Praktikum 4

Konsep Inheritance, Polymorphism, dan Encapsulation

Dosen : Ir. Nanang Syahroni M.Kom

Pokok Bahasan

- Konsep pewarisan dan deklarasi pewarisan dalam bahasa Java
- Konsep polimospisme dan deklarasi polimorpisme dalam bahasa Java
- Konsep enkapsulasi dan deklarasi enkapsulasi dalam bahasa Java

Tujuan Belajar

- Mengenalkan tentang konsep paket, class, dan konstruktor pada bahasa pemrograman java
- Mengenalkan tentang konsep pemprograman berorientasi obyek dengan cara mempraktekkan konsep pewarisan, polimorpisme dan enkapsulasi.

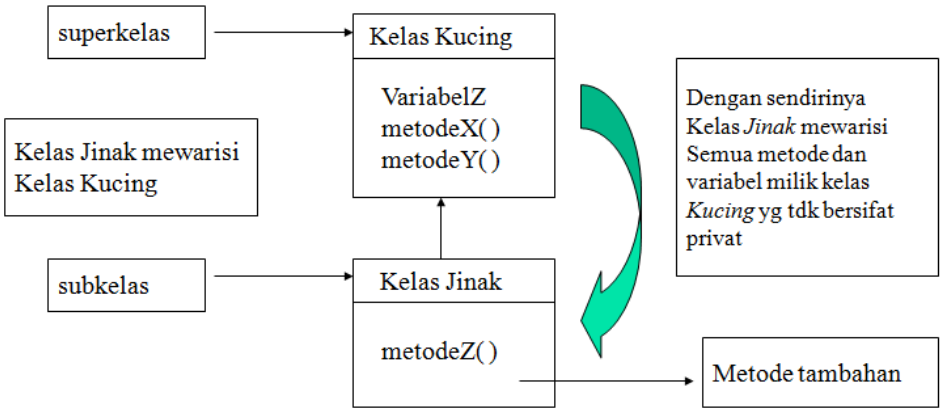
Inheritance

- Inheritance (Pewarisan) merupakan salah satu dari tiga konsep dasar OOP. Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan.
- Dengan konsep inheritance, memungkinkan untuk membuat suatu kelas dengan didasarkan pada kelas yang sudah ada sehingga mewarisi semua metode dan variabelnya
- Suatu class yang mempunyai class turunan dinamakan parent class atau base class. Sedangkan class turunan itu sendiri seringkali disebut subclass atau child class. Suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class.
- Karena suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class-nya,maka member dari suatu subclass adalah terdiri dari apa-apa yang ia punyai dan juga apa-apa yang ia warisi dari class parent-nya. Sehingga boleh dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parent class-nya.

- Dengan menambahkan kata kunci extends setelah deklarasi nama class, kemudian diikuti dengan nama parent class-nya. Kata kunci extends tersebut memberitahu kompiler Java bahwa kita ingin melakukan perluasan class.
- Biasanya kita baru perlu menerapkan inheritance pada saat kita jumpai ada suatu class yang dapat diperluas dari class lain.
- Semua class di dalam Java adalah merupakan subclass dari class super induk yang bernama Object. Pada saat dikompilasi, Kompiler Java akan membacanya sebagai subclass dari class Object.

Tabel 1: Kontrol Pengaksesan

Modifier	class yang sama	package yang sama	subclass package lain	class manapun
private	√			
default	√	√		
protected	√	√	√	
public	√	√	√	√



Gambar 1: Konsep Pewarisan

Praktek 1:

Buatlah menggunakan Inheritance seperti yang telah dibahas didalam kuliah. Kelas kucing adalah superclass dari kelas Jinak. Kelas Jinak berperan sebagai subclass. Untuk melakukan proses penurunan suatu kelas, diperlukan kata kunci extends, bentuk umum dari penggunaan kata kunci tersebut sebagai berikut:.

```
package turunan;

public class Kucing {

    public int a=5;
    protected int b=7;

    public void info1()
    {
       System.out.println("a= "+a);
       System.out.println("b= "+b);
    }
}

class Jinak extends Kucing
{
   private int c=9;

   public void info2()
   {
      System.out.println("a= "+a);
      System.out.println("b= "+b);
      System.out.println("c= "+c);
   }
}

public class Turunan {
    public static void main(String args[]) {
       Kucing coba1=new Kucing();
       coba1.info1();

       Jinak coba2=new Jinak();
       coba2.info2();

    }
}
```

Polimorphism

- Polimorfisme bisa diartikan satu bentuk banyak aksi, sekilas mirip dengan inheritance tetapi dalam polimorfisme kita data memerintah sebuah objek

untuk melakukan tindakan yang secara prinsip sama tapi secara proses dan outputnya berbeda.

- Polimorfisme mengizinkan kelas induk untuk mendefinisikan sebuah method general (bersifat umum) untuk semua kelas turunanya, dan selanjutnya kelas-kelas turunan dapat memperbaharui implementasi dari method tersebut secara lebih spesifik sesuai dengan karakteristiknya masing-masing.

Praktek 2:

Buatlah program menggunakan Polimorphism seperti berikut ini.

```
package testkucing;

public class Kucing {
 public void aksi(){
  System.out.println("Kucing bisa memanjat...");
 }
}
class Macan extends Kucing{
 public void aksi(){
  System.out.println("Macan Bisa Berenang...");
 }
}
class Singa extends Kucing {
   public void aksi(){
    System.out.println("Singa Hidup Berkelompok..");
 }
}
class Cheetah extends Kucing{
 public void aksi(){
  System.out.println("Cheetah bisa berlari cepat");
 }
}

public class TestKucing {
    public static void main(String[] args) {
    Kucing kc = new Kucing();
    kc.aksi();

    kc = new Macan();
    kc.aksi();

    kc = new Singa();
    kc.aksi();

    kc = new Cheetah();
    kc.aksi();
 }
}
```

Encapsulation

- Enkapsulasi adalah teknik pembuatan kelas pribadi (private) dan menyediakan akses melalui metode (public). Jika dinyatakan pribadi (public), ini tidak dapat diakses oleh siapa pun di luar kelas, sehingga bersembunyi bidang dalam kelas. Untuk alasan ini, enkapsulasi juga dirujuk sebagai data bersembunyi.
- Enkapsulasi dapat digambarkan sebagai penghalang pelindung yang mencegah kode dan data yang diakses secara acak oleh kode lainnya yang didefinisikan di luar kelas. Akses ke data dan kodenya dikontrol oleh sebuah interface.
- Manfaat utama dari enkapsulasi adalah kemampuan untuk mengubah kode kita dilaksanakan tanpa melanggar kode orang lain yang menggunakan kode kita. Dengan fitur ini enkapsulasi memberikan fleksibilitas dalam pengenmbangan bagi kode program kita.
- Untuk mengimplementasikan enkapsulasi, kita tidak menginginkan sembarang object dapat mengakses data kapan saja. Untuk itu, kita deklarasikan atribut dari class sebagai private. Namun, ada kalanya dimana kita menginginkan object lain untuk dapat mengakses data private. Dalam hal ini kita gunakan accessor methods. Accessor Methods digunakan untuk membaca nilai variabel pada class, baik berupa instance maupun static. Sebuah accessor method umumnya dimulai dengan penulisan *get<namaInstanceVariable>*. Method ini juga mempunyai sebuah return value.
- Jika kita menghendaki object lain untuk mengubah data, maka yang dapat kita lakukan adalah membuat method yang dapat memberi atau mengubah nilai variable dalam class, baik itu berupa instance maupun static. Method semacam ini disebut dengan mutator methods. Sebuah mutator method umumnya tertulis *set<namaInstanceVariabel>*.

Praktek 3:

Buatlah program menggunakan Encapsulation seperti berikut ini.

```
package latencapsulasi;

class Encapsulasi {

   private String name;
   private String idNum;
   private int age;

   public int getAge(){
      return age;
   }
   public String getName(){
      return name;
   }
   public String getIdNum(){
      return idNum;
   }

   public void setAge( int newAge){
      age = newAge;
   }
   public void setName(String newName){
      name = newName;
   }
   public void setIdNum( String newId){
      idNum = newId;
   }
}

public class LatEncapsulasi {
public static void main(String[] args) {
      Encapsulasi encap = new Encapsulasi();
      encap.setName("Iko Uwais");
      encap.setAge(25);
      encap.setIdNum("123456789");
      System.out.println("Nama : " + encap.getName()+
            ", Umur : "+ encap.getAge()+", ID : "+
encap.getIdNum());

    }
}
```

Tugas :

Buat flowchart dan program menggunakan array sesuai materi dalam kuliah:

a. Array satu dimensi untuk mencetak deret bilangan Fibonacci.
b. Array dua dimensi untuk mencetak bilangan Segitiga Pascal.