PRAKTIKUM 5

SIFAT PEWARISAN

A. TUJUAN PEMBELAJARAN

1. Memahami dan menerapkan konsep inheritance dalam pemrograman
2. Memahami pengaksesan member pada parent class
3. Memahami konsep single inheritance
4. Memahami konsep multi level inheritance
5. Melakukan pengontrolan akses pada pengkodean
6. Menggunakan kata kunci super
7. Menghindari kesalahan pada pewarisan konstruktor

B. DASAR TEORI

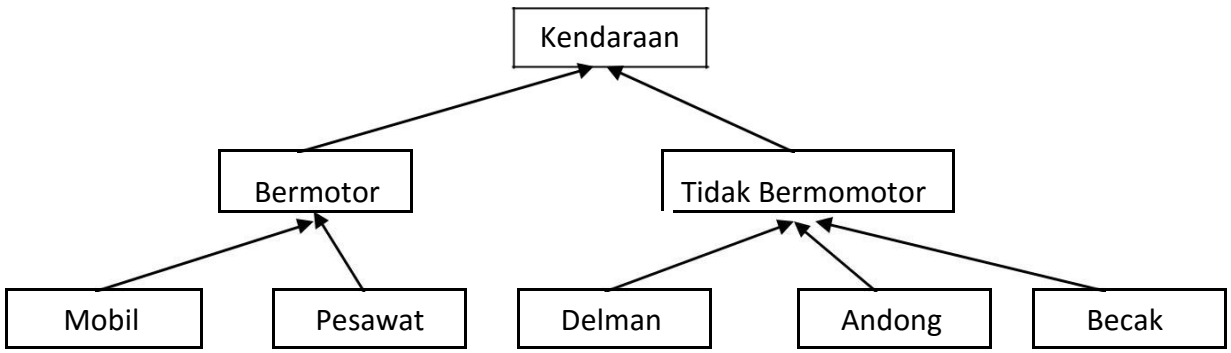
Konsep inheritance ini mengadopsi dunia riil dimana suatu entitas/obyek dapat mempunyai entitas/obyek turunan. Dengan konsep inheritance, sebuah class dapat mempunyai class turunan. Suatu class yang mempunyai class turunan dinamakan parent class atau base class. Sedangkan class turunan itu sendiri seringkali disebut subclass atau child class. Suatu subclass dapat mewarisi apa-apa yang dipunyai oleh parent class-nya, sehingga member dari suatu subclass adalah terdiri dari apa-apa yang ia punyai dan juga apa-apa yang ia warisi dari class parent-nya. Kesimpulannya, boleh dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parent class-nya.

Di dalam Java untuk mendeklarasikan suatu class sebagai subclass dilakukan dengan cara 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. Berikut adalah contoh deklarasi inheritance. Contoh:

```
public class B extends A {
      ...
}
```

Contoh diatas memberitahukan kompiler Java bahwa kita ingin meng-extend class A ke class B. Dengan kata lain, class B adalah subclass (class turunan) dari class A, sedangkan class A adalah parent class dari class B.

Java hanya memperkenankan adanya **single inheritance**. Konsep single inheritance hanya memperbolehkan suatu sublass mempunyai satu parent class. Dengan konsep single inheritance ini, masalah pewarisan akan dapat diamati dengan mudah. Konsep single inheritance bisa dilihat pada Gambar 5.1. Pada gambar tersebut bisa dilihat bahwa tiap class pasti memiliki hanya satu perant class. Java mengijinkan suatu sub class untuk memiliki anak lagi. Hal ini disebut dengan konsep multi level inheritance. Contohnya class Kendaraan memiliki class anak yaitu Bermotor, dan class Bermotor memiliki class anak lagi yaitu Mobil dan Pesawat.



Gambar 5.1 Gambaran konsep single dan multi level inheritance

Dalam konsep dasar inheritance dikatakan bahwa suatu subclass adalah tidak lain hanya memperluas (extend) parent class-nya. Contoh :

```
public class Pegawai {
        public String nama;
        public double gaji;
```

```
public class Manajer extends Pegawai {
        public String departemen;
}
```

Pada saat class Manajer menurunkan atau memperluas (extend) class Pegawai, maka ia mewarisi data member yang dipunyai oleh class Pegawai. Dengan demikian, class Manajer mempunyai data member yang diwarisi oleh Pegawai (nama, gaji), ditambah dengan data member yang ia punyai (departemen).

Pengaksesan member yang ada di parent class dari subclass-nya tidak jauh berbeda dengan pengaksesan member subclass itu sendiri. Contoh:

```
public class Manajer extends Pegawai {

      public String departemen;

      public void IsiData(String n, String d) {
            nama = n;
            departemen = d;
      }
}
```

Suatu parent class dapat tidak mewariskan sebagian member-nya kepada subclass-nya. Sejauh mana suatu member dapat diwariskan ke class lain, ataupun suatu member dapat diakses dari class lain, sangat berhubungan dengan access control (kontrol pengaksesan). Di dalam java, kontrol pengaksesan dapat digambarkan dalam Tabel 5.1.

Tabel 5.1. Akses modifier

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

Kata kunci *super* dipakai untuk merujuk pada member dari parent class, sebagaimana kata kunci *this* yang dipakai untuk merujuk pada member dari class itu sendiri. Adapun format penulisannya adalah sebagai berikut:

super.data_member → merujuk pada data member pada parent class

super.function_member(→ merujuk pada function member pada parent

class super() → merujuk pada konstruktor pada parent class

Dalam inheritance, member kelas yang diwariskan hanyalah variabel dan method saja. Sedangkan konstruktor parent class tidak diwariskan ke sub class. Tetapi ketika suatu obyek anak dibuat dalam artian ketika konstruktor anak dijalankan maka konstruktor parent class dijalankan terlebih dahulu dan selanjutnya menyelesaikan konstruktor anak.

C. TUGAS PENDAHULUAN

1. Apa yang dimaksud dengan inheritance?
2. Buatlah contoh kasus yang menerapkan konsep inheritance!
3. Adakah perbedaan cara mengakses member class milik parent dan member class milik sendiri? Jelaskan melalui contoh! (Silahkan memanfaatkan jawaban soal nomor 2.)
4. Apa yang dimaksud dengan konsep single inheritance?
5. Apa yang dimaksud dengan konsep multi level inheritance?
6. Ada berapa modifier untuk pengontrolan akses? Jelaskan masing-masing!
7. Apakah kegunaan kata kunci super? Jelaskan!
8. Apakah yang dimaksud dengan konstruktor tidak diwariskan?

D. PERCOBAAN

Percobaan 1 : Penyembunyian informasi

Buatlah class Pegawai seperti dibawah ini:

```
public class Pegawai {
    private String nama;
    public double gaji;
}
```

Kemudian buatlah class Manajer seperti ini dibawah ini.

```
public class Manajer extends Pegawai {

     public String departemen;

     public void IsiData(String n, String d) {
          nama=n;
          departemen=d;
       }
}
```

Sekarang cobalah untuk mengkompilasi class Manajer diatas. Apa yang terjadi?. Pesan kesalahan akan muncul seperti ini:

```
Manajer.java:5: nama has private access in Pegawai
             nama=n;
```

Ini membuktikan bahwa class Manajer tidak mewarisi data member nama dari parent class-nya (Pegawai).

Percobaan 2 : Menggunakan kata kunci super

Berikut ini listing penggunaan kata kunci super.

```
class Parent {
  public int x = 5;
}


class Child extends Parent {
  public int x = 10;
  public void Info(int x) {
     System.out.println("Nilai x sebagai parameter = " + x);
     System.out.println("Data member x di class Child = " + this.x);
     System.out.println("Data member x di class Parent = " + super.x);
  }
}
public class NilaiX {
  public static void main(String args[]) {
     Child tes = new Child();
     tes.Info(20);
  }
}
```

Ketika program tersebut dijalankan, akan tampak hasil seperti dibawah ini :

```
Nilai x sebagai parameter = 20
Data member x di class Child = 10
Data member x di class Parent = 5
```

Percobaan 3 : Konstruktor tidak diwariskan

Buatlah class kosong bernama Parent seperti dibawah:

```
public class Parent {


}
```

Buatlah class Child yang menurunkan class Parent seperti dibawah ini:

```
public class Child extends Parent{
      int x;
      public Child() {
          x    =    5;
          super();
     }
}
```

Lakukan kompilasi pada Child diatas. Apa yang terjadi?. Pasti disana terjadi error. Sekarang ubahlah sedikit class Child diatas seperti dibawah ini:

```
public class Child extends Parent {
      int x;
      public Child() {
          super();
          x = 5;
     }
}
```

Setelah dikompilasi, anda tidak mendapatkan error sebagaimana yang sebelumnya. Ini yang harus kita perhatikan bahwa untuk pemanggilan konstruktor parent class, kita harus melakukan pemanggilan tersebut di baris pertama pada konstruktor subclass.

E. LATIHAN

Latihan 1. Tempatkan class Base dan class Class1 di direktori yang sama. Apa yang terjadi ketika Class1.java dikompile dan dijalankan jika sebelumnya Base.java belum dikompile? Jelaskan !

```
//Base.java
package Base;
class Base{
    protected void amethod(){
        System.out.println("amethod");
    }//End of amethod

}//End of class base
```

```
package Class1;
//Class1.java
public class Class1 extends Base{
    public static void main(String
      argv[]){ Base b = new Base();
      b.amethod();
    }//End of main
}//End of Class1
```

Latihan 2. Aturan overriding

a. Berdasarkan kode di bawah ini, akses modifier (public, protected atau private) apa yang diijinkan di tambahkan sebelum myMethod() baris 3?

b. Jika baris 3 seperti kode di bawah (apa adanya tanpa perubahan) keywords apa yang diijinkan ditambahkan sebelum myMethod baris 8?

```
1.      class HumptyDumpty
2.      {
3.              void myMethod() {}
4.      }
5.
6.      class HankyPanky extends HumptyDumpty
7.      {
8.              void myMethod() {}
9.      }
```

Latihan 3.

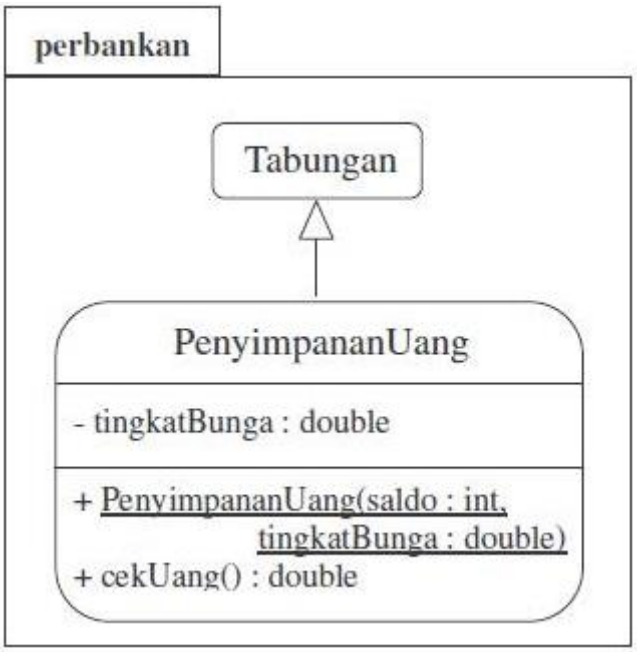
a. Apa yang terjadi bila <u>kedua kode dibawah</u> ini dikompile dan dijalankan dalam <u>satu direktori</u>? Jelaskan !

b. Bagaimana solusi supaya tidak terjadi error?

```
//File P1.java

package MyPackage;

class P1{
      void aFancyMethod(){
           System.out.println("What a fancy method");
      }
}
```

```
//File P2.java
public class P2 extends P1{
    public static void main(String argv[]){
        P2 p2 = new P2();
        p2.aFancyMethod();

     }
 }
```

Latihan 4. Mengimplementasikan UML class diagram dalam program untuk package perbankan



Ubahlah mode akses atribut saldo pada Tabungan menjadi protected. Lalu Transformasikan class diagram diatas ke dalam bentuk program! Tulislah listing program berikut ini sebagai pengetesan.

```
import perbankan.*;

public class TesLatihan{
    public static void main(String args[]){
        PenyimpananUang tabungan = new PenyimpananUang(5000,
        8.5/100); System.out.println("Uang yang ditabung : 5000");
        System.out.println("Tingkat bunga sekarang : 8.5%");
        System.out.println("Total uang anda sekarang : " +
                           tabungan.cekUang());
      }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti diatas.

```
Uang yang ditabung : 5000
Tingkat bunga sekarang : 8.5%
Total uang anda sekarang : 5425.0
```

Latihan 5. Konstruktor tidak diwariskan

Buatlah class berikut ini

```
class Base{
    Base(int i){
      System.out.println("base constructor");
    }
    Base(){
    }
}
```

```
public class Sup extends Base{
    public  static  void  main(String  argv[]){
      Sup s= new Sup();

      //baris 1
    }

    Sup(){
      // baris 2
    }

    public void derived(){
      // baris 3
    }
}
```

Modifikasilah class Sup (di bagian **//baris 1, //baris 2** *atau* **//baris 3)** sedemikian hingga konstruktor kelas **Base** (konstruktor **Base(int i)**) dipanggil dan menampilkan string "base constructor" ke layar!

Latihan 6. Konstruktor tidak diwariskan

```
private class Base{

    Base(){

      int i = 100; System.out.println(i);

    }

}
```

```
public class Pri extends Base{
    static int i = 200;
    public static void main(String argv[]){
        Pri p = new Pri();
        System.out.println(i);
    }
}
```

Kompile dan jalankan program diatas! Apa yang terjadi? Jelaskan !

Latihan 7. Apa yang tampil di layar bila kode dibawah ini dijalankan?

```
class X{
            Y b = new Y(); X(){
                System.out.print("X");
            }
}
```

```
class Y{
            Y(){
                System.out.print("Y");
            }
}
```

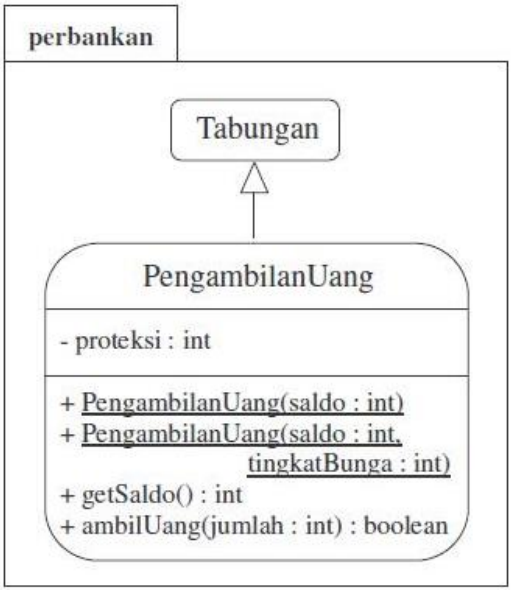
```
public class Z extends X{
            Y y = new Y();
            Z(){
                System.out.print("Z");
            }
            public static void main(String[] args){
                new Z();
            }
}
```

Latihan 8. Kompile dan jalankan program berikut! Apa yang terjadi? Jelaskan!

```
public class Hope{
    public static void main(String argv[]){
        Hope h = new Hope();
    }

    protected Hope(){
        for(int  i  =0;  i  <10;  i  ++){
            System.out.println(i);
        }
    }
}
```

F. TUGAS

Mengimplementasikan UML class diagram dalam program untuk package perbankan



Transformasikan class diagram diatas ke dalam bentuk program! Tulislah listing program berikut ini sebagai pengetesan.

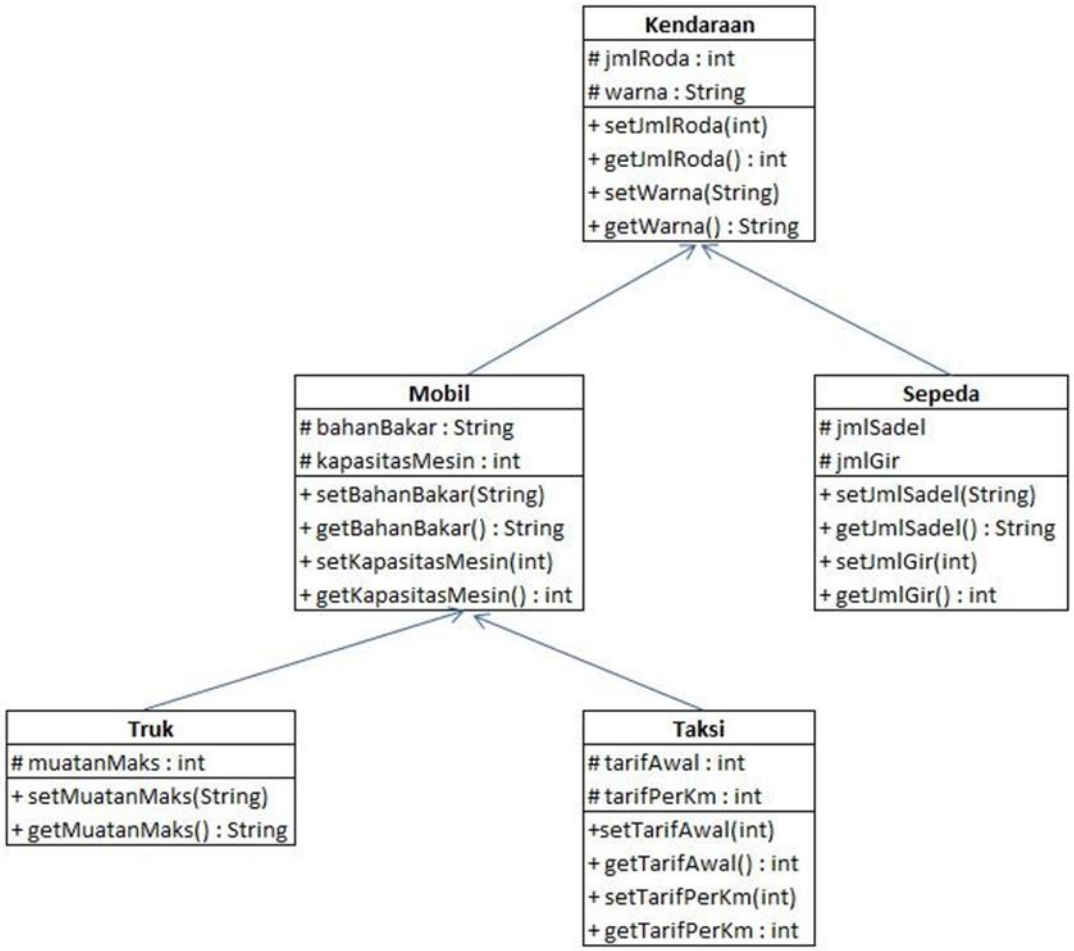
```
import perbankan.*;

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

        PengambilanUang tabungan = new PengambilanUang(5000,
        1000); System.out.println("Uang yang ditabung : 5000");
        System.out.println("Uang yang diproteksi : 1000");
        System.out.println("----------------------------");
        System.out.println("Uang yang akan diambil : 4500 " +
                           tabungan.ambilUang(4500));
        System.out.println("Saldo sekarang : " + tabungan.getSaldo());
        System.out.println("----------------------------");
        System.out.println("Uang yang akan diambil : 2500 " +
                           tabungan.ambilUang(2500));
        System.out.println("Saldo sekarang : " + tabungan.getSaldo());
    }
}
```

Lakukan kompilasi pada program diatas dan jalankan. Jika tampilan di layar tampak seperti dibawah ini, maka program anda sudah benar. Jika tidak sama, benahi kembali program anda dan lakukan hal yang sama seperti diatas.

```
Uang yang ditabung : 5000
Uang yang diproteksi : 1000
-----------------
Uang yang akan diambil : 4500
false Saldo sekarang : 5000
-----------------
Uang yang akan diambil : 2500

true Saldo sekarang : 2500
```

Mengimplementasikan UML class diagram dalam program.

a. Buatlah kelas-kelas berdasarkan UML class diagram dibawah ini!



b. Selanjutnya buatlah kelas Tes.java yang membuat obyek-obyek serta mengeset nilai variabel seperti pada Tabel 5.2.dan tampilkan data per obyek.

Tabel 5.2. Data obyek

Obyek	jmlRoda	warna	bahanBakar	kapasitasMesin	muatanMaks	
truk1	4	kuning	solar	1500	1000	
truk2	6	merah	solar	2000	5000	
					tarifAwal	tarifPerKm
taksi1	4	oranye	bensin	1500	10000	5000
taksi1	4	biru	bensin	1300	7000	3500
			jmlSadel	jmlGir		
sepeda1	3	hitam	1	2		
sepeda2	2	putih	2	5		

G. LAPORAN RESMI

Kumpulkan hasil latihan dan tugas di atas. Tambahkan analisa dalam laporan resmi.