiliki attribute panjang1, panjang2, panjang3.
- ✓ sub class Kubus yang mempunyai attribut sisi, dan method Luas Permukaan dan Volume.
- ✓ Rumus luas permukaan kubus = 6 x sisi x sisi
- ✓ Rumus volume kubus = sisi x sisi x sisi

- ✓ sub class balok yang mempunyai attribut panjang, lebar, tinggi, dan method LuasPermukaan dan Volume.
- ✓ Rumus luas permukaan balok = 2 x ((panjang x lebar) +(panjang x tinggi) + (lebar x tinggi))
- ✓ Rumus volume balok = panjang x lebar x tinggi
- ✓ sub class Tabung yang memiliki attribut jari-jari, tinggi, dan method LuasPermukaan dan Volume. (p = 3,14)
- ✓ Rumus luas permukaan tabung = (2 x p x jari-jari x tinggi) + (2 x p x jari-jari x jarijari)

Contoh keluaran

```
======================
Perhitungan Bangun Ruang
======================
Menu:
1. Kubus
2. Balok
3. Tabung
4. Exit
Masukkan pilihan anda : 2

Masukkan panjang (cm) : 10
Masukkan lebar (cm)   : 5
Masukkan lebar (cm)   : 3
Luas Permukaan (cm2)  : 190
Volume (cm3)          : 150
(Bila ditekan Enter akan kembali ke menu awal )
```

MODUL 4
ENCAPSULATION
[Pert. 4]

Tugas Pendahuluan

1. Jelaskan konsep dari sifat encapsulation pada pemrograman berorientasi objek!
2. Jelaskan atribut yang digunakan untuk proses enkapsulasi java!

1. TUJUAN

Mahasiswa mampu memahami konsep encapsulation dan aplikasinya dalam sebuah program java pada pemrograman bebrbasis objek

2. DASAR TEORI

Encapsulation

Inti dari konsep encapsulation adalah menyembunyikan kompleksitas suatu class dari class yang lain yang mengaksesnya. Dengan menggunakan ***public, protected, private dan package*** dapat digunakan untuk memilih data mana saja yang ditampilkan dan disembunyikan.

Encapsulation (Enkapsulasi) adalah suatu cara untuk menyembunyikan implementasi detail dari suatu class. Enkapsulasi mempunyai dua hal mendasar, yaitu:

- Information hiding, menyediakan suatu perantara (method) untuk pengaksesan data
- Enkapsulasi dapat dianggap sebagai bungkusan. Ekapsulasi inilah yang diimplemtasikan dalam sebuah kelas dimana dalam sebuah kelas terdiri dari atribut dan metode yang dibungkus dalam sebuah kelas. Ekapsulasi dalam sebuah kelas bertujuan untuk melindungi atribut dan metode yang ada didalam kelas gar tidak sembarangan diakses oleh kelas lain.

```
import java.io.*;
class Manusia {
```

```
  private String nama, jns_kel;
  private double tinggi, berat;
  public Manusia(String nama, String jns_kel, double tinggi, double
berat) {
    this.nama=nama;
    this.jns_kel=jns_kel;
    this.tinggi=tinggi;
    this.berat=berat;
  }
}
class Mahasiswa extends Manusia {
private String npm, asalSMU, fakultas, jurusan;
private int nTugas, nUjian;
public Mahasiswa(String nama, String jns_kel, double tinggi, double
berat, String npm, String asalSMU,
String fakultas, String jurusan,
int nTugas, int nUjian) {
     super(nama, jns_kel, tinggi, berat);
  }
}
```

Bila kita perhatikan pada atribut class Manusia yaitu nama, jenis kelamin, tinggi dan berat memiliki access modifier private, sehingga attribut tersebut tidak dapat diakses atau digunakan pada class lain kecuali class itu sendiri yaitu class Manusia. Encapsulation sangat berkaitan dengan access modifier.

Penentu Akses/Access Modifier

Berkaitan dengan boleh-tidaknya suatu variable instant diakses dari luar kelas, java menyediakan penentu akses, yaitu :

a. ***Public*** berarti bahwa pengaksesan suatu variable instant atau metode dapat dilakukan dari luar kelas
b. ***Private*** berarti bahwa pengaksesan suatu variable instant atau metode hanya dapat dilakukan di dalam kelas; tidak bias diakses dari luar kelas.
c. ***Protected*** berati bahwa pengaksesan suatu variable instant atau metode yang terdapat pada sebuah kelas dapat diakses pada kelas itu sendiri dan pada sub kelas

3. KEGIATAN PRAKTIKUM

a. Analisalah program dibawah ini:

```
import java.io.*;
class Manusia {
  private String nama, jns_kel;
  private double tinggi, berat;
  public Manusia(String nama, String jns_kel, double tinggi, double berat) {
    this.nama=nama;
    this.jns_kel=jns_kel;
    this.tinggi=tinggi;
    this.berat=berat;
  }
} class Mahasiswa extends Manusia { private
String npm, asalSMU, fakultas, jurusan;
private int nTugas, nUjian;
public Mahasiswa(String nama, String jns_kel, double tinggi, double berat,
String npm, String asalSMU, String
fakultas, String jurusan, int nTugas,
int nUjian) { super(nama, jns_kel,
tinggi, berat); }
}
```

b. Apa perbedaan antara access modifier public, protected, private ?

c. Apa yang terjadi jika anda membuat sebuah property atau method menjadi private, protected, public ?

4. TUGAS

Buatlah program untuk menghitung gaji bersih dari seorang pegawai, pajak ppn sebesar 10% dari gaji kotor

MODUL 5
POLIMORFISME
[Pert. 5]

Tugas Pendahuluan

Jelaskan konsep dari sifat polimorfisme pada pemrograman berorientasi objek!

1. TUJUAN

Mampu membuat method Polimorfisme untuk implementasi listing code yang berbeda sesuai dengan behavior masing- masing kelas.

2. DASAR TEORI

Polimorfisme adalah kemampuan mengungkap suatu hal yang berbeda melalui satu cara yang sama. Apabila mempunyai objek yang bertipe superkelas, variable objek ini bisa diisi dengan objek superkelas ataupun objek subkelas tanpa perlu melakukan perubahan tipe.

Polymorphism berarti sebuah object dapat memiliki berbagai bentuk, yaitu dengan memiliki nama yang sama, tetapi behavior yang berbeda. Sebuah object dapat memiliki berbagai bentuk baik dari class itu sendiri ataupun dari superclass-nya. Ada 2 macam bentuk polymorphism, yaitu:

a. **Overloading:** penggunaan satu nama untuk beberapa method yang berbeda dalam suatu class. Nama method yang sama dibedakan dari jumlah parameter dan

tipe data parameternya.

Overloading mempunyai ciri-ciri sebagai berikut:

- Nama method harus sama
- Daftar parameter harus berbeda
- Return type boleh sama, juga boleh berbeda

Contoh penggunaan overloading dilihat di bawah ini:

Gambar(int t1) →1parametertitik,untuk menggambar titik

Gambar(int t1, int t2) →2parametertitik,untuk menggambar garis

Gambar(int t1, int t2,int t3) →2parametertitik,untuk menggambar segitiga
Gambar(int t1, int t2,int t3, int t4) →2parametertitik,untuk menggambar segiempat

b. **Overriding**: mendeklarasikan sebuah method dengan nama dan parameter yang sama dari superclass-nya, kemudian dimodifikasi ulang (menumpuk/mendefinisi ulang).

Pada polymorphism, dikenal juga dynamic binding atau ikatan dinamis yang umumnya digunakan pada variabel bertipe class. Jika kita mempunyai variabel bertipe superclass, variabel ini dapat diisi dengan object superclass ataupun subclass-nya tanpa melakukan perubahan tipe. Untuk lebih jelas mengenai dynamic binding kita akan melihat contoh berikut:

```
public class Manusia { private
String nama; public void
setNama(String nama) {
   this.nama=nama;
 } public String
 getNama() {
  return "Nama manusia: " + nama;
 }
}
class Mahasiswa extends Manusia{
 private String namaMhs; public
 void setNama(String nama) {
  this.namaMhs = nama;
 }
 public String getNama() {
  return "Nama mahasiswa: " + namaMhs;
 }
}
class Karyawan extends Manusia{
private String namaKar; public void setNama(String nama) { }
 public String getNama() {
  return "Nama Karyawan : " + namaKar; } } class Main{
```

```
    public static void main(String[] args) {
      Mahasiswa mhs = new Mahasiswa();
      Karyawan kar = new Karyawan(); Manusia mns; mns =
      mhs;
      mns.setNama("Aisyah                    azzahra");
      System.out.println(mns.getNama());    mns    =    kar;
      mns.setNama("maryam asyifa"));
      System.out.println(mns.getNama());
    }
    }
  this.namaKar = nama;
```

Pada contoh class Manusia merupakan superclass, sedangkan class Mahasiswa dan class Karyawan merupakan subclass dari class Manusia. Ketiga class tersebut memiliki method setNama() dan getNama(). Kesamaan nama dan tipe data method pada subclass yang mengacu superclass dinamakan overriding method. Kemudian variabel object Manusia (mns) menunjuk ke object yang dirujuk oleh Mahasiswa (mhs), pada baris mns = mhs; variabel object Manusia (mns) ke object yang dirujuk oleh object Karyawan (kar), pada baris mns = kar;

3. KEGIATAN PRAKTIKUM

1. Cobahlah contoh program diatas (class Manusia)
2. Contoh program polimerfisme

```
class Binatang{
   public void info() {
     System.out.println("Info tentang Hewan : ");
 }
}
class Herbivora extends Binatang {
  public void info() {
    System.out.println("Info  pada  herbivora:  Memakan  makanan berupa tumbuh-tumbuhan");
  }
}
class Kelinci extends Herbivora{
  public void info(){
    System.out.println("Info pada Kelinci: Memakan makanan berupa wortel");
 }
}
class Polimorfisme {
  public static void main(String[] args){
        Herbivora herbivora;
        Kelinci kelinciku;
        Binatang hewan;

  ......................
   }
}
```

- Lengkapi program no.1 diatas!
- Bagaimana hasil keluarannya?

3. Berikut contoh penggunaan class dengan menerapkan constructor, inheritance, access modifier (encapsulation) dan polymorphism. Cobalah code di bawah ini sebagai latihan dan compile-running untuk melihat hasilnya.

```
import java.io.*; class
Manusia{
  private String nama, jns_kel;
  private double tinggi, berat;
  public Manusia() {
  this.nama="";
  this.jns_kel=""; this.tinggi=0;
      this.berat=0;
    } public Manusia(String nama, String jns_kel, double
  tinggi,
double berat) {
    this.nama=nama;
    this.jns_kel=jns_kel;
    this.tinggi=tinggi;
```

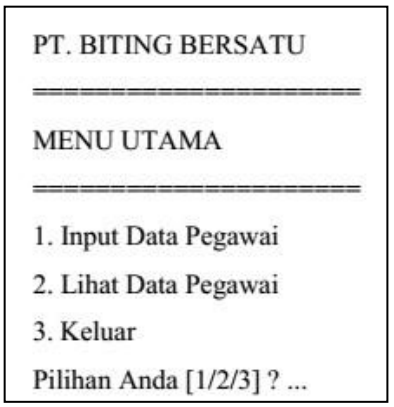
```
    this.berat=berat;
  }
  public String cetak() { return("Nama: " + nama + "\nJenis
Kelamin: " + jns_kel + "\nTinggi: " + tinggi + "\nBerat: " +
berat); }
} class Mahasiswa extends
Manusia {
  private String npm, asalSMU, fakultas, jurusan; public
  Mahasiswa(String nama, String jns_kel, double tinggi,
  double berat, String npm, String asalSMU,
          String fakultas, String jurusan)
    { super(nama, jns_kel, tinggi, berat);
    this.npm = npm; this.asalSMU =
    asalSMU; this.fakultas = fakultas;
    this.jurusan = jurusan;
  } public String
  cetak() {
    return(super.cetak()+"\n\nNPM: " + npm + "\nAsal SMU: " +
asalSMU + "\nFakultas: " + fakultas + "\nJurusan: " + jurusan +
"\nNilai Tugas: " + nTugas + "\nNilai Ujian: " + nUjian);
  } }
public class Main {
  public static void main(String[] args) throws IOException { Mahasiswa
    mhs = new Mahasiswa("Andi", "L", 170, 60,
"08120080035", "SMAN 1 " , "Ilmu Komputer" , "Pendidikan
Informatika", 80, 90);
    System.out.println(mhs.cetak());
  }
}
```

4. TUGAS

PT. Biting Bersatu memiliki 50 orang pegawai, pegawainya tersebut terbagi menjadi 2 status kepegawaian, yaitu sebagian pegawai tetap dan sebagian yang lain (sisanya) adalah pegawai kontrak. Secara umum perbedaan antara keduanya adalah pegawai tetap selain mendapatkan gaji juga mendapatkan tunjangan yang besarnya 500 ribu, sedangkan pegawai kontrak hanya mendapatkan gaji saja. Dari kasus diatas, dapat digambarkan class-class pegawai sebagai berikut:

Class PegawaiKontrak, dengan atribut:
NIK : no pegawai kontrak (diinputkan)
Nama : nama pegawai (diinputkan)
MasaKontrak : lama kontrak pegawai (diinputkan)
Kehadiran : jumlah hari masuk dalam 1 bulan (diinputkan)
UangMakan : besarnya 8 ribu dikali jumlah kehadiran (tidak diinputkan)
Gaji : besarnya gaji pokok yang diterima tiap bulan (diinputkan)
Class PegawaiTetap, dengan atribut: NIK : no pegawai tetap (diinputkan)
Nama : nama pegawai (diinputkan)
Kehadiran : jumlah hari masuk dalam 1 bulan (diinputkan)
Tunjangan : besarnya 500 ribu (tidak diinputkan)
UangMakan : besarnya 8 ribu dikali jumlah kehadiran (tidak diinputkan)
Gaji : besarnya gaji pokok yang diterima tiap bulan (diinputkan)
Method-method yang harus ada:
hitungGaji(): untuk menghitung gaji bersih pegawai, dimana untuk
pegawai kontrak = Uang Makan + Gaji,
pegawai tetap = Tunjangan + Uang Makan + Gaji
lihatData(): untuk menampilkan data-data pegawai secara lengkap beserta
total gaji yang diterima (gaji bersih)

Dari kedua class diatas, analisa dan desainlah superclass yang tepat untuk kasus tersebut. Setelah itu buatlah class utama/main class yaitu class PegawaiBitingBersatu (yang menggunakan classclass diatas) yang memiliki menu utama sebagai berikut:



Tentukan pula modifier yang tepat untuk semua class diatas. Tteknik polymorphism bisa digunakan dalam menyelesaikan permasalahan ini.

MODUL 6
ABSTRACTION
[Pert. 6]

Tugas Pendahuluan

1. Apa fungsi abstract pada pemrograman java?
2. Apakah superclass dan subclass?

1. TUJUAN

Mahasiswa mampu membuat kelas Abstrak untuk Method Abstraksi

2. DASAR TEORI

Abstract class digunakan sebagai model atau acuan (superclass) bagi penurunan untuk sub-sub class-nya. Pada umumnya abstract class hanya berisi variable-variabel umum dan abstract method tanpa body, kemudian baru pada class turunannya mendefinisikan secara detil body methodnya. Untuk membuat abstract class dan abstract method menggunakan keyword abstract. Berikut contoh untuk mendefinisikan abstract class, abstract method, non abstract method.

```
public abstract class Manusia{
  protected String nama, jns_kel;
  public Manusia() { }
  public abstract String setNama(String nama);
  public abstract void getNama();
  public String cetak() {
    return this.nama + "/n" + this.jns_kel;
  }
}
```

Abstraksi adalah proses pengabstrakan atau penyembunyian detail program yang sangat rumit sehingga kita tidak perlu untuk mempermasalahkan pembuatannya. Kita hanya perlu obyek tersebut dapat kita gunakan sesuai fungsinya. Dalam java suatu metode ditentukan dari dalam kelas tetapi tidak disertai definisinya, metode ini dikenal dengan metode abstrak, sedangkan kelasnya disebut kelas abstrak. Definisi kelas diletakkan pada masing-masing kelas turunannya.

Kelas abstrak biasanya dibuat apabila pada subkelas-subkelas memerlukan operasi yang

sama dengan metode tersebut, akan tetapi antara subkelas dengan subkelas lain memiliki tindakan yang berbeda.Untuk mendeklarasikan kelas abstrak dan metodenya, dipergunakan kata kunci abstract.

The "Final" Keyword

Pada Java terdapat keyword "final" yang digunakan untuk mencegah suatu class diturunkan atau suatu method di-override. Keyword final dituliskan pada class atau method setelah access modifier. Sebagai contoh perhatikan penggunaan keyword final berikut:

```
public final class TestFinal {
private String username, password;
public TestFinal() {
   this.username = "labkom";
            this.password = "LABKOM-123";
          }
           public final boolean isValid(String n, String p) {
            if(n.equals(username) && p.equals(password)) {
              return true;
            }
            else return false;
           }
          }
```

Sebagai catatan, class atau method yang diberi attribut atau keyword final tidak boleh berupa abstract class atau abstract method. Karena abstract class harus diturunkan, sedangkan abstract method harus di-override pada subclass

Interface

Interface berisi sekumpulan konstanta dan atau deklarasi method tanpa menyertakan body methodnya yang dapat digunakan pada sejumlah class. Sekilas Interface mirip dengan Abstract Class karena sama-sama menentukan bentuk method untuk subclass-nya. Untuk mengetahui perbedaan keduanya, kita amati table perbedaan Interface dengan Abstract Class berikut:

Abstract Class

- Dapat berisi abstract method dan non abstract method.
- Dapat berisi variabel instant dan konstanta .
- Menggunakan extends .

- Pada 1 class pewarisan hanya dapat mewarisi 1 abstract class.
- Abstract method harus di-override pada class turunan, sedagkan yang bukan abstract method tidak harus dituliskan pada class turunannya.

Interface

- Hanya berisi abstract method saja.
- Hanya dapat berisi konstanta saja .
- Menggunakan implements .
- Pada 1 class pewarisan dapat mewarisi lebih dari 1 interface
- Semua method harus diturunkan pada class implement-nya

Berikut contoh interface :

```
public interface Manusia {
public String setNama(String nama);
public void getNama();
}
```

Package

Sejumlah class yang terkait pada Java biasa dikelompokkan dalam sebuah unit yang disebut package (paket). Pada NetBeans kita dapat mendeklarasikan JavaPackage dalam 1 project, atau dengan kata lain 1 project dapat memiliki 2 atau lebih package. Untuk membuat atau mendeklarasikan package dengan menggunakan keyword package. Berikut contoh membuat class Mahasiswa08 dan class Mahasiswa09 dalam package modul2c.

```
package modul2c;
public class Mahasiswa08 {
  public void getData() {
    System.out.println("Mahasiswa SI 2008");
  }
}
package modul2c;
public class Mahasiswa09 {
  public void getData() {
    System.out.println("Mahasiswa SI 2009");
  }
}
```

Untuk melakukan tes terhadap package yang sudah kita buat, buatlah sebuah main class yang terletak di luar package modul2c seperti pada gambar berikut. Untuk memanggil class dalam package, gunakanlah keyword import. Jika tidak, maka class tidak bisa di-instance-kan karena tidak dikenali.

3. KEGIATAN PRAKTIKUM

a. Contoh program abstraction

```
public abstract class Pelajar{
  protected String nama;
  public abstract void belajar();
 }
// Berkas : TesAbstrak1.java
class siswa1 extends Pelajar{
  public siswa1( String nama){
    this.nama=nama;
 }
public void belajar(){
 System.out.println( this.nama+ " memperhatikan gurunya yang mengajar didepan kelas");
 }
}
class siswa2 extends Pelajar{
  public siswa2 ( String nama){
     this.nama=nama;
 }
public void belajar(){
 System.out.println(this.nama+ " memperhatikan gurunya yang mengajar di depan kelas");
 }
}
class TesAbstrak1{
  public static void main ( String[] args) {
siswa1 mhs=new siswa1("Azzahra");
       mhs.belajar();
       siswa2 mrd=new siswa2("Aisyah");
       mrd.belajar();
}
```

b. Program abstract

```
abstract class Titik{
/*kelas yang digunakan untuk mengimpelementasikan sebuah tipe titik*/
 private int x; /*koordinat x*/
 private int y; /*koordinat y*/
 Titik(){
  /*konstruktor*/
  x = 0;
  y = 0;
 }
 Titik(int xp, int yp){
  /*konstruktor*/
  x = xp;
  y = yp;
 }
 public void setX(int xp){
  /*mengeset nilai koordinat x*/
  x = xp;
 }
 public int getX(){
 /*mengembalikan nilai koordinat x*/
```

```
    return x;
  }
  public void setY(int yp){
    /*mengeset nilai koordinat y*/
    y = yp;
  }
  public int getY(){
  /*mengembalikan nilai koordinat y*/
    return y;
  }
  public abstract void printTitik();
  public void finalize(){
    /*destruktor*/
  }
}
class Titik3D extends Titik{
  /*kelas turunan kelas Titik*/
  private int z; /*koordinat z*/
  Titik3D(){
    /*konstruktor*/
    z = 0;
  }
  Titik3D(int xp, int yp, int zp){
    /*konstruktor*/
    setX(xp);
    setY(yp);
    z = zp;
  }
  public void setZ(int zp){
    /*mengeset nilai koordinat z*/
    z = zp;
  }
  public int getZ(){
  /*mengembalikan nilai koordinat z*/
    return z;
  }
  public void printTitik(){
    /*menampilkan nilai koordinat titik*/
    System.out.println("nilai X : " + getX());
    System.out.println("nilai Y : " + getY());
    System.out.println("nilai Z : " + getZ());
  }
}
class CobaTitik3DP{
/*metode main untuk mengetes kelas Titik dan kelas Titik3D*/
  public static void main(String[] args) {
    Titik3D t = new Titik3D(0, 0, 7);
    t.setX(28);
    t.setY(1);
    t.printTitik();
    Titik t1;
    System.out.println("====================");
    t1 = t;
    t1.printTitik();
  }
}
```

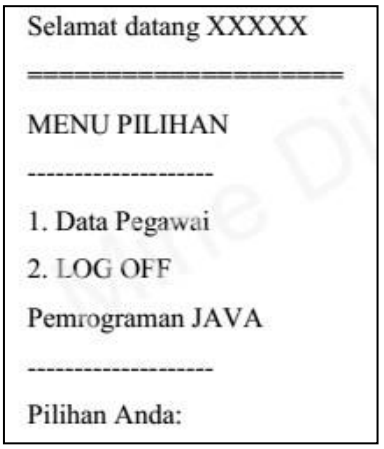
4. TUGAS

1. Buatlah sebuah abstract class Login dimana pada class tersebut terdapat 2 method, yaitu validasi() dan cekData(). Buatlah sebuah class yang mengimplementasikan interface tersebut yaitu class DataLogin dengan penjelasan sebagai berikut:
   - Method validasi(): bertipe boolean, digunakan untuk memastikan bahwa username dan password tidak boleh kosong (wajib diisi), tentukan sendiri parameternya.
   - Method cekData(): bertipe boolean, digunakan untuk mengecek username dan password apakah cocok dengan yang terdapat di atribut class DataLogin.
   - (private String username1="mhs", private String pass1="Mahasiswa@#$", private String username2="mahasiswa", private String pass2="praktikum@#$"), tentukan sendiri parameternya.

   Buatlah class TestLogin yang menggunakan class DataLogin dengan aturan sebagai berikut, saat program dijalankan, tampilkan menu utama sebagaiberikut:

```
MENU UTAMA
----------
Menu Pilihan:
A. LOGIN
B. EXIT
----------
Pilihan Anda:
```

   User dapat memilih pilihan dengan huruf besar maupun huruf kecil. Jika LOGIN dipilih, maka user diminta menginputkan nama user dan password. Jika salah, maka tampilkan pesan bahwa user salah menginputkan username atau password atau keduanya dan program kembali ke menu utama. Jika benar, maka tampilkan menu pilihan sebagai berikut:



XXXXX = nama user yang sedang login. Jika user memilih pilihan 1, maka jalankan class utama. Jika user memilih pilihan 2, maka tampilan kembali ke menu utama (tidak keluar dari program). Program akan selesai jika user memilih EXIT.

2. Dengan soal nomor 1, gunakan konsep interface, yaitu interface Login. Untuk melakukan testing program, buatlah sebuah class main dengan menggunakan konsep package.

3. Buatlah sebuah outer class Email yang memiliki inner class level 1 dengan nama class BodyContent dan class Recipient. Variabel dan method pada tiap class dapat kalian tentukan sendiri. Setelah itu buatlah class main yang menginstancekan outer class dan inner class tersebut, buatlah contoh implementasi sesuai kreatif kalian.

MODUL 7 JAVA GUI (GRAPHICAL USER INTERFACE)

[Pert. 7]

1. TUJUAN

Mampu menerapkan dan membuat program dengan interface secara graphis (GUI) memanfaatkan method AWT dan SWING dalam pembuatan komponen.

2. DASAR TEORI

Pada bab-bab sebelumnya interaksi antara user dengan program hanya berbasis console editor dengan tampilan dos yang membosankan, maka agar interaksi antara user dengan program tidak membosankan diperlukanlah sebuah interface yang menghubungkan antara user dengan program dengan tampilan grafis, interface ini dinamakan dengan GUI (Graphical User Interface).

Dalam pemrograman GUI terdapat beberapa bagian yang harus dilakukan yaitu:

a. Membuat windows utama
b. Menentukan komponen-komponen pendukung program
c. Menentukan tata letak layout agar nantinya semua komponen- komponen yang sudah dipersiapkan bisa diaatur sedemikian rupa
d. Event Handling dari sebuah aktivitas, seperti penekanan button, check box dan lain-lain

Java menyediakan 2 kelas untuk bekerja dengan GUI, yaitu AWT dan Swing

1. AWT(abstract windows tollkit): sebuah kelas yang terdapat dalam package java.awt, dimana berisi komponen-komponen GUI yang bersifat platform oriented atau tegantung pada sistem operasi

   Beberapa fasilitas yang disediakan oleh java.awt adalah:

   - Pengaturan tata letak (layout management) komponen dalam sebuah container
   - Mendukung event handling, yaitu mekanisme pendeteksian event dan penentuan respons yang yang akan diberikan ketika pengguna akan mengakses komponen tersebut
   - Menipulasi grafis dari komponen, seperti font,warna,icon dan sebagainya Penamaan kelas – kelas pada AWT adalah :Button,Panel dll…
2. Swing : API (Aplication Program Interface) yang disediakan oleh java untuk pemrograman GUI, Swing merupakan bagian dari JFC (Java Foundation Class) terdapat pada package javax.swing dan bersifat lightweight, yaitu dapat di

aplikasikan untuk semua platform (multipaltform), sebelum Swing fitur GUI oleh API java disebut AWT , untuk java versi 1.4 keatas kita memakai Swing tapi kita masih bisa menggunkan AWT bila benar-benar digunakan.Penamaan kelas-kelas pada Swing memakai huruf depan J. Contohnya Jbutton, Jpanel,Jframe dll

Kedua package diatas memiliki event handling yang sama sehingga kita bisa mempelajari eventhandling keduanya secara bersamaan, sebelum melakukan eksplorasi lebih lanjut dengan GUI java, perlu dipahami beberapa elemen berkenaan dengan pemrograman GUI. Container: adalah sebuah wadah untuk meletakkan komponen- komponen GUI, contohnya seperti canvas seorang pelukis, dalam hal ini lukisan yang ditempelkan pada container adalah komponen- komponen GUI seperti Button, textedit, radiobutton, dan lainlain, container GUI pada java bisa dibuat berlapis , artinya sebuah GUI dapat dibuat container dan diletakkan pada container lain , container level tinggi disebut dengan toplevel-container yang merupakan windows utama dari sebuah tampilan GUI. Untuk membuat windows utama kita bisa memanfaatkan kelas Jframe, Jdialog atau Japplet yang merupakan kelas-kelas top-level-container, setiap kita membuat program GUI minimal kita harus memiliki atau membuat 1container.

Subkelas dari container:

- Panel : Container paling ideal dan sering dogunakan untuk memuat kompenen dan container lainnya dalam suatu hierarki
- Applet : Panel khusus yang digunakan untuk membuat program yang dijalankan pada browser internet
- Windows : Kelas pada top level windows yang tidak memilioki title border
- Frame : Kelas pada top level windows yang memiliki title bar, menu bar, border,selain itu juga frame dapat diubah-ubah ukurannya (resize) dan dapat dipindahkan
- Dialog : Kelas pada top level windows yang memiliki satu title bar dan suatu border
- FRAME, program berikut akan memperlihatkan contoh pembuatan frame/tampilan windows

```
import javax.swing.*;
public class Windowsku
{
static JFrame frame;
public static void main(String [] args)
{
 frame = new JFrame("Ruhe...in action..");
 frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setSize(300,
100);//lebar,tinggi windows
frame.setLocation(200,200);//x,y tampiln pada windows
frame.setVisible(true);
}
}
```

Hasil:

Dalam merancang sebuah program berbasis GUI kita membutuhkan suatu tools sebagai pendukung untuk tiap vent yang kita inginkan dalam java semua tools untuk pemrograman GUI terangkum dalam kelas Component yang natinya akan diwariskan pada tiap program yang mengaksesnya

Kelas Component:

Kelas Component merupakan kelas abstrak dari semua komponen yang digunakan untuk merancang GUI, kelas ini merupakan kelas super dari semua komponen GUI memiliki beberapa subkelas yang merupakan kelas konkrit dari komponen-komponen konkrit GUI, selain itu juga kelas ini memiliki basic support untuk event handling , pengubahan ukuran komponen, pengaturan ont, warna dan lain-lain.

Subkelas dari Component:

Button : Tombol dengan label teks dan akan mentriger event ketika user mengklik/menekan tombol tersebut

1. Canvas

Komponen yang digunakan untuk menggambar dan merancang GUI.

2. Checkbox

Komponen yang menyediakan suatu set pilihan yang dapat di toggle on/off. Checkbox dapat dikelompokkan menjadi radio button.

3. Choice

Komponen yang memberikan menu yang dapat muncul secara otomatis (pop- up).

4. Label

Komponen yang digunakan untuk menampilkan yang hanya dapat dibaca saja.

5. List

Komponen yang berupa list pilihan dapat digulung(scroll).

6. Scrollbar

Komponen yang dapat digeser secara vertical maupun horizontal untuk menunjukkan posisi atau suatu nilai.

7. TextField

Komponen yang digunakan untuk menulis text yang berukuran kecil.

8. TextArea

Sama seperti TextField, namun memiliki kapasitas yang lebih besar. Sehingga misalnya kita ingin membuat komponen button kita harus membuat objek dulu dari Component lalu memanggil kelas JButton.

Berikut ini kita akan membuat salah satu komponen dari GUI yaitu button :

```
import javax.swing.*;
import java.awt.*;
public class buttonku
{
static JFrame frame;
public static void main (String args [])
import javax.swing.*;
        import java.awt.*;
        public class buttonku
        {
        static JFrame frame;
        public static void main (String args [])
        {
        JLabel label;
        }
        JButton button;
        frame = new JFrame("action dengan button..");
        Container containerPane = new Container();
        containerPane = frame.getContentPane();
        button = new JButton("ruhe..");
        containerPane.add(button);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        rame.pack();
        frame.setLocation(200,200);//x,y tampiln pada windows
        frame.setVisible(true);
        }
        }
```

Hasilnya:

Message BOX, disebut juga dialog, berikut kita akan membuat sebuah message box sederha

```
// Mengimport kelas JOptionPane dari package javax.swing
import javax.swing.JOptionPane;
public class msgBox {

  /**Main Method*/
  public static void main( String args[] )
  {
    // Menampilkan pesan dengan Message Dialog dari
    // kelas JOptionPane
    JOptionPane.showMessageDialog(
      null, "hallo Lab-programing....","contoh dialog", 1);
    System.exit( 0 );  // Keluar dari program
  }
}
```

Hasilnya:

Layout Management

Dalam merancang sebuah program kita tidak hanya membutuhkan komponenkomponen apa saja yang harus digunakan tetapi kita juga harus menbgatur komponenkomponen tersebuat dalam lcaontainer, pengaturan komponen-komponen inilah disebuat dengan Layout Management Dalam layout management kelas-kelas komponen didalamnya diperlakukan sebagai skenario tata letak (layout), bukan seperti komponenkomponen pada GUI seperti JButton,Jlist,Jframe dll, jadi hasil dari Layout management ini tidak akan berupa tampilan grafis tapi hanya pengaturan letak komponen-komponen GUI pada top-levelcontainer.

Java menyediakan 5 kelas dalam perancangan GUI. Kelas–kelas tersebuat antara lain sebagai berikut :

a. FlowLayout :Penentuan tata letak komponen didasarkan pada baris ,yaitu mulai dari kiri ke kanan dan baris atas ke baris bawah
b. GridLayout :Penentuan letak komponen didasarkan pada persegi panjang Grid
c. BorderLayout :Penantuan tata letak komponen didasarkan pada lima posisi yaitu east,west,north,south dan center
d. CardLayout : Penentuan tata letak komponen diperlakukan mirip dengan tumpukan kartu dimana akan bergantian , yang visible hanyalah komponen yang terletak di bagian atas
e. GridBagLayout : Penentuan tata letak komponen yang menggunakan prinsip yang sama pada GridLayout, namun komponen dapat menempati multiple cell

Langkah-langkah merancang Layout:

a. Membuat objek container
b. Meregristasi Layout Manager ke objek kontainer dengan memanfaatkan method setLayout(objek) pada container
c. Menambahkan komponen-komponen pada objek container
d. Menambahkan objek container itu ke induk container

Berikut ini contoh penataan letak komponen menggunakan kelas FlowLayout import

```
import javax.swing.JFrame; import
java.awt.Container; import
java.awt.FlowLayout; public class
Layout1 extends JFrame
{ public Layout1()
  {
Container container = getContentPane();
   javax.swing.JButton; for (int x=0; x<8; x++)
{ int noButton = 1;
     container.add(new JButton("Button "  + (noButton += x)));
  } }
 public static void main(String[] args)
 {
  Layout1 frame = new Layout1(); frame.setTitle("tata letak
  Flow Layout");
  frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  frame.setSize(800, 150); frame.setVisible(true);
 }
}
container.setLayout(new FlowLayout(FlowLayout.LEFT, 10, 25));
```

Hasilnya:

Event Handling

Event adalah sebuah proses atau aksi yang dilakukan oleh user saat user menggunakan perangkat I/O seperti mouse, keyboard, setiap objek dari komponen GUI dapat merespon event, dan itu dapat dilakukan dengan cara mengimplementasi suatu interface tertentu yang sesuai, kemudian diterapkan sebagai event listener pada event source.

Berikut ini kita akan menggabungkan 3 contoh program diatas, event nya adalah penekanan button lalu akan ditampilkan sebuah messagebox

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;

public class Event1 extends JFrame
{ private JButton btnText; public Event1()
  { super("Demo event-1"); Container c = getContentPane();
    c.setLayout( new FlowLayout() );
    // Menugaskan objek-objek Jbutton btnText = new JButton("ini
    Button"); c.add(btnText);
    // membuat objek (instans) dari inner class ButtonHandler
    // yang digunakan untuk button event handling ButtonHandler handler = new
    ButtonHandler();

    btnText.addActionListener(handler);

    setSize( 300, 100 ); show();
  }
  // Main Method public static void main( String args[] )
  {
    Event1 app = new Event1(); app.addWindowListener( new
    WindowAdapter() { public void windowClosing(WindowEvent e)
        {
          System.exit( 0 );
        }
      } );
  }
  // Inner class untuk "event handling" pada button private
  class ButtonHandler implements ActionListener { public
  void actionPerformed(ActionEvent e)
    {
      JOptionPane.showMessageDialog(  null,"hai  labprograming","hasil
event",1);
} } }
```

Hasilnya:

setelah button diatas ditekan maka event yang akan dihasilkan :

3. KEGIATAN PRAKTIKUM

a. Praktekkan listing program diatas (pada contoh-contoh program didasar teori tersebut)

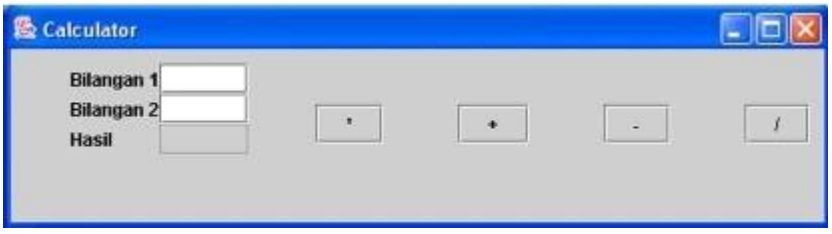
b. Contoh Program sederhana CALCULATOR

```
import java.awt.*;

import java.awt.event.*;

import javax.swing.*;

import java.awt.FlowLayout.*; import

java.awt.Container.*;

public class calculator extends JFrame implements ActionListener

{
private JTextField jTFInput1,jTFInput2,jTFHasil; private JButton
btnkali, btntambah, btnkurang, btnbagi; public static void main
(String args [])

{

calculator frame = new calculator(); frame.pack();

frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

frame.setTitle("Calculator"); frame.setVisible(true);

frame.setSize(550,150);

} public

calculator()

{
```

```
JPanel p1 = new JPanel();
p1.setLayout(new GridLayout(4, 1));
p1.add(new JLabel("Bilangan 1"));
p1.add(jTFInput1 = new JTextField(3));
p1.add(new JLabel("Bilangan 2"));
p1.add(jTFInput2 = new JTextField(3));
p1.add(new JLabel("Hasil"));
p1.add(jTFHasil = new JTextField(4));
jTFHasil.setEditable(false); JPanel p2 =
new JPanel(); p2.setLayout(new
FlowLayout()); p2.add(btnkali = new
JButton(" * "));
                    btnkali.addActionListener(this);
JPanel p3 = new JPanel(); p3.setLayout(new
FlowLayout()); p3.add(btntambah = new JButton("
+ ")); btntambah.addActionListener(this);
JPanel p4 = new JPanel(); p4.setLayout(new FlowLayout());
p4.add(btnkurang = new JButton(" - "));
                    btnkurang.addActionListener(this); JPanel
                    p5 = new JPanel();
p5.setLayout(new FlowLayout()); p5.add(btnbagi
= new JButton(" / "));
                    btnbagi.addActionListener(this); Container container1
                    = getContentPane();
container1.setLayout(new BorderLayout(150,10));
container1.add(p1);
Container container = getContentPane(); container.setLayout(new
FlowLayout(FlowLayout.LEFT,40,10));
container.add(p2); container.add(p3);
container.add(p4); container.add(p5);
}
        public void actionPerformed(ActionEvent e)
{ if (e.getSource()==
btnkali)
```

```
    { double
bilangan1=(Double.parseDouble(jTFInput1.getText().trim())); double
bilangan2=(Double.parseDouble(jTFInput2.getText().trim())); double
hasil = bilangan1 * bilangan2;
jTFHasil.setText(String.valueOf(hasil));
    } else
if (e.getSource()== btntambah)
{
double  bilangan1=(Double.parseDouble(jTFInput1.getText().trim())); double
        bilangan2=(Double.parseDouble(jTFInput2.getText().trim())); double
hasil = bilangan1 + bilangan2;
jTFHasil.setText(String.valueOf(hasil));
} else if (e.getSource()==
btnkurang)
{ double
bilangan1=(Double.parseDouble(jTFInput1.getText().trim())); double
bilangan2=(Double.parseDouble(jTFInput2.getText().trim()));
} else { double hasil = bilangan1 - bilangan2;
jTFHasil.setText(String.valueOf(hasil)); double
bilangan1=(Double.parseDouble(jTFInput1.getText().trim())); double
bilangan2=(Double.parseDouble(jTFInput2.getText().trim()));
}
} } double hasil = bilangan1 / bilangan2; jTFHasil.setText(String.valueOf(hasil));
```

Hasil



4. TUGAS

Kerjakan kembali tugas-tugas pada pertemuan sebelumnya dengan menggunakan konsep GUI.

MODUL 8 CONDITIONAL STATEMENT

[Pert. 8]

1. TUJUAN :

- Mengenal pernyataan if, if..else..
- Mengenal pernyataan if..else.. bertingkat
- Mengenal pernyataan switch..case
- Mengenal penggunaan kondisi majemuk
- Mengimplementasikan OOP pada struktur kondisi

2. DASAR TEORI/KEGIATAN PRAKTIKUM

Catatan : Setelah selesai mencoba potongan program pada kegiatan praktikum ini, implementasikan juga menggunakan konsep OOP untuk kasus yang sama

2.1 Program dengan pernyataan If

Program di bawah adalah program untuk menentukan suatu kelulusan mahasiswa terhadap suatu mata kuliah. Program tersebut menggunakan pernyataan If untuk penentuan keputusannya. Tulis dan jalankan program berikut (**Prak8_1.java**).

a. Buat paket dengan nama percabangan

b. Buat tiga kelas dengan nama masing – masing Model, ProsesPercabanganIf dan MainClass

c. Ketik / tuliskan kode dibawah ini pada kelas masing – masing yang telah tertera pada kode program:

```java
package percabangan;
public class Model {
    private int nilai;
    private int nilaiMatematika;
    private int nilaiFisika;
    public int getNilai() {
        return nilai;
    }
    public void setNilai(int nilai) {
        this.nilai = nilai;
    }
    public int getNilaiMatematika() {
        return nilaiMatematika;
    }
    public void setNilaiMatematika(int nilaiMatematika) {
        this.nilaiMatematika = nilaiMatematika;
    }
    public int getNilaiFisika() {
        return nilaiFisika;
    }
    public void setNilaiFisika(int nilaiFisika) {
        this.nilaiFisika = nilaiFisika;
    }
}
```

```java
package percabangan;
public class ProsesPercabanganIf extends Model {

    public void prosesIf() {
        if (getNilai() < 55) {
            System.out.println("Mahasiswa tidak Lulus");
        }
    }
}
```

```
package percabangan;
import java.util.Scanner;
public class MainClass {
    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        System.out.println("~~~~~~~~~~~~~~ Conditional If Statement ~~~~~~~~~~");
        ProsesPercabanganIf p = new ProsesPercabanganIf();
        System.out.print("Masukkan nilai: ");
        p.setNilai(input.nextInt());
        p.prosesIf();
        System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
    }
}
```

Jika diinputkan nilai < 55, maka akan muncul keterangan mahasiswa tersebut tidak lulus, tetapi jika nilai > atau = 55, maka program akan berhenti tanpa menampilkan suatu pesan.

2.2 Program dengan pernyataan If..Else..

Sempurnakan program di atas dengan menambahkan berikut pada kelas ProsesPercabanganIf.

(**Prak8_2.java**).

```
else{
   System.out.println("Mahasiswa tersebut lulus");}
```

2.3 Program dengan pernyataan If..Else.. bertingkat

Program di bawah adalah program untuk mengkonversi nilai angka ke nilai huruf, yang dilakukan dengan menggunakan pernyatan if..else.. bertingkat. Dengan menggunakan paket dan projek yang sama, tambahkan sebuah **method** dengan nama **prosesIfElse** pada kelas **ProsesPercabanganIf** dengan kode sebagai berikut:(**Prak8_3.java**)

```
public void prosesIfElse() {
    if (getNilai() >= 80) {
        System.out.println("Nilainya A");
    } else if (getNilai() >= 70) {
        System.out.println("Nilainya B");
    } else if (getNilai() >= 55) {
        System.out.println("Nilainya C");
    } else if (getNilai() >= 40) {
        System.out.println("Nilainya D");
    } else {
        System.out.println("Nilainya E");
    }

}
```

Dan ganti pemanggilan method prosesIf pada kelas Mainclass menjadi prosesIfElse, jalankan program! (**p.prosesIf(); ➔ p.prosesIfElse();)**

2.4 Program dengan pernyataan Switch..Case..

Program berikut akan menampilkan tulisan sesuai dengan input yang diberikan, dengan batasan input 1 sampai 3. Tulis kode program berikut pada kelas baru dengan nama ProsesPercabanganSwitchCase (**Prak8_4.java**).

```
1   package percabangan;
2   public class ProsesPercabanganSwitchCase extends Model{
3       public void switchCase() {
4           switch(getNilai()){
5               case 1: System.out.println("Anda memilih angka 1");
6               break;
7               case 2: System.out.println("Anda memilih angka 2");
8               break;
9               case 3: System.out.println("Anda memilih angka 3");
10              break;
11          }
12      }
13  }
```

Tambahkan koding program berikut pada kelas MainClass yang telah dibuat:

```
System.out.println();
System.out.println("~~~~~~~ Conditional Switchcase Statement ~~~~~~~~");
ProsesPercabanganSwitchCase ppsc= new ProsesPercabanganSwitchCase();
System.out.print("Masukkan angka 1-3: ");
ppsc.setNilai(input.nextInt());
ppsc.switchCase();
```

Coba tambahkan pernyataan di bawah, yang diletakkan setelah pernyataan case 3 pada kelas ProsesPercabanganSwitchCase, kemudian inputkan bilangan lebih besar dari 4 pada kelas MainClass. (**Prak8_5.java**)

```
default : System.out.println("Bilangan di luar range");
```

2.5 Program dengan kondisi majemuk

Program berikut adalah program untuk menyeleksi mahasiswa baru lewat jalur SPMB, dimana mahasiswa dinyatakan diterima bila nilai matematikanya di atas 80 dan nilai fisikanya di atas 70. Tulis dan jalankan program di bawah ini pada kelas baru dengan nama KondisiMajemuk (**Prak8_6.java**).

```
1   package percabangan;
2   public class KondisiMajemuk  extends Model{
3       public void prosesKondisiMajemuk() {
4           if ((getNilaiMatematika()>80) && (getNilaiFisika()>70)) {
5               System.out.println("Siswa tersebut diterima");
6           }else{
7               System.out.println("Siswa tersebut tidak diterima");
8           }
9       }
10  }
```

Tambahkan kode program berikut pad akelas MainClass untuk memanggil dan menjalankan kode program diatas:

```
System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
System.out.println("");
System.out.println("~~~~~~~~~~~~~~~~~ Kondisi Majemuk ~~~~~~~~~~~~~~~~~");
KondisiMajemuk km= new KondisiMajemuk();
System.out.println("Masukkan nilai Matematika");
km.setNilaiMatematika(input.nextInt());
System.out.println("Masukkan nilai Fisika");
km.setNilaiFisika(input.nextInt());
km.prosesKondisiMajemuk();
```

3. TUGAS

Soal 8.1 (**Soal8_1.java**)

Buat program untuk menentukan kriteria kegemukan dengan menggunakan Indeks Massa Tubuh (IMT), yang dihitung berdasarkan rumus :

IMT = b / t^2 dengan b : berat badan (kg) dan t : tinggi badan (m)

Kriteria penentuan berat berdasarkan IMT adalah sebagai berikut :

Nilai IMT	**Kriteria**
IMT ≤ 18.5	Kurus
18.5 < IMT ≤ 25	Normal
25 < IMT ≤ 30	Gemuk
IMT > 30	Kegemukan (Obesitas)

Soal 8.2 (**Soal8_2.java**)

Buat program untuk menentukan gaji seorang karyawan berdasarkan waktu/jam kerjanya, dengan aturan bahwa batasan jam kerja normal adalah 50 sampai 60 jam/minggu dengan upah Rp.5000/jam. Jika lebih dari batas tersebut, maka kelebihan perjamnya dibayar Rp 7500/jam. Tetapi jika jam kerjanya kurang dari 50 jam/minggu, maka dikenakan potongan Rp.2500/jam nya. (Input : jam kerja/minggu, dan output : total gaji)

Soal 8.3 (**Soal8_3.java**)

Buat program untuk menentukan kuadran dari suatu titik koordinat. Nilai inputnya adalah nilai X dan Y suatu titik, dan outputnya adalah nomor kuadran dari koordinat tersebut.

MODUL 9 LOOP STATEMENT

[Pert. 9]

1. Tujuan :

- Mengenal pernyataan **for**
- Mengenal pernyataan **break**
- Mengenal variabel **counter**
- Mengenal pernyataan **while** dan pernyataan **do..while**
- Mengimplementasikan OOP pada struktur Perulangan

2. DASAR TEORI/KEGIATAN PRAKTIKUM

Catatan : Setelah selesai mencoba potongan program pada kegiatan praktikum ini, implementasikan juga menggunakan konsep OOP untuk kasus yang sama

2.1 Program dengan pernyataan for

Program di bawah ini akan menampilkan tulisan "Hello World!!" sebanyak 5 kali. Tulis dan jalankan program tersebut (**Prak9_1.java**).

```
01  public class Prak301 {
02     public static void main (String[] args) {
03     for(int i=1; i<5; i++)
04        System.out.println("Hello World!!");
05     }
06  }
```

Program di bawah ini akan menampilkan bilangan genap mulai dari 2 sampai bilangan yang diinputkan oleh pengguna. Tulislah dan jalankan program berikut dengan membuat 3 class dengan nama masing – masing: Model, PerulanganFor,MainClass. (**Prak9_2.java**)

1. Model.java

```
public class Model {

    private int batas;
    private boolean prima= true;
    private int bilangan;
    private int hasil;
    public int getBatas() {
        return batas;
    }

    public void setBatas(int batas) {
        this.batas = batas;
    }

    public boolean isPrima() {
        return prima;
    }

    public void setPrima(boolean prima) {
        this.prima = prima;
    }

    public int getBilangan() {
        return bilangan;
    }

    public void setBilangan(int bilangan) {
        this.bilangan = bilangan;
    }

    public int getHasil() {
        return hasil;
    }
```

2. PerulanganFor.java

```
package perulangan;

/**
 *
 * @author Dika
 */
public class PerulanganFor extends Model {
    int hasil;
    public void perulanganForBilanganGenap() {
        for (int i = 2; i < getBatas(); i += 2) {
            System.out.println(i + " ");
        }
    }
}
```

3. Main Class.java

```java
package perulangan;

import java.util.Scanner;

/**
 *
 * @author Dika
 */
public class MainClass {
    public static void main(String[] args) {
        Scanner input= new Scanner(System.in);
        System.out.println("~~~~~~~~~~~~~ Looping Statement For ~~~~~~~~~~~~~");
        PerulanganFor pf= new PerulanganFor();
        System.out.print("Masukkan batasan perulangan: ");
        pf.setBatas(input.nextInt());
        pf.perulanganForBilanganGenap();


    }
}
```

2.2 Program dengan pernyataan break

Pernyataan break adalah pernyataan untuk menghentikan perulangan, sehingga akan keluar dari perulangan tersebut walaupun proses perulangan belum berakhir. Berikut ini adalah program untuk menentukan apakah suatu bilangan itu termasuk bilangan prima atau tidak.

Tambahkan dan kode jalankan program dibawah ini: (**Prak9_3.java**).

1. PerulanganFor.java

```java
public class PerulanganFor extends Model {

    int hasil;

    public void perulanganForBilanganGenap() {
        for (int i = 2; i < getBatas(); i += 2) {
            System.out.println(i + " ");
        }
    }

    public void perulanganForDenganBreak() {
        for (int i = 2; i < getBilangan(); i++) {
            if ((getBilangan() % i) == 0) {
                setPrima(false);
                break;
            }
        }
        if (isPrima()) {
            System.out.println(getBilangan() + " adalah bilangan PRIMA");
        } else {
            System.out.println(getBilangan() + " adalah BUKAN bilangan PRIMA");
        }
    }
}
```

2. MainClass.java

```
6   package perulangan;
7
8   import java.util.Scanner;
9
10  /**
11   *
12   * @author Dika
13   */
14  public class MainClass {
15      public static void main(String[] args) {
16          Scanner input= new Scanner(System.in);
17          System.out.println("~~~~~~~~~~~~~ Looping Statement For ~~~~~~~~~~~~~");
18          PerulanganFor pf= new PerulanganFor();
19          System.out.print("Masukkan batasan perulangan: ");
20          pf.setBatas(input.nextInt());
21          pf.perulanganForBilanganGenap();
22          System.out.print("Masukkan sebuah bilangan: ");
23          pf.setBilangan(input.nextInt());
24          pf.perulanganForDenganBreak();
25
26
27      }
28  }
```

2.3 Program dengan variabel counter

Variabel counter adalah suatu variabel yang menyimpan hasil operasi secara kontinyu (mis : hasil = hasil + 5 atau ditulis hasil += 5). Berikut adalah program untuk menjumlahkan bilangan sampai dengan bilangan yang diinputkan oleh pengguna.Tambahkan dan jalankan code program di bawah ini: (**Prak9_4.java**).

1. PerulanganFor.java

```
12  public class PerulanganFor extends Model {
13      int hasil;
14      public void perulanganForBilanganGenap() {
15          for (int i = 2; i < getBatas(); i += 2) {
16              System.out.println(i + " ");
17          }
18      }
19
20      public void perulanganForDenganBreak() {
21          for (int i = 2; i < getBilangan(); i++) {
22              if ((getBilangan() % i) == 0) {
23                  setPrima(false);
24                  break;
25              }
26          }
27          if (isPrima()) {
28              System.out.println(getBilangan() + " adalah bilangan PRIMA");
29          } else {
30              System.out.println(getBilangan() + " adalah BUKAN bilangan PRIMA");
31          }
32      }
33
34      public void perulanganForDenganVariabelCounter(){
35          for (int i = 1; i < getBatas(); i += 2) {
36              hasil+=i;
37          }
38          setHasil(hasil);
39          System.out.println(getHasil());
40      }
41
42  }
```

2. MainClass.java

```
6   package perulangan;
7
8   import java.util.Scanner;
9
10  /**
11   *
12   * @author Dika
13   */
14  public class MainClass {
15      public static void main(String[] args) {
16          Scanner input= new Scanner(System.in);
17          System.out.println("~~~~~~~~~~~~ Looping Statement For ~~~~~~~~~~~~");
18          PerulanganFor pf= new PerulanganFor();
19          System.out.print("Masukkan batasan perulangan: ");
20          pf.setBatas(input.nextInt());
21          pf.perulanganForBilanganGenap();
22          System.out.print("Masukkan sebuah bilangan: ");
23          pf.setBilangan(input.nextInt());
24          pf.perulanganForDenganBreak();
25          System.out.print("Masukkan batasan perulangan: ");
26          pf.setBatas(input.nextInt());
27          pf.perulanganForDenganVariabelCounter();
28          System.out.println("~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~");
29
30
31      }
32  }
```

2.4 Program dengan pernyataan while

Pernyataan perulangan dengan while, umumnya digunakan untuk perulangan yang belum dapat dipastika jumlah perulangannya. Berikut contoh program untuk menjumlahkan suatu bilangan sampai input yang dimasukkan adalah bilangan 0. Tulis dan jalankan program berikut (**Prak9_5.java**).

```
14  public class PerulanganWhile extends Model{
15      Scanner scan = new Scanner(System.in);
16
17      public void whileDo() {
18          int bil=9;
19          int n = 0;
20          int total=0;
21          while (bil!=0 && n<9) {
22              n++;
23              System.out.println("Masukkan bilangan ke "+n+": ");
24              bil=scan.nextInt();
25              total+=bil;
26          }
27          System.out.println("Total jumlah "+(n-1)+"bilangan tersebut: "+total);
28      }
29
30      public static void main(String[] args) {
31          PerulanganWhile pw= new PerulanganWhile();
32          pw.whileDo();
33      }
34
35  }
```

2.5 Program dengan pernyataan do..while

Pernyataan do..while memiliki prinsip kerja yang sama seperti pernyataan while. Perbedaannya hanya pada pengujian kondisi perulangan, dimana do..while pengujiannya dilakukan pada akhir blok perintah. Buatlah program di atas dengan menggunakan pernyataan do..while (**Prak9_6.java**).

3. TUGAS

Soal 9.1 (**Soal9_1.java**)

Buat program untuk menghitung angsuran hutang, dimana diinputkan besar hutang/pinjaman, lamanya angsuran (bulan), dan besarnya bunga perbulan. Bunga dihitung dari hutang/pinjaman yang tersisa. Tampilan daftar pembayaran mulai dari pembayaran pertama sampai terakhir (lunas).

Soal 9.2 (**Soal9_2.java**)

Buat program untuk mengkonversi bilangan bulat ke bilangan romawi dengan input bilangan bulat dengan range 1- 5000.

Soal 9.3 (**Soal9_3.java**)

Buat program untuk mencetak bilangan prima mulai dari bilangan prima pertama (2) sampai dengan bilangan prima terakhir, yang banyaknya bilangan primanya ditentukan oleh pengguna.

MODUL 10
ARRAY/LARIK
[Pert. 10-11]

1. TUJUAN :

- Mengenal tipe data Array dan array 2 dimensi
- Mengimplementasikan konsep OOP pada Array

2. KEGIATAN PRAKTIKUM

2.1 Program dengan Array 1 dimensi

Program berikut adalah program untuk mencari nilai rata-rata sekelompok bilangan, dimana bilangan yang akan dirata-rata dimasukkan terlebih dahulu ke dalam array. (**Prak10_1.java**).

1. Class Model (Model.java)

```
1  package array;
2  public class Model {
3  private int jumbil;
4  private double total;
5  private double rata;
6  private double data[];
7      public int getJumbil() {
8          return jumbil;
9      }
10     public void setJumbil(int jumbil) {
11         this.jumbil = jumbil;
12     }
13     public double getTotal() {
14         return total;
15     }
16     public void setTotal(double total) {
17         this.total = total;
18     }
19     public double getRata() {
20         return rata;
21     }
22     public void setRata(double rata) {
23         this.rata = rata;
24     }
25     public double[] getData() {
26         return data;
27     }
28     public void setData(double[] data) {
29         this.data = data;
30     }
31 }
```

2. Class Array 1D (Array1Dimensi.java)

```java
package array;
import java.util.Scanner;
public class Array1Dimensi extends Model{
    public void hitungMean(){
        Scanner scan= new Scanner(System.in);
        double data[] = new double[getJumbil()];
        double total=0;

        for (int i = 0; i < getJumbil(); i++) {
            System.out.print("Masukkan data ke-"+(i+1)+": ");
            data[i]= scan.nextInt();
            total+=data[i];
        }
        setTotal(total);
        System.out.println("Nilai rata - rata dari ke-" +getJumbil()+" bilangan adalah: ");
        for (int i = 0; i < (getJumbil()); i++) {
            setRata(total/getJumbil());
        }
        System.out.println(getRata());
    }
}
```

3. Class Main (MainClass.java)

```java
package array;
import java.util.Scanner;
public class MainClass {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        Array1Dimensi mc= new Array1Dimensi();
        System.out.print("Masukkan banyak bilangan: ");
        mc.setJumbil(scan.nextInt());
        mc.hitungMean();
    }
}
```

2.2 Program dengan Array 2 dimensi

Program berikut adalah program untuk menjumlahkan dua buah matriks. Untuk membuat suatu matriks, maka digunakan array 2 dimensi. Tulis dan jalankan program berikut (**Prak10_2.java**).

```java
package array;
import java.util.Scanner;
public class Array2D {
    Scanner masukan;
    int baris, kolom;
    public void proses() {
        masukan= new Scanner(System.in);
        System.out.println("Masukkan data : ");
        System.out.println("Matriks 1 : ");
        int matriks1[][] = new int[baris][kolom];
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                System.out.print("Bil baris ke-" + i + " kolom ke-" + j + " : ");
                matriks1[i][j] = masukan.nextInt();
            }
        }
        System.out.println();
        System.out.println("Matriks 2 : ");
        int matriks2[][] = new int[baris][kolom];
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                System.out.print("Bil baris ke-" + i + " kolom ke-" + j + " : ");
                matriks2[i][j] = masukan.nextInt();
            }}
        System.out.println("Data sebelum dijumlahkan : ");
        System.out.println("Matriks 1 : ");
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                System.out.print(matriks1[i][j] + "  ");
            }
            System.out.println();
        }
        System.out.println();
        System.out.println("Matriks 2 : ");
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                System.out.print(matriks2[i][j] + "  ");
            }
            System.out.println();
        }
        System.out.println();
        int hasil[][] = new int[baris][kolom];
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                hasil[i][j] = matriks1[i][j] + matriks2[i][j];
            }
        }
        System.out.println("Hasil penjumlahan Matriks tersebut : ");
        for (int i = 0; i < baris; i++) {
            for (int j = 0; j < kolom; j++) {
                System.out.print(hasil[i][j] + "  ");
            }
            System.out.println();
        }
        System.out.println();
    }
```

```
57
58      public static void main(String[] args) {
59
60          Array2D ad= new Array2D();
61          ad.masukan = new Scanner(System.in);
62          System.out.println("Program Penjumlahan 2 buah matriks");
63          System.out.print("Masukkan jumlah baris Matriks : ");
64          ad.baris = ad.masukan.nextInt();
65
66          System.out.print("Masukkan jumlah kolom Matriks : ");
67          ad.kolom = ad.masukan.nextInt();
68          System.out.println();
69          ad.proses();
70      }
71
72  }
```

3. TUGAS

Buat sebuah program yang memiliki menu sebagai berikut :

Menu Utama :

1. Input Data
2. Tampilkan Data
3. Distribusi Frekuensi
4. Statistik
5. Keluar

Dimana input data berupa nilai (0-100) ke dalam suatu array. Menampilkan data adalah menampilkan semua isi array. Distribusi frekuensi adalah mencari nilai frekuensi untuk masing-masing range nilai (dengan interval 20). Statistik adalah mencari nilai mean, minimum dan maksimum dari data.

MODUL 11 REKURSI

[Pert. 12-13]

1. TUJUAN :

- Mengenal proses rekursif
- Mengenal contoh-contoh rekursif
- Mengimplementasikan konsep OOP pada Rekursif

2. DASAR TEORI

Rekursi merupakan algoritma untuk memecahkan masalah menggunakan suatu fungsi yang memanggil dirinya sendiri. Sebenarnya permasalahan yang dapat diselesaikan menggunakan rekursi dapat diselesaikan tanpa rekursi namun hal tersebut sulit untuk dilakukan.

Beberapa maslah klasik yang bisa di selesaikan menggunakan rekursi adalah : **a)**

Factorial

Bila anda pernah mempelajari teori kemungkinan (probabilitas) tentu tidak asing dengan konsep factorial. System perhitungan factorial memiliki ketentuan sebagai berikut:

```
0! = 1
```

```
n! = n x (n-1) x (n-2) …. (n-k) dengan n >= k
```

Berdasarkan ketentuan di atas, bisa anda tuliskan beberapa konsekuensi yang menyertainya. Jika nilai factorial suatu bilangan sama dengan hasil perkalian bilangan tersebut dengan bilanganbilangan sebelumnya, selama bernilai positif, maka :

```
(n-2)! = (n-2) x (n-3) x (n-4) …

(n-1)! = (n-1) x (n-2) …
      n! = n x (n-1)!
```

Dengan memperhatikan pola di atas, cara yang paling efisien untuk mengalikan bilangan-bilangan factorial adalah dengan membuat fungsi perkalian yang dapat memanggil dirinya sendiri. Proses tersebut harus dikerjakan dengan indek yang menurun dan akan berhenti setelah tercapai n = k.

```
Factorial(int n){
If (n==0)
            return 1
else
  return n*factorial(n-1);
Factorial
}
```

b) Deret Fibonacci

Deretan Fibonacci adalah deretan bilangan yang dihasilkan dari penjumlahan dua bilangan sebelumnya. Deret ini dimulai dengan bilangan 0 dan 1, dan ber-tutut-turut akan dihasilkan bilangan (0+1), (1+1), (1+2),(2+3), dan seterusnya.

0, 1, 1, 2, 3, 5, 8, 13, 21, ……

Bilangan-bilangan yang menyusun deret Fibonacci disebut juga sebagai bilangan Fibonacci. Selanjutnya akan dibuat method sedemikian rupa, sehingga bisa mengacu kepada suatu bilangan Fibonacci. Proses ini dilakukan dengan memanggil nama method tersebut dan menyebutkan indeksnya, contoh fib(0), fib(3), fib(100), dan sebagainya.
Berdasarkan pola Fibonacci,

Fib(0) = 0;

Fib(1) = 1;

Fib(n) = fib(n-1) + fib(n-2); n >= 2

Sekali lagi, diperlukan pola rekursi untuk memecahkan masalah tersebut. Pola Fib(n) = fib(n1) + fib(n-2); n >= 2 bermakna untuk mencari bilangan Fibonacci ke-n. Terlebih dahulu harus mengerjakan proses perhitungan untuk dua bilangan sebelum indeks ke n (dengan indeks menurun). Proses akan terus dikerjakan dan berakhir bila n = 2 atau menghasilkan fib(0).

Contoh : menghitung nilai bilangan Fibonacci ke-7. Maka urutannya adalah:

Fib(7) = fib(6) + fib(5);

Fib(6) = fib(5) + fib(4);

Fib(5) = fib(4) + fib(3);

Fib(4) = fib(3) + fib(2);

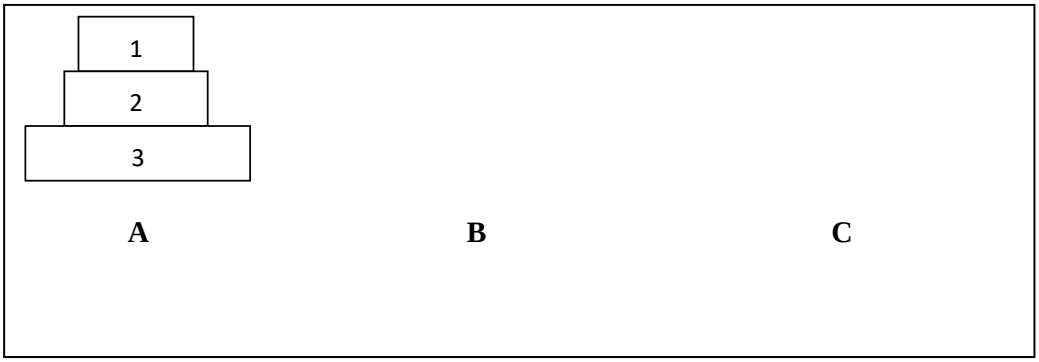
Fib(3) = fib(2) + fib(1);
Fib(2) = fib(1) + fib(0);

Fib(1) = 1;

Fib(0) = 0;

c) Menara Hanoi

Menara Hanoi adalah istilah yang di gunakan untuk menggambarkan suatu tumpukan balok atau cakram yang tersusun seperti sebuah pyramid. Sebagai ilustrasi, tinjaulah menara Hanoi yang tersusun dari 3 balok. Pada bagian dasarnya, terdapat balok dengan penampang terluas, dan di bagian tengahnya, balok dengan luas penampang di antara keduanya.



Gambar 1. Menara Hanoi dengan 3 balok

Ide selanjutnya yang menjadi inti permasalahan dalam menara Hanoi adalah memindahkan balokbalok tersebut ke posisi baru, sedemikian rupa sehingga pada posisi tersebut, balok-balok itu akan tersusun kembali menjadi menara Hanoi yang sama.

Aturan pemindahan menara Hanoi: memindahkan seluruh balok dari posisi A ke B. pada setiap langkah hanya boleh satu blok dipindahkan. Tidak boleh balok kecil menjadi alas bagi balok besar. Posisi C boleh dijadikan transit sementara. Posisi A dan B bisa juga di jadikan transit sementara , apabila saat proses pemindahan masih berlangsung, semua blok terkumpul pada dua posisi lainnya.

3. KEGIATAN PRAKTKUM

3.1. Rekursif pada Faktorial

Berikut ini adalah contoh program rekursif untuk menghitung nilai faktorial suatu bilangan. Tulis dan jalankan program berikut. (**Prak11_1.java**). Setelah berhasil menjalankan progarm yang telah dibuat, lakukan modifikasi dengan menggunakan konsep OPP !

```
01  import java.util.Scanner;
02  public class Prak404 {
03     public static void main (String[] args) {
04       Scanner masukan = new Scanner(System.in);
05       int bil, hasil;
06       System.out.print("Masukkan suatu bilangan : ");
07       bil = masukan.nextInt();
08       hasil = faktorial(bil);
09       System.out.println("Nilai faktorial "+ bil +" adalah "+ hasil);
10     }
11
12     private static int faktorial(int a){
13        if (a==1)
14           return 1;
15        Else
16           return (a * faktorial(a-1));
17     }
18  }
```

3.2. Buatlah program menggunakan konsep OOP untuk memecahkan masalah pada deret fibonacci dan menara hanoi

PROJECT AKHIR

[Pert. 14]

Buat program Sistem Informasi Perhotelan dengan menggunakan Java OOP, dimana data yang disimpan adalah :

1. Data kamar, yang berisi data jumlah kamar yang dimiliki oleh hotel tersebut, termasuk status dari kamar tersebut, apakah terisi, dibooking atau kosong. Kamar yang ada terdiri dari tiga jenis kamar, yaitu single (harga sewanya Rp. 175.000,-), double (harga sewanya Rp. 225.000,-) dan suite (harga sewanya Rp. 300.000,-).
2. Data penyewa, yang berisi data tamu yang menginap di hotel tersebut, yang minimal menyimpan data no KTP/SIM, nama, alamat, dan no telepon.
3. Data transaksi, yang berisi proses transaksi antara penyewa dan kamar yang disewa

Adapun fasilitas/proses yang dimiliki oleh program ini adalah sebagai berikut :

1. Dapat menampilkan data kamar beserta statusnya. Termasuk memberikan summary berapa jumlah kamar yang terisi, yang kosong, dan yang dibooking
2. Dapat menampilkan data penyewa termasuk kamar yang disewa. Juga dapat memberikan summary tentang penyewa yang sering menginap di hotel tersebut
3. Dapat mengetahui total transaksi yang dilakukan pada suatu waktu tertentu (perbulan atau pertahun tertentu) 4. Tambahan fasilitas lain akan memberikan nilai tambah.

Untuk proses pemilihan proses digunakan sistem menu, yang formatnya ditentukan sendiri oleh pembuat program.

REFERENSI

- Java Software Solutions 6th Edition Lewis, John & Loftus, William. 2009. Pearson International
- Dasar Pemrograman Java 2 Kadir, Abdul. 2004. Yogyakarta: Penerbit Andi Offset
- Tuntunan Pemrograman Java Jilid 3 Purnama, Rangsang. 2005. Jakarta: Prestasi Pustaka
- Absolute Java 3rd Editon Savitch, Walter. 2008. Pearson International
- Belajar Pemograman dengan Bahasa C++ dan Java, M.Shalahuddin dan Rosa A.S, .2007. Bandung:Informatika