MODUL PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

Erma Susanti
Erna Kumalasari Nurnawati

ISBN: 978-602-7619-43-2

Java

AKPRIND PRESS

MODUL PRAKTIKUM

PEMROGRAMAN BERORIENTASI OBJEK

ISBN: 978-602-7619-43-2

Penulis : Erma Susanti, Erna Kumalasari Nurnawati
Desain Cover : Erna Kumalasari Nurnawati
Diterbitkan oleh : AKPRIND PRESS

KARTU PRAKTIKUM
PEMROGRAMAN TERSTRUKTUR

Nim : _____

Nama : _____

Minggu ke	Tanggal	Materi	Paraf Asisten dan Cap
1.			
2.			
3.			
4.			
5.			
6.			
7.			
8.			
9.			
10.			
INHAL 1			
INHAL 2			
RESPONSI			

TATA TERTIB PRAKTIKUM

1. Praktikum hadir sebelum praktikum dimulai. Keterlambatan ditoleransi 15 menit. Jika keterlambatan lebih dari 15 menit maka nilai yang diperoleh pada praktikum pada sesi tersebut maksimal 75% dari nilai yang diperoleh
2. Praktikan berpakaian sopan, tidak diperkenankan memamakai kaos oblong, topi serta tidak diijinkan makan di ruang laboratorium (minum boleh tetapi membawa sendiri).
3. **Mengisi daftar hadir dan meminta paraf dan Cap asisten pada kartu praktikum** anda
4. Praktikan tidak diijinkan menggunakan flasdisk
5. Praktikan WAJIB memiliki dan menggunakan modul terbaru. Modul bisa diperoleh di laboratorium dengan mengganti biaya cetak (Kartu praktikum terdapat di dalam modul). **Modul WAJIB dibawa saat praktikum.**
6. Praktikan menjaga kebersihan ruangan dan merapikan komputer, kursi dan peralatan lain setelah praktikum selesai. Matikan komputer dengan prosedur yang benar, serta mematikan stabilizer
7. Demi keamanan, praktikan hendaknya memakai sandal yang disediakan lab. Kembalikan sandal ditempatnya setelah selesai praktikum. Sandal tidak diijinkan digunakan di luar ruangan.
8. Penilaian praktikum dilakukan selama 8 minggu, sejak minggu ke 2 hingga minggu ke 9. Nilai max 90 (karena anda dibimbing asisten dan tidak ada tugas yang 100% anda kerjakan sendiri) dengan bobot masing-masing 10%. Nilai responsi max 100 berbobot 20%. Nilai anda akan diakumulasi dan menjadi nilai tugas ke 4 untuk matakuliah masing-masing. Tidak ada KETIDAK LULUSAN dalam praktikum, tetapi nilai praktikum mempunyai bobot 20% dari nilai matakuliah
9. Kehadiran praktikum harus mencapai 75% (mutual dengan kehadiran kuliah), sehingga apabila kehadiran kurang, maka anda tidak bisa mengikuti ujian akhir. Jika anda tidak hadir, anda diharapkan mengikuti inhal. Mutual dengan perkuliahan, anda diijinkan mengikuti inhal 2x, dan nilai dan kehadiran akan disinkronkan dengan nilai yang anda peroleh. Jika ketidakhadiran lebih dari 2 kali, maka yang bisa diinhal hanya 2x saja, sisanya dianggap tidak hadir dan nilai nol. Nilai inhal max 75.
10. Demikian tata tertib praktikum harap mahasiswa menyesuaikan

KATA PENGANTAR

Puji Syukur ke hadirat Allah Tuhan Yanag Maha Esa yang melimpahkan karunianya sehingga perbaikan modul Pemrograman Terstruktur dapat direvisi dengan cukup signifikan.

Modul ini disusun sebagai pegangan pada praktikum Pemrograman Berorientasi Objek dan diharapkan dapat membantu mahasiswa melaksanakan kegiatan praktikum dengan lebih baik. Mengingat kelengkapan modul ini, maka mahasiswa juga dapat menggunakan modul sebagai diktat kuliah Pemrograman Berorientasi Objek di kelas. Dengan menggunakan modul ini diharapkan mahasiswa dapat mengembangkan logika berpikir dan kemampuan melakukan coding dalam bahasa JAVA.

Modul ini dilengkapi dengan banyak banyak latihan yang harus dikerjakan mahasiswa di labroratorium. Diharapkan dengan banyaknya latihan makin meningkatkan kemampuan pemrograman mahasiswa dan mengasah logika penrograman.

Materi pada modul ini meliputi pengenalan Bahasa C++ sebagai bahasa pemrograman terstruktur, bagaimana struktur pemrograman, mengenal tipe data, operator, identifier dan sintak-sintak bahasa pemrograman. Setelah itu mahasiswa diberikan materi statemen input, output, percabagan dan perulangan. Kemudian dilanjutkan dengan materi atribut,hak akses atribut, kelas, perulagan, kondisional, metod, konstruktor, overloading, pakage, interface, abstrak..

Kami mengucapkan terimakasih atas semua pihak yang berkontribusi atas terselenggaranya revisi modul ini, terutama asisten di Lab Pemrograman Dasar. Semoga bisa berguna dan bisa menjadi acuan belajar Pemrograman Berorientasi Objek dengan bahasa java. Jika ada pertanyaan, kritik dan saran bisa disampaikan ke ernakumala@akprind.ac.id atau erma@akprind.ac.id

Yogyakarta, Februari 2019

Penyusun.

Erma Susanti
Erna Kumalasari Nurnawati

DAFTAR ISI

MODUL KE-1

Pengenalan JAVA & Object Oriented Programming

1.1 Perangkat keras

Pada praktikum pemrograman berorientasi objek di laboratorium komputer 1 IST AKPRIND, digunakan perangkat keras komputer dengan spesifikasi sebagai berikut :

Processor Intel Core 2 Duo, 2,93 GHZ

DDR 2 RAM 2GB

250 GB Hardisk

1.2 Perangkat lunak

Pada praktikum pemrograman berorientasi objek di laboratorium komputer 1 IST AKPRIND, digunakan perangkat lunak komputer dengan spesifikasi sebagai berikut :

Compiler : Java Standard Edition (J2SE) versi 1.8.0

IDE : Netbeans 8.0.2

1.3 JAVA

Java adalah bahasa pemrograman yang dapat dijalankan di berbagai komputer termasuk telepon genggam. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (general purpose), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda. Saat ini java merupakan bahasa pemrograman yang digunakan dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi berbasis web. (*Sumber : Wikipedia*)

1.3 Netbeans IDE

NetBeans IDE adalah sebuah lingkungan pengembangan untuk pemrogram menulis, mengompilasi, mencari kesalahan dan menyebarkan program. Netbeans IDE ditulis dalam Java - namun dapat mendukung bahasa pemrograman lain. Terdapat banyak modul untuk memperluas Netbeans IDE. Netbeans IDE adalah sebuah produk bebas dengan tanpa batasan bagaimana digunakan.

Tersedia juga *NetBeans Platform*; sebuah fondasi yang modular dan dapat diperluas yang dapat digunakan sebagai perangkat lunak dasar untuk membuat aplikasi desktop yang besar. Mitra ISV menyediakan plug-in bernilai tambah yang dapat dengan mudah diintegrasikan ke dalam Platform dan dapat juga digunakan untuk membuat kakas dan solusi sendiri.

Kedua produk adalah kode terbuka (open source) dan bebas (free) untuk penggunaan komersial dan non komersial. Kode sumber tersedia untuk guna ulang dengan lisensi Common Development and Distribution License (CDDL). (*Sumber : Wikipedia*)

1.4 Object Oriented Programming

Pemrograman berorientasi objek (Inggris: object-oriented programming disingkat OOP) merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya.

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung OOP mengklaim bahwa OOP lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan OOP lebih mudah dikembangkan dan dirawat.

Konsep dasar Pemrograman Berorientasi Objek

1.4.1 Kelas

Kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebagai contoh 'class of dog' adalah suatu unit yang terdiri atas definisi-definisi data dan fungsi-fungsi yang menunjuk pada berbagai macam perilaku/turunan dari anjing. Sebuah class adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi object. Sebuah class secara tipikal sebaiknya dapat dikenali oleh seorang non-programmer sekalipun terkait dengan domain permasalahan yang ada, dan kode yang terdapat dalam sebuah class sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak menggunakan OOP). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.

1.4.2 Objek

membungkus data dan fungsi bersama menjadi suatu unit dalam sebuah program komputer; objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek.

1.4.3 Abstraksi

Kemampuan sebuah program untuk melewati aspek informasi yang diproses olehnya, yaitu kemampuan untuk memfokus pada inti. Setiap objek dalam sistem melayani sebagai model dari "pelaku" abstrak yang dapat melakukan kerja, laporan dan perubahan keadaannya, dan berkomunikasi dengan objek lainnya dalam sistem, tanpa mengungkapkan bagaimana kelebihan ini diterapkan. Proses, fungsi atau metode dapat juga dibuat abstrak, dan beberapa teknik digunakan untuk mengembangkan sebuah pengabstrakan.

1.4.3 Enkapsulasi

Memastikan pengguna sebuah objek tidak dapat mengganti keadaan dalam dari sebuah objek dengan cara yang tidak layak; hanya metode dalam objek tersebut yang diberi

izin untuk mengakses keadaannya. Setiap objek mengakses interface yang menyebutkan bagaimana objek lainnya dapat berinteraksi dengannya. Objek lainnya tidak akan mengetahui dan tergantung kepada representasi dalam objek tersebut.

1.4.4 Polimorfisme melalui pengiriman pesan

Tidak bergantung kepada pemanggilan subrutin, bahasa orientasi objek dapat mengirim pesan; metode tertentu yang berhubungan dengan sebuah pengiriman pesan tergantung kepada objek tertentu di mana pesa tersebut dikirim. Contohnya, bila sebuah burung menerima pesan "gerak cepat", dia akan menggerakan sayapnya dan terbang. Bila seekor singa menerima pesan yang sama, dia akan menggerakkan kakinya dan berlari. Keduanya menjawab sebuah pesan yang sama, namun yang sesuai dengan kemampuan hewan tersebut. Ini disebut polimorfisme karena sebuah variabel tungal dalam program dapat memegang berbagai jenis objek yang berbeda selagi program berjalan, dan teks program yang sama dapat memanggil beberapa metode yang berbeda di saat yang berbeda dalam pemanggilan yang sama. Hal ini berlawanan dengan bahasa fungsional yang mencapai polimorfisme melalui penggunaan fungsi kelas-pertama.

MODUL ke-2

Memahami Output, Proses dan Input

Tujuan :

1. Mengenal statement output
2. Mengenal penggunaan variabel
3. Mengenal statement input

2.1.1 Mengenal statement output dengan print dan println

Dalam bahasa pemrograman Java, kita akan menggunakan pernyataan masukan dan keluaran. Pernyataan masukan adalah pernyataan untuk mendapatkan masukan dari keyboard. Sedangkan pernyataan keluaran adalah pernyataan untuk menampilkan sesuatu nilai ke layar.

Komponen keluaran

Untuk menampilkan ke layar secara tekstual, java mempunyai fasilitas output, yaitu dengan perintah :

```
System.out.print( statemen);
        Menampilkan ke layar dan setelah selesai tidak berpindah baris.
System.out.println(statemen);
        Menampilkan ke layar dan setelah selesai berpindah baris.
```

PRAKTIKUM – Output dengan print & println

1. Tulis program berikut dan beri simpan sebagai file dengan nama SelamatBelajar.java, kemudian kompile dan jalankan.

```
public class SelamatBelajar {
    public static void main(String[] args) {
            System.out.println("Selamat Belajar Java");
     }
}
```

Modifikasi program di atas, jika diinginkan hasilnya
Selamat
Belajar Java

2. Tulis program berikut dan beri simpan sebagai file dengan nama coba.java, kemudian kompile dan jalankan.

```
public class coba {
   public static void main(String[] args) {
      System.out.println("abc\ndef");
      System.out.println("abc\tdef");
      System.out.println("\"Halo\"");
   }
}
```

Modifikasi program di atas, jika diinginkan hasilnya

abcdef
Abcdef
Hallo

2.1.2 Mengenal statement input menggunakan JoptionPane

Cara lainnya untuk melakukan output pada java adalah dengan menggunakan kelas JoptionPane. Kelas ini berada pada package javax.swing. Syntax sebagai berikut :

```
javax.swing.JoptionPane.showMessageDialog (null, <Statement yang akan ditampilkan>);
```

Contoh:

```
javax.swing.JoptionPane.showMessageDialog (null, "Nama Saya Erlina, Saya tinggal di Yogyakarta");
```

PRAKTIKUM – output menggunakan JoptionPane

1. Buat kelas dengan nama "latihanIOGUI"

```
public class latihanIOGUI {
   public static void main (String[]args){
      javax.swing.JoptionPane.showMessageDialog(null,"Nama saya
        Erlina, saya tinggal di Yogyakarta");
      javax.swing.JoptionPane.showMessageDialog(null,"Galuh adalah
        teman saya, dia tinggal di Cilacap");
      javax.swing.JoptionPane.showMessageDialog(null,"Taufiq juga
        teman saya, dia tinggal di Klaten");
      javax.swing.JoptionPane.showMessageDialog (null,25+25);
   }
}
```

2. Modifikasi kelas diatas, menjadi :

```
public class latihanIOGUI {
   public static void main (String[]args){
      javax.swing.JoptionPane.showMessageDialog(null,"Nama saya
        Erlina, saya tinggal di Yogyakarta"
      + "\n Galuh adalah teman saya, dia tinggal di Cilacap"
      + "\n Taufiq juga teman saya, dia tinggal di Klaten")
   }
}
```

Lihat apa perbedaannya ?

2.2 Mengenal penggunaan variabel

Tipe data sederhana merupakan tipe inti. Tipe sederhana tidak diturunkan dari tipe lain.

Tipe ini adalah tipe data primitif. Terdapat delapan tipe data primitif di Java:

No	Tipe Data	Kategori	Ukuran	Range
1	int	Integers	4 bytes	2,147,483,648 s/d 2,147,483, 647
2	short		2 bytes	32,768 s/d 32,767
3	long		8 bytes	9,223,372,036,854,775,808 s/d 9,223,372,036,854,775,807
4	byte		1 byte	128 to 127
5	float	Floating Point	4 byte	±3.40282347E+38F
6	double		8 byte	±1.79769313486231570E+308
7	char	Character		
8	boolean	Boolean		True & false

PRAKTIKUM

1. Tulis program berikut dan beri simpan sebagai file dengan nama ketiga.java, kemudian kompile dan jalankan.

```
public class ketiga {
     public static void main(String[] args){
            double luas;
            double alas,tinggi;
        alas=10; tinggi=2;
        luas=1/2.0*alas*tinggi;
        System.out.println("hasil"+luas) ;
     }
}
```

2. Modifikasi program di atas sesuai prigram di bawah ini , apa hasilnya dan kenapa

```
public class ketiga {
     public static void main(String[] args){
            int  luas;
            double alas,tinggi;
        alas=10; tinggi=2;
        luas=1/2.0*alas*tinggi;
        System.out.println("hasil"+luas) ;
     }
}
```

3. Modifikasi program di atas sesuai program di bawah ini , apa hasilnya dan kenapa

```
public class ketiga {
     public static void main(String[] args){
            int  luas;
            double alas,tinggi;
        alas=10;
        tinggi=2;
```

```
            luas=1/2*alas*tinggi;
            System.out.println("hasil" +luas) ;
        }
    }
```

2.3.1 Mengenal statement input menggunakan kelas Scanner

Perintah input di Java bersifat unik, karena perintah yang digunakan dapat lebih dari satu alternatif, salah satu cara yang dapat digunakan adalah dengan menggunakan kelas Scanner. Penggunaan kelas Scanner menghasilkan output dalam bentuk console. Penggunaan syntaks sebagai berikut :

```
Scanner sc=new Scanner(System.in);
```

Setelah diinisisasi, objek sc dapat diterjemahkan menjadi beberapa tipe sebagai berikut:

Tipe data	Contoh penggunaan syntaks
Integer	int a = sc.nextInt();
Byte	byte b = sc.next Byte();
Short	short c = sc.nextShort();
Long	long d = sc.nextLong();
Float	float e = sc.nextFloat();
Double	Double f =sc.nextDouble();
Char/String	String g = sc.next(); String g = sc.nextLine();

PRAKTIKUM – Statement input menggunakan kelas Scanner;

1. Tulis program berikut, dan berikan nama “latihanIO2”

```
public class latihanIO2{
   public static void main (String[]args){
      java.util.Scanner sc=new java.util.Scanner(System.in);
      String nama;
      int umur;
      System.out.println(“Masukkan nama anda : ”);
      nama=sc.nextLine();
      System.out.println(“Masukkan umur anda : ”);
      umur=sc.nextInt();
      System.out.println(“Nama saya ”+nama+”saya ”+umur+”tahun”);
   }
}
```

2.3.2 Mengenal statement input menggunakan kelas JOptionPane

JoptionPane didapatkan dari package javax.swing . JoptionPane membuat kemudahan dengan memunculkan dialog box standar yang memberikan kepada user sebuah nilai atau menginformasikan sesuatu.

Perintah sederhana :

```
String jr =JOptionPane.showInputDialog("statement");
```

Perintah di atas digunakan untuk membuat input dialog JOptionPane, yang akan menampilkan dialog, yang terdiri atas sebuah message, sebuah textfield dan sebuah button OK. Dialog tersebut akan memberikan *return value* String yang akan disimpan di variabel jr .

PRAKTIKUM

1. Tulis program berikut dan beri simpan sebagai file dengan nama …….. , kemudian kompile dan jalankan.

```
public class ketiga {
      public static void main (String[] args) {
            int bil_1, bil_2, hasil;
            bil_1 = 10;
            bil_2 = 500;
            hasil = bil_1 + bil_2;
            System.out.print("Hasil Penjumlahan kedua bilangan :");
            System.out.println(hasil);
      }
}
```

2. Sekarang gunakan perintah INPUT == Joptionpane

```
public class ketiga{
      public static void main (String args[])   {
          String a = javax.swing.JOptionPane.showInputDialog(null,
"Siapa
                        nama anda ?");
            System.out.println("hAllo " + a + " selAmaT bElajaR
Java ..");
      }
}
```

3. Modifikasi program di atas sesuai program di bawah ini, bagaimana programnya untuk melakukan proses penjumlahan dua bilangan

```
public class ketiga{
    public static void main (String args[])   {
       int  a,b,c;
```

```
        String data = javax.swing.JOptionPane.showInputDialog(null,
           "Masukan bilangan pertama ?");
        a=Integer.parseInt(data);
        String data1 = javax.swing.JOptionPane.showInputDialog(null,
           "Masukan bilangan kedua ?");
        b=Integer.parseInt(data1);
        c=a+b;
        System.out.println("hasil penjumlahan " +c);
    }
}
```

Kenapa dan bagaimana program yang benar.

TUGAS PRAKTIKUM

1. Buatlah sebuah program dengan menggunakan bahasa Java untuk mengonversikan nilai suhu dari Celcius ke Reaumur, Kelvin dan Fahrenheit!

   Celcius ke Fahrenheit = (9/5 x celcius) + 32

   Celcius ke Reamur = 4/5 x celcius

   Celcius ke kelvin = c+273
2. Buat program yang meminta inputan suatu nilai rupiah, yang kemudian dihitung nilai US Dollar dan Euro

   Misal 1 US Dolar : Rp. 10.000

   1 Euro : Rp. 14.000

MODUL ke-3
Class & Object

Tujuan :

Mahasiswa mampu mengenal class dan objek

3.1 Mengenal class dan Objek

Class adalah template untuk obyek-obyek yang memiliki sifat yang sama. Dalam program OOP, sebuah obyek bisa didefinisikan setelah suatu class ada terlebih dahulu. Sebagai ilustrasi kita bisa menyebut YAMAHA, HONDA, SUZUKI adalah sebagai alat transportasi. Merk-merk tersebut sering disebut dengan MOTOR.

MOTOR inilah nanti yang di OOP sering disebut dengan suatu class, sedangkan YAMAHA, HONDA, SUZUKI sering disebut dengan obyek. Sebuah class akan mendefinisikan secara keseluruhan dari ciri class tersebut.

Sebuah MOTOR pasti mempunyai ciri ,antara lain

Warna

Tahun produksi

Cc

Proses menghidupkan mesin

Proses mematikan mesin

Ciri tersebut dalam pemrograman berorientasi obyek sering disebut dengan ATRIBUT, proses dalam Oop disebut dengan METHOD. Setiap suatu produk MOTOR dapat dipastikan/ mempunyai kemungkinan mempunyai ciri tersebut diatas.

Cara pendeklarasian kelas :

```
<modifier> class <nama kelas> {
      // body kelas
}
```

Contoh :

```
public abstract class kendaraan{
      // body kelas
}
```

<modifier> dapat menggunakan beberapa modifier yang berbeda

private : modifier yang menandakan kelas hanya dapat diakses dari dalam package.

protected : modifier yang menandakan kelas tersebut dapat diakses dari dalam package dan dari luar package yang masih dalam hirarki pewarisan.

public : modifier yang menandakan kelas tersebut dapat diakses dari dalam maupun dari luar package

abstract : modifier yang menandakan bahwa methos merupakan jenis abstrak.

PRAKTIKUM

1. Tulis program berikut dan beri simpan sebagai file dengan nama …….. , kemudian kompile dan jalankan. Buat new java application project, simpan → pertemuan3

```
package pertemuan3;

class M {
      String warna;
      int tahunProduksi;
      void isiData(){
            System.out.println("Hallo");
      }
}
public class KelasMobil  {
   public static void main(String[] args)   {

            }
}
```

Kenapa hasilnya tidak ada tulisan Hallo ??

Pertanyaan

Apa nama classnya ?

Apa nama metodenya ?

Apa nama atributnya ?

2. Modifikasi program di atas, sesuai instruksi di bawah ini. Apa hasilnya dan kenapa

```
class M  {
      String warna;
      int tahunProduksi;
      void isiData(){
            System.out.println("Hallo");
      }
}
public class ketiga{
      public static void main (String[] args) {
            M mobilku=new M();
            M mobilmu = new M();
            M mobilnya = new M();
            mobilmu. ...();
            mobilku.isiData();
            mobilmu. ...();
      }
}
```

3. Modifikasi program di atas, sesuai instruksi di bawah ini, apa hasilnya dan kenapa?

   Tambahkan objek mobilmu dan objek mobilnya

   Tambahkan bagaimana objek tersebut mengakses metode isiData

4. Tulis program berikut dan beri simpan sebagai file dengan nama …….. , kemudian kompile dan jalankan.

```
class M {
     private String warna;
     int tahunProduksi;
     void isiData(){
          System.out.println("Hallo");
     }
}

public class ketiga{
     public static void main (String[] args){
          M mobilku=new M();
          String warna="hitam";
          mobilku.isiData();
     }
}
```

   Kenapa ???

5. Modifikasi program di atas, sesuai instruksi di bawah ini apa hasilnya dan kenapa :

   Ganti String warna; menjadi private String warna

TUGAS PRAKTIKUM

Tugas Kerjakan dengan program berbasis objek

1. Buat program untuk mencari luas segitiga dengan menggunakan class
2. Buatlah sebuah program dengan menggunakan bahasa Java untuk mengkonversikan nilai suhu dari Celcius ke Reamur, Kelvin dan Fahrenheit!

   Celcius ke Fahrenheit = (9/5 x celcius) + 32

   Celcius ke Reamur = 4/5 x celcius
3. Buat program yang meminta inputan suatu nilai rupiah, yang kemudian dihitung nilai US Dollar dan Euro

   Misal 1 US Dolar : Rp. 10.000

   1 Euro : Rp. 14.000

double nama_variable = Double.parseDouble(nama_variabel1);

MODUL ke-4
Method (non-parameter & berparameter)

Tujuan

Mengenal penggunaan method

Sebelum kita membahas method apa yang akan dipakai pada class, mari kita perhatikan penulisan method secara umum. Dalam pendeklarasian method, kita tuliskan :

<modifier> <returnType> <name>(<parameter>*) { <statement>* }

dimana,

<modifier> dapat menggunakan beberapa modifier yang berbeda

private : modifier yang menandakan variabel dan method hanya dapat diakses dari dalam kelas itu sendiri.

protected : modifier yang menandakan variabel dan method dapat diakses dari kelas itu sendiri dan dari kelas lain yang men-extends kelas tersebut.

public : modifier yang menandakan variabel dan method dapat diakses dari kelas manapun.

abstract : modifier yang menandakan bahwa methos merupakan jenis abstrak.

static : modifier yang menunjukkan bahwa method bersifat static, artinya perubahan nilai variabel pada suatu objek akan mempengaruhi nilai objek lainnya.

<returnType> dapat berupa seluruh tipe data, termasuk void

<name> nama methos

<parameter> ::= <tipe_parameter> <nama_parameter>[,]

Contoh method yang memiliki parameter :

```
public void cetakNama(String nama){
      System.out.println("Nama saya :"+nama);
}
```

Contoh method tanpa parameter :

```
public void cetakNama(){
      System.out.println("Nama saya Erlina");
}
```

Contoh method tanpa nilai balik

```
public void cetakNama(){
      System.out.println(“Nama Saya Erlina”);
}
```

Contoh method yang memberikan nilai balik

```
public String cetakNama(){
      return nama;
}
```

PRAKTIKUM

1. Tulis program berikut dan beri simpan sebagai file dengan nama mobil, kemudian kompile dan jalankan.'

```
Class Mobil{
      private String warna;
      int tahunProduksi;
      void isiWarna(String color){
            warna=color;
            System.out.println("Warna mobil  " +warna);
      }

       String tampilWarna(){
            return warna;
      }
}


public class ketiga
{
      public static void main (String[] args) {
            Mobil mobilku=new Mobil();
            mobilku.isiWarna(“Hijau”);
            Mobil mobilmu=new Mobil();
            mobilmu.isiWarna(“merah”);
            System.out.println(“mobilku  berwarna”
+mobilku.tampilWarna());
            System.out.println(“mobilnya
berwarna”+mobilmu.tampilWarna());
      }
}
```

TUGAS PRAKTIKUM

```
class handphone  {
      private String merk;
      private double seri;
      private double tahun;
      void isiMerk(String  merk1) {
            this.merk=merk1;
            SOP
      }

      void isiSeri(double  seri1, double seri2) {
            this.seri=seri1;
      }

      void isiSeri(string seri3) {
      ….................
      }
}

public class toko {
   public static void main(String[] args) {
   handphone nokia =new handphone();
   nokia.isiSeri(1112, 1113);
   nokia.isiSeri("seri3");
   }
}
```

Modifikasi program di atas :

1. Tambahkan metode untuk memasukan data seri dan tahun
2. Tambahkan metode untuk mengambil data merk,seri dan tahun
3. Buat objek untuk mengakses data handpone, masukan data misal merk=TOKIO, Seri=1111, tahun 2013
4. Tampilkan hasilnya

   Merk HP : Tokio

   Seri : 1111

   Tahun : 2013

5. Buat suatu metode yang ‘seolah-olah ‘ dapat mengirimkan data SMS
6. Buat suatu metode yang ‘seolah-olah ‘ dapat menampilkan isi SMS
7. Bagaimana kalau diinginkan menciptakan satu objek lagi misal SAMBUNG (dari class handphone)
8. Buat class PC, dengan atribut harddisk, ram, prosesor,harga,merk dengan metode isiData, ambilData
9. Dalam class tersebut terdapat metode :

a. Meminta isi tentang informasi PC tersebut
b. Menampilkan isi tentang informasi PC tersebut

10. Buat objek misal PENTIUM dan objek tersebut menjalankan semua metode yang ada PC

MODUL ke-5

Memahami Perulangan dan Keputusan

Tujuan :

1. Mengenal bentuk-bentuk perintah perulangan
2. Mengenal bentuk-bentuk perintah keputusan

5.1 Mengenal Bentuk Perulangan

Dalam pemrograman java terdapat beberapa perintah untuk proses perulangan

5.1.1 *while (ungkapan)*

Pernyataan;

Keterangan :

- bagian pernyataan akan diekseskusi selama ungkapan dalam while bernilai benar.
- Pengujian terhadap ungkapan pada while dilakukan sebelum bagian pernyataan.
- Kemungkinan pernyataan pada while tidak dijalankan sama sekali, jika ketemu kondisi yang pertama kali bernilai salah.

5.1.2 *for (ungkapan1;ungkapan2;ungkapan3)*
Pernyataan;

Keterangan :

- ungkapan1 merupakan pernyataan inisialisasi
- ungkapan2 sebagai kondisi yang menentukan pengulangan terhadap pernyataan atau tidak
- ungkapan3 digunakan sebagai pengatur variabel yang digunakan didalam ungkapan1

PRAKTIKUM

1. Tulis program berikut dan beri simpan sebagai file dengan nama "DemoFor", kemudian kompile dan jalankan.

```
public class DemoFor {
      public static void main(String[] args) {
            System.out.println("\nPROGRAM DEMO FOR");
            System.out.println("----------------\n");
      for ( int counter=0; counter<=5; counter++ )
         System.out.println( "Mencetak counter ke-" + counter );
   }
}
```

2. Modifikasi program di atas, sesuai intruksi di bawah ini

   Mencetak counter dari 10 sampai 20

   Mencetak counter dari 1 sampai 20, tetapi dengan hasil yang dicetak hanya bilangan genap saja

   Mencetak counter dari 20 mundur ke 1

Bagaimana programnya ??

3. Mengenal Bentuk Perulangan dengan while

```
public class DemoWhile {
      public static void main(String[] args) {
      int counter = 0;  // Inisialisasi counter
            System.out.println("\nPROGRAM DEMO WHILE");
            System.out.println("------------------\n");
      while ( counter <= 5 ) {  // Kondisi perulangan
         System.out.println( "Mencetak counter ke-" + counter );
         cunter=counter+1;   // Menaikkan counter dengan 1
      }
   }
}
```

4. Modifikasi program di atas, sesuai intruksi di bawah ini
   1. Mencetak counter dari 10 sampai 20
   2. Mencetak counter dari 1 sampai 20, tetapi dengan hasil yang dicetak hanya bilangan genap saja
   3. Mencetak counter dari 20 mundur ke 1

   Bagaimana programnya ??

5.2 Mengenal bentuk keputusan

5.2.1 <<IF Statement>>

Dalam JAVA, syntaks keputusan (IF) didefinisikan sebagai berikut :

```
if(pernyataan kondisional){
      //eksekusi kode program
}
```

Jika terdapat pernyataan kondisional dalam jumlah lebih dari dua, maka syntaks sebagai berikut :

```
if(pernyataan kondisional_1){
   //eksekusi kode program 1
}else if (pernyataan kondisional_2){
   //eksekusi kode program_2
} else{
   //eksekusi kode program lainnya
}
```

Contoh :

```
if(nilai>80){
      System.out.println("nilai A");
else if (nilai>70 && nilai<80){
      System.out.println ("nilai B");
}else{
      System.out.println("nilai C");
}
```

PRAKTIKUM – IF STATEMENT

1. Buat kelas "Mahasiswa"

```
public class Mahasiswa(){
            String nama;
            double nilai;

            public void setNama(String nama){
                  this.nama=nama
            }

            public void setNilai(double nilai){
                  this.nilai=nilai
            }

            public String getNama(){
                  return this.nama;
            }

            public double getNilai(){
                  return this nilai;
            }
      }
```

2. Buat kelas “Penilaian”

```
public class Penilaian {
      public static void main(String[]args){
            Mahasiswa mhs=new Mahasiswa();
            mhs.setNilai=88.8;
            mhs.setNama=”Erlina”;

            if(mhs.getNilai>80 && mhs.getNilai<=100){
                  System.out.println(“Nilai = A”);
            }else if (mhs.getNilai>70 && mhs.getNilai<=80){
                  System.out.println(“Nilai=B”);
            }else if (mhs.getNilai>50 && mhs.getNilai<=70){
                  System.out.println(“Nilai = C”);
            }else System.out.println(“Nilai=D”);
      }
}
```

5.2.2 <<**Switch & Case Statement**>>l

Dalam JAVA, syntaks switch & case didefinisikan sebagai berikut :

```
switch (switch expression){
            case case_selector_1 :{
                  //eksekusi program jika kondisi 1 terpenuhi
                  break;
            }
            case case_selector_2 :{
                  //eksekusi program jika kondisi 2 terpenuhi
                  break;
                  }
             default :{
                   //eksekusi program jika tidak ada satupun kondisi
                   //yang terpenuhi.
                   break;
            }
      }
```

Contoh :

```
switch (grade){
      case 100 :{
            System.out.println(“Luar biasa”);
      break;
      }
      case 90 :{ System.out.println(“Biasa”);
      break;
      }
      case 70 :{
            System.out.println(“Sedang”);
      break;
      }
      default :{
            System.out.println(“Kurang”);
      break;

      }
}
```

PRAKTIKUM – SWITCH CASE

1. Buat kelas “Mahasiswa”

```
public class Mahasiswa(){
String nama; double nilai; String jurusanpublic void setNama(String
nama){
                this.nama=nama
            }

            public void setNilai(double nilai){
                this.nilai=nilai
            }

            public String getNama(){
                return this.nama;
            }

            public double getNilai(){
                return this nilai;
            }
      }
```

2. Buat Kelas “Penilaian”

```
public class Penilaian {
      public static void main(String[]args){
            Mahasiswa mhs=new Mahasiswa();
            mhs.setJurusan(“informatika”)
            mhs.setNilai(88.8);
            mhs.setNama=(”Erlina”);

            System.out.println(“Nama : ”+mhs.getNama());
            System.out.println(“Nilai :”+mhs.getNilai());
            switch (mhs.getJurusan()){
                  case 01:{
                        System.out.println(“Jurusan : Teknik
                        kimia”);
                  }
                  case 02:{
                        System.out.println(“Jurusan : Teknik
                        industri”);
                  }
                  case 03:{
                        System.out.println(“Jurusan : Teknik
                        mesin”);
                  }
                  case 04:{
                        System.out.println(“Jurusan : Teknik
                        elektro”);
                  }
                  default :{
                        System.out.println(“Jurusan : Teknik
                        informatika”);
                  }
            }
      }
}
```

TUGAS PRAKTIKUM

1. Suatu pasar swalayan memberikan diskon dengan ketentuan :

   Bila besar belanja lebih 100.000,00 diskon 10%

   Bila besar belanja antara 50.000 s.d. 100.000 diskon 5%

   Bila belanja di bawah 50000 diskon 0 %

   Buat program untuk menentukan diskon

2. Buatlah program untuk suatu kondisi sebagai berikut :

   Jika nilai = 81 - 90, nilai = A, keterangan = lulus

   Jika nilai = 61 – 80, nilai = B, keterangan = lulus

   Jika nilai = 41 – 60, nilai = C, keterangan = lulus

   Jika nilai = 21 – 40, nilai = D, keterangan = ulang

   Jika nilai = 5 – 20, nilai = E, keterangan = ulang

   Buat program untuk melakukan konversi nilai dari nilai angka ke nilai abjad

MODUL ke-6
Mengenal konstruktor dan overloading

Tujuan :

1. Mengenal metode konstruktor
2. Mengenal metode overloadning

6.1 Konstruktor

Konstruktor pada Java merupakan method khusus yang dipakai oleh Java untuk membuat sebuah object didalam kelas dan tiap kelas boleh memiliki lebih dari satu konstruktor. Karakteristik konstruktor :

1. Nama Konstruktor = Nama Kelas
2. Tidak mengembalikan nilai termasuk void.
3. Cara menggunakan konstruktor adalah dengan menggunakan kata kunci*new*.

6.2 Overloading

Overloading di Java diterapkan di dalam method/fungsi. Dengan overloading dapat dibuat method dengan nama yang sama tetapi berbeda parameternya. Java sendiri akan menggunakan daftar parameter-parameter itu sebagai acuan untuk method manakah yang akan dijalankan.

PRAKTIKUM

1. Tulis program berikut dan beri simpan sebagai file dengan nama "PasanganDadu" , kemudian kompile dan jalankan.

```
class PasanganDadu {
    private int dadu1;
    private int dadu2;
      PasanganDadu() { //jadikan konstruktor, gimana caranya?
        // Kocok dadu dengan menggunakan bilangan acak antara 1 dan 6
        dadu1 = (int)(Math.random()*6) + 1;
        dadu2 = (int)(Math.random()*6) + 1;
    }
    public void cetak(){
        System.out.println("isi dadu 1 = "+dadu1);
        System.out.println("isi dadu 2 = "+dadu2);
    }
} // akhir kelas PasanganDadu
public class latihan {
    public static void main(String[] args) {
        PasanganDadu dadu = new PasanganDadu();
//       dadu.kocok();
```

```
        dadu.cetak();
    }
}
```

2. Modifikasi program di atas, sesuai instruksi di bawah ini, apa hasilnya dan kenapa? Ganti metode

```
public void kocok() {
      // Kocok dadu dengan menggunakan bilangan acak antara 1 dan 6
      dadu1 = (int)(Math.random()*6) + 1;
      dadu2 = (int)(Math.random()*6) + 1;
  }
```

Menjadi metode konstruktor dan bagaimana perubahan yang harus dilakukan pada class utama.

3. Tulis program berikut dan beri simpan sebagai file dengan nama …….. , kemudian kompile dan jalanka**n.**

```
class Manusia{
  String nama;
  String jenkel;
  void setNilai(String param1){
   nama = param1;
  }
  void setNilai(String param1,String param2){
   nama = param1;
   jenkel = param2;
  }
  void cetak(){
   System.out.println(nama+" adalah "+jenkel);
  }
}

class DemoManusia{
 public static void main(String args[]){
  Manusia m1,m2;
  m1 = new Manusia();
  m2 = new Manusia();
  m1.setNilai("Hendro");
  m2.setNilai("Hendro","Laki-laki");
  m1.cetak();
  m2.cetak();
  }
}
```

Dari program di atas:

1. Mana yang termasuk overloading
2. Tambahkan satu objek, misal SANTI, jenis kelamin Perempuan

TUGAS PRAKTIKUM

1. Buat program dengan menggunakan konstruktor, dengan ketentuan

   Terdapat atribut (private) yang terdiri dari :

   merk

   tahun

   noPolisi

   warna

   Buat metode konstruktor untuk memasukan data-data atribut tersebut

   Buat metode yang dapat menampilkan informasi data di atas

   Buat objek yang mengakses data-data di atas.

2. Buat program dengan menggunakan overloading dengan ketentuan :

   Terdapat atribut (private) yang terdiri dari :

   nomhs , nama, umur, Jurusan, alamat , Kota

   Buat metode yang memasukan data-data di atas, menggunakan Joption

   Kemudian buat metode overloading yang bukan menjadi konstruktor yang dapat menampilkan

   Hanya menampilkan nomhs, nama saja

   Hanya menampilkan nama, alamat dan kota saja

   Hanya menampilkan nomhs,nama,umur,alamat dan kota

MODUL ke-7
Mengenal Pewarisan & Package

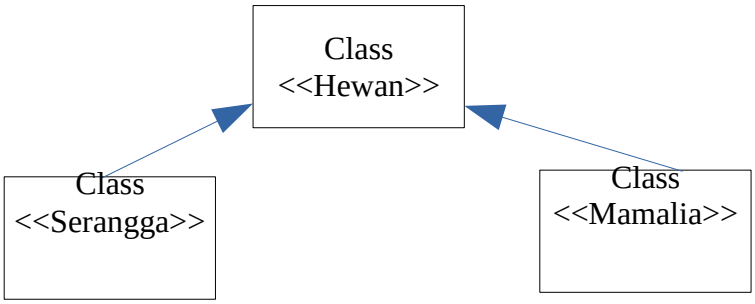
Tujuan :

1. Mahasiswa dapat dan mampu memahami pewarisan
2. Mahasiswa dapat dan mampu memahami penggunaan package

7.1. Pewarisan

Pewarisan di Java hanya mengenal pewarisan tunggal, artinya sebuah kelas hanya mewarisi atribut dan method dari satu kelas induk. Untuk menggunakan pewarisan di Java digunakan keyword extends.

Contoh pewarisan dapat dilihat pada diagram kelas berikut ini:



Gambar Contoh Pewarisan Kelas

Cara Pewarisan Kelas.

Kelas turunan secara prinsip dapat dibuat dengan menggunakan bentuk :

```
Class KelasTurunan extends KelasDasar{
    Tubuhkelas
}
```

PRAKTIKUM - PEWARISAN

1. Tulis program berikut dan beri simpan sebagai file dengan nama …….. , kemudian kompile dan jalankan**.**

```
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();
    }}
```

2. Modifikasi program di atas, sesuai instruksi di bawah ini, apa hasilnya dan kenapa

   Hilangkan kata extends Salam

   Tambahkan satu metode di class Salam

```
public void info3() {
    System.out.println("semangat pagi");
   }
```

   Kemudian buat bagaimana supaya objek obj bisa mengakses metode tersebut

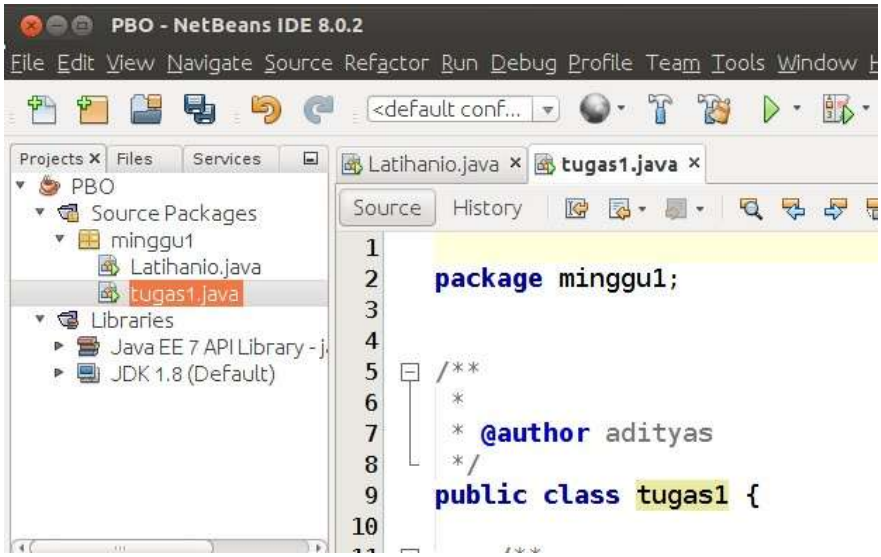
   Buat objek baru misal coba yang menunjuk ke class Salam, kemudian tambahkan di class utama objek coba mengakses info2, apahasilnya dan kenapa

   Bisakah objek coba mengakses metode info2, kenapa

7.2. Package

Dalam JAVA, package dapat diartikan sebagai folder. Fungsi dari package ini ditujukan untuk mengelompokkan kelas-kelas. Penggunaan package memungkinkan kelas dengan nama yang sama dapat dibuat dengan peletakan kelas pada package yang berbeda.

Deklarasi package diletakkan pada source code paling atas.



Gambar 7.1 Penggunaan package dalam java

PRAKTIKUM – PACKAGE

1. Buat package pada netbeans dengan nama “package1” , pada netbeans silahkan ikuti langkah sebagai berikut :

   File – New File

   Kemudian pada kolom categories pilih “JAVA” dan pada kolom “File Types” pilih “Java package” kemudian klik “next”.

   Pada bagian Package name berikan nama “package1” kemudian klik “finish”

2. Buat package lagi dengan nama “package2” , langkah-langkah sama seperti nomor 1.
3. Buat kelas baru dengan nama “Mahasiswa”. Pada netbeans silahkan ikuti langkah sebagai berikut :

   File – New File

   Kemudian pada kolom categories pilih “JAVA” dan pada kolom “File Types” pilih “Java Class” kemudian klik “next”.

   Pada bagian “Class Name” berikan nama “Mahasiswa” dan pada bagian “Package” pilih “package1”. Kemudian klik finish.

   Lengkapi kelas Mahasiswa dengan source code sebagai berikut :

```
public class Mahasiswa{
      private String nama;
      private int umur;
      public Mahasiswa(String nama, String umur){
            this.nama=nama;
            this.umur=umur;
      }

      public void setNama(String nama){
            this.nama=nama;
```

```
        }

        public void setUmur(int umur){
              this.umur=umur;
        }

        public String getNama(){
              return this.nama;
        }

        public int getUmur(){
              return this.umur;
        }
  }
```

4. Buat kelas dengan nama yang sama (Kelas Mahasiswa), lakukan sesuai dengan instruksi nomor 3, letakkan pada "package2". Lengkapi kelas Mahasiswa pada package2 dengan source code sebagai berikut :

```
public class Mahasiswa{

        private String jurusan;
        private String fakultas;
        public Mahasiswa(String jurusan, String fakultas){
              this.jurusan=jurusan;
              this.fakultas=fakultas;
        }

        public void setJurusan(String jurusan){
              this.jurusan=jurusan;
        }

        public void setFakultas(int fakultas){
              this.fakultas=fakultas;
        }

        public String getJurusan(){
              return this.jurusan;
        }

        public String getFakultas(){
              return this.fakultas;
        }
    }
```

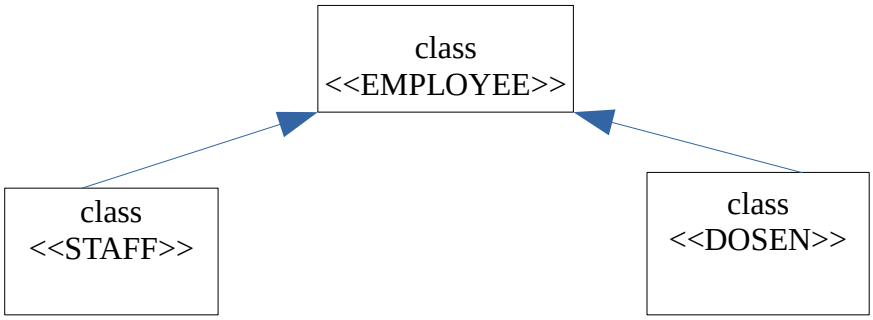
5. Buat main kelas dengan nama "LatihanPackage" dan letakkan pada package1. Lengkapi kelas LatihanPackage dengan source code sebagai berikut :

```
public class LatihanPackage{
     public static void main (String [] args){
           package1.Mahasiswa mhs1=new package1.Mahasiswa("Erlina",
23);
```

```
            package2.Mahasiswa mhs2=new
                        package2.Mahasiswa("Teknik
Informatika","Teknologi                          Industri");
            System.out.println("Nama saya "+mhs1.getNama()+" dan
                              saya"+mhs1.getUmur()+" tahun");
            System.out.println ("Saya kuliah di IST AKPRIND pada
jurusan                       "+mhs2.getJurusan()+"dan Fakultas
                                    "+mhs2.getFakultas());
    }
}
```

TUGAS PRAKTIKUM

Dari gambar di bawah ini



Class Employee merupakan class induk

- Dalam class ini terdapat beberapa metode
  - Metode setNama digunakan menset NAMA
  - Metode setNip digunakan menset NIP
  - Metode setGolongan digunakan menset GOLONGAN dan GAJI
  - Metode get ….. digunakan untuk menampilkan data ke layar

Class staff dan class dosen merupakan class anak

- Class Staff berisi metode Staff yang berisi misal staff di bagian mana
- Class dosen berisi metode Dosen berisi nama dosen dan seterusnya

Kembangan diagram di atas dengan konsep pewarisan dan ciptakan objek untuk melihat hasilnya

Kembangkan proses sesuai pemahaman kalian.

MODUL ke-8
Abstract class dan Interface

Tujuan

1. Mengenal penggunaan kelas abstrak
2. Mengenal penggunaan interface

8.1 Abstract Class

Abstract Class Dalam Java Abstract class dalam java digunakan untuk mendeklarasikan karakteristik umum dari subclass. Abstract class tidak bisa diinstansiasi. Abstract class hanya bisa digunakan sebagai super class, tapi juga bisa diturunkan dari class abstract lainnya. Untuk mendeklarasikan sebuah abstract class digunakan keyword abstract, contoh:

```
abstract class karyawan{
}
```

Sebuah abstract class pada dasarnya tidak jauh beda dengan class lainnya, yakni juga berisi method yang menggambarkan carakteristik dari kelas abstract tersebut, bedanya yakni sebuah abstract class bisa berisi method tanpa diimplementasikan artinya sebuah method tanpa body, method seperti ini disebut method abstract. untuk pendeklarasiannya digunakan keyword abstract:

```
abstract String getNama(){
}
```

Sifat kelas abstract :

1. Jika suatu kelas menggunakan modifier “abstract” , maka kelas tersebut beserta method didalamnya diaggap sebagai kelas abstract.
2. Suatu abstract class merupakan kelas yang berisi method tanpa body method dan variabel.
3. Jika suatu kelas tidak menggunakan modifier “abstract” , akan tetapi didalam kelas tersebut terdapat method dengan modifier abstract, maka secara otomatis kelas tersebut dianggap sebagai kelas abstract.
4. Kelas apapun yang meng-extends/mewarisi kelas abstract maka kelas apapun tersebut harus mengimplementasikan method dari kelas abstract, jika tidak dilakukan maka akan terjadi compile error.

5. Kelas apapun yang menggunakan modifier abstract, kemudian meng-extends/mewarisi kelas abstract, maka kelas apapun tersebut tidak perlu mengimplementasikan method dalam kelas abstract tersebut.
6. Suatu kelas hanya boleh meng-extends satu kelas abstract.

PRAKTIKUM-ABSTRACT CLASS

1. Buat kelas abstract, dengan nama 'kendaraan' :

```
abstract class kendaraan{
      public void jalankan();
}
```

2. Buat kelas yang "TestAbstract' meng-ekstends kelas abstract 'kendaraan'

```
class sedan extends kendaraan{
      public sedan(String nama){
            this.nama=nama;
      }
      public void jalankan(){
            System.out.println("putar  kunci  sampai  mobil  menyala"
+this.nama + " dan kemudikan....");
      }
}

class Minibus extends kendaraan{
      public Minibus(String nama){
            this.nama=nama;
      }
      public void jalankan(){
            System.out.println("Duduklah didepan setir " + this.nama +
" dan hidupkan mesin...");
      }
}

public class TesAbstrak{
      public static void main( String[]args){
            sepeda sepedaku =new sepeda("sepeda ontel");
            sepedaku.jalankan();
            Minibus mobilku =new Minibus("Minibus keri");
            mobilku.jalankan();
            }
}
```

8.2 Interface

Interface adalah jenis khusus dari blok yang hanya berisi method signature (atau constant). Interface mendefinisikan sebuah(signature) dari sebuah kumpulan method tanpa tubuh. Interface mendefinisikan sebuah cara standar dan umum dalam menetapkan sifat-sifat dari class-class. Mereka menyediakan class-class, tanpa memperhatikan lokasinya dalam hirarki class, untuk mengimplementasikan sifat-sifat yang umum. Dengan catatan bahwa interface-interface juga menunjukkan polimorfisme, dikarenakan program dapat memanggil method interface dan versi yang tepat dari method yang akan dieksekusi tergantung dari tipe object yang melewati pemanggil method interface.

Sifat interface :

1. Kelas apapun yang meng-implements/mengimplementasi interface maka kelas apapun tersebut harus mengimplementasikan method pada interface, jika tidak dilakukan maka akan terjadi compile error.
2. Suatu interface hanya berisi method tanpa body & konstatnta.
3. Suatu kelas boleh meng-implements lebih dari satu interface.

Cara pendeklarasian interface :

```
public interface modelGerakDarat{
      public void maju();
      public void mundur();
      public void belokKiri();
      public void belokKanan();
}

public interface modelGerakUdara{
      public void terbang();
      public void mendarat();
      public void manuver();
}
```

PRAKTIKUM – INTERFACE

1. Buat abstract kelas dengan nama “kendaraan” :

```
abstract class kendaraan{
        public void jalankan();
}
```

2. Buat Interface “ModelGerakDarat” , interface “ModelGerakLaut” dan interface “Lighting”

```
public interface ModelGerakDarat{
      public void maju();
```

```
        public void mundur();
        public void belokKiri();
        public void belokKanan();
 }

public interface ModelGerakUdara{
            public void terbang();
            public void mendarat();
            public void manuver();
}
public interface Lighting{
            public void lampuMenyala();
            public void lampuMati();
}
```

3. Buat kelas “Sedan ” yang meng-extends kelas abstrak “kendaraan” dan mengimplementasi interface “ModelGerakDarat” dan interface “Lighting ”

```
public class Sedan extends Kendaraan implements ModelGerakDarat,
Lighting{
String nama;
public Sedan(String nama){
            this.nama=nama;
}

      public void maju(){
            System.out.println(“sedan bergerak maju”);
      }

      public void mundur(){
            System.out.println(“sedan bergerak mundur”);
      }

      public void belokKiri(){
            System.out.println(“sedan bergerak belok kiri”);
      }

      public void belokKanan(){
            System.out.println(“sedan bergerak belok kanan”);
      }

      public void lampuMenyala(){
            System.out.println(“lampu sedan berhasil dinyalakan”);
      }

      public void lampuMati(){
            System.out.println(“lampu sedan berhasil dimatikan”);
      }
}
```

4. Buat kelas “Sukhoi ” yang meng-extends kelas abstrak “kendaraan” dan mengimplementasi interface “ModelGerakUdara” dan interface “Lighting ”

```
public class sukhoi extends Kendaraan implements modelGerakUdara,
lighting{
String nama;
public Sukhoi(String nama){
            this.nama=nama;
}
```

```
        public void terbang(){
              System.out.println("pesawat sukhoi terbang");
        }

  public void mendarat(){
              System.out.println("pesawat sukhoi mendarat");
        }

        public void manuver(){
              System.out.println("pesawat sukhoi mendarat");
        }

        public void lampuMenyala(){
              System.out.println("lampu   pesawat   sukhoi   berhasil
  dinyalakan");
        }
  public void lampuMati(){
              System.out.println("lampu   pesawat   sukhoi   berhasil
  dimatikan");
        }
  }
```

5. Buat kelas untuk program utama dengan nama "ProgramUtama"

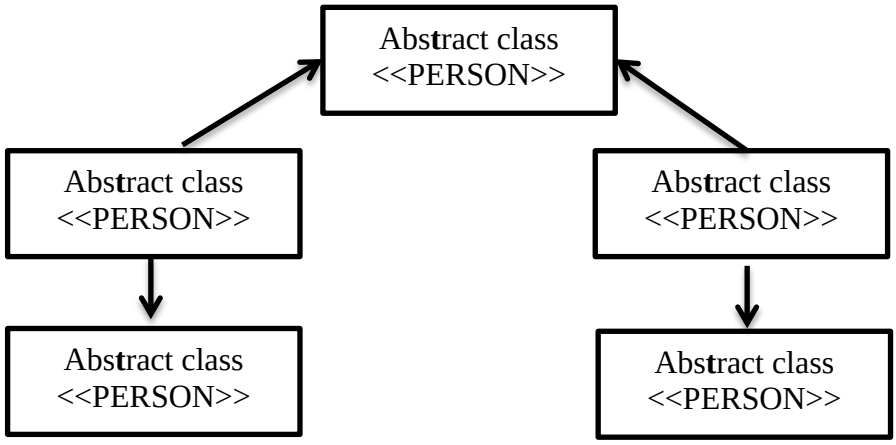
```
public class ProgramUtama {
      public static void main (String[]args){ Sedan
            mobilku=new Sedan("sedanku"); Sukhoi
            pesawatku=new Sukhoi("sukhoiku");
            mobilku.lampuMenyala();
            mobilku.maju();
            pesawatku.lampuMenyala();
            pesawatku.terbang();
      {
}
```

Abstract class Vs Interface

Abstract class	Interface
Satu kelas hanya bisa meng-ekstends satu kelas abstract	Satu kelas dapat meng-implements banyak interface
Digunakan untuk membentuk kelas	Digunakan untuk memberikan sifat pada kelas
Misalkan kita akan membuat class sukhoi, maka kita akan meng-ekstends class kendaraan.	Misalkan kita akan memberikan sifat pada class sukhoi, maka kita akan meng-implement interface ModelGerakUdara dan interface Lighting
Contoh: class sukhoi extends kendaraan	Contoh: Class sukhoi implements ModelGerakUdara, Lighting

TUGAS PRAKTIKUM

1. Diberikan diagram sebagai berikut :



2. Kelas “PERSON” adalah kelas abstrak yang memiliki variabel/atribut & metode sebagai berikut :

   variabel nama, alamat, telepon

   method getNama, getAlamat, getTelepon

3. Kelas “MAHASISWA” adalah kelas nyata yang meng-extend/mewarisi kelas “PERSON” . Kelas “MAHASISWA” memiliki variabel dan method sebagai berikut :

   variabel nim, jurusan, fakultas, semester, krs,ipk

   method getNim, getJurusan, getFakultas, getSemester, getKrs, getIpk

4. Kelas “DOSEN” adalah kelas nyata yang meng-extends/mewarisi kelas “PERSON”. Kelas “DOSEN” memiliki variabel danmethod sebagai berikut :

   variabel nip, mataKuliah, jurusan, fakultas

   method getNim, getJurusan, getFakultas

5. Interface “if_MAHASISWA” memiliki method sebagai berikut :

   method hitungKrs()
   method tampilKrs() contoh : (“Mahasiswa dapat mengambil 24 sks”);

6. Interface “if_DOSEN” memiliki method sebagai berikut :

   method mengajar() , method membimbingSkripsi()

7. Ketentuan Krs (> 3.00 = 24, 2.75 s/d 3.00 = 22, < 2.75 = 20)

7. Berdasarkan hirarki kelas yang sudah dicontohkan pada praktikum sebelumnya, buatlah kelas, pewarisan dan implementasi interface sesuai dengan ketentuan pada poin nomor 1-6 :)

ISBN 978-602-7619-43-2