PERTEMUAN 11
POLYMORPHISM

Rifki Dwi Kurniawan, S.Kom | 2023

PERTEMUAN 11

Overview

1. Implementasi Polimorphism
2. Overloading method VS Overriding Method
3. Ad Hoc & Subtyping Polimorphism

DEFINISI

- Berarti kemampuan objek untuk mempunyai banyak bentuk
- Suatu aksi yang memungkinkan programmer menyampaikan pesan tertentu keluar dari hierarki objeknya.
- Objek yang berbeda memberikan tanggapan/respon terhadap pesan yang sama sesuai dengan sifat masng-masing objek
- Proses polimorphism : overloading method & overriding method

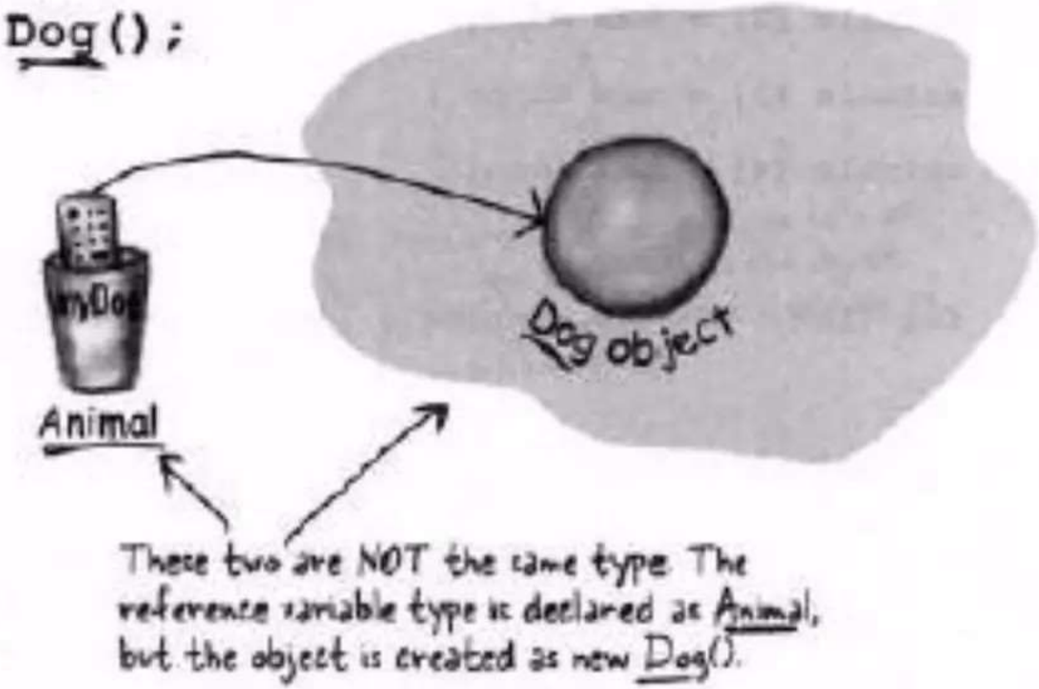
  Jenis polymorph : Ad hoc & Subtyping

Source : AdamMukharilBachtiar, M.T

OBJEK POLYMORPHISM (1)

with polymorphism, the reference and the object can be *different*.

```
Animal myDog = new Dog();
```



OBJEK POLYMORPHISM (2)

OK, OK maybe an example will help.

```
Animal[] animals = new Animal[5];
animals [0] = new Dog();
animals [1] = new Cat();
animals [2] = new Wolf();
animals [3] = new Hippo();
animals [4] = new Lion();

for (int i = 0; i < animals.length; i++) {

    animals[i].eat();

    animals[i].roam();
}
```

Declare an array of type Animal. In other words, an array that will hold objects of type Animal.

But look what you get to do... you can put ANY subclass of Animal in the Animal array!

And here's the best polymorphic part (the raison d'être for the whole example), you get to loop through the array and call one of the Animal-class methods, and every object does the right thing!

When 'i' is 0, a Dog is at index 0 in the array, so you get the Dog's eat() method. When 'i' is 1, you get the Cat's eat() method

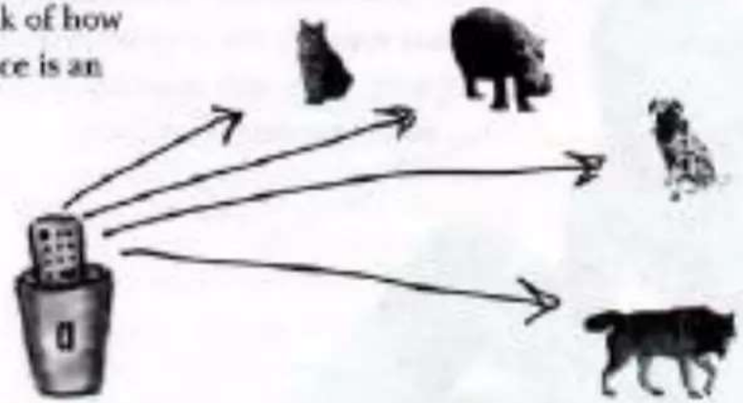
Same with roam().

OBJEK POLYMORPHISM (3)

But wait! There's more!

You can have polymorphic arguments and return types.

If you can declare a reference variable of a supertype, say, Animal, and assign a subclass object to it, say, Dog, think of how that might work when the reference is an argument to a method...



```
class Vet {

   public void giveShot(Animal a) {

      // do horrible things to the Animal at

      // the other end of the 'a' parameter

      a.makeNoise();
```

The Animal parameter can take ANY Animal type as the argument. And when the Vet is done giving the shot, it tells the Animal to makeNoise(), and whatever Animal is really out there on the heap, that's whose makeNoise() method will run.

OBJEK POLYMORPHISM (4)

```
class PetOwner {
    public void start() {
        Vet v = new Vet();
        Dog d = new Dog();
        Hippo h = new Hippo();
        v.giveShot(d);
        v.giveShot(h);
    }
}
```

The Vet's giveShot() method can take any Animal you give it. As long as the object you pass in as the argument is a subclass of Animal, it will work.

Dog's makeNoise() runs

Hippo's makeNoise() runs

OVERLOADING METHOD VS OVERRIDING METHOD

An overloaded method is just a different method that happens to have the same method name. It has nothing to do with inheritance and polymorphism. An overloaded method is NOT the same as an overridden method.

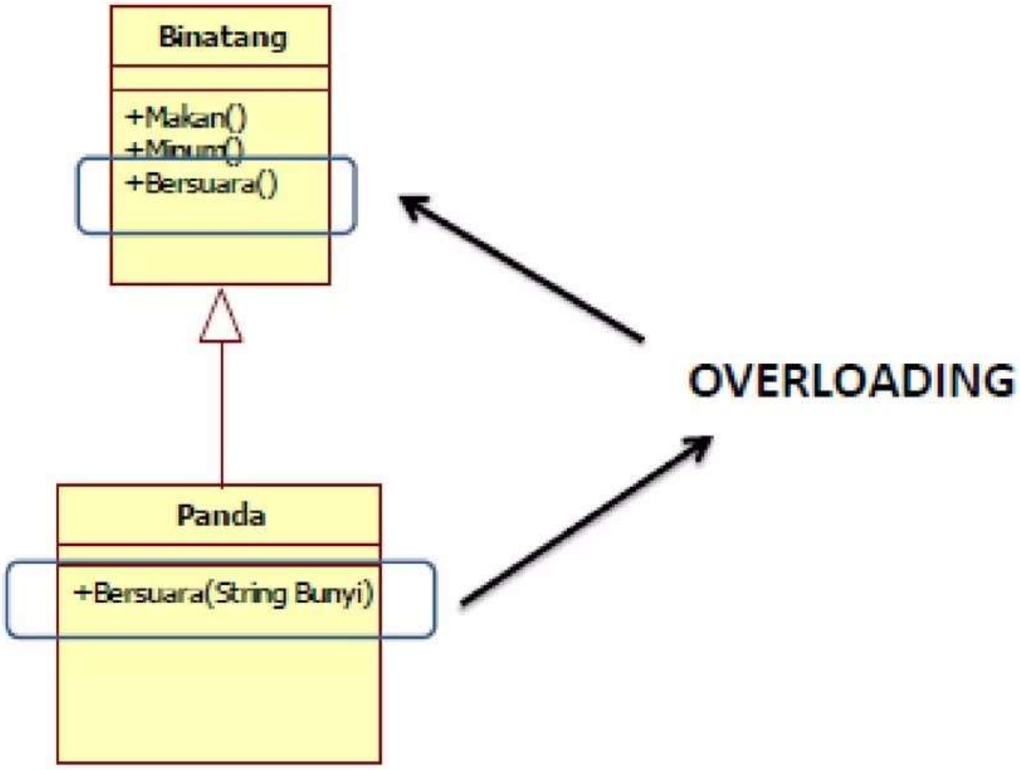
Sebagai contoh:

titik(x,y) → titik dipandang dua dimensi

titik(x,y,z) → titik dipandang tiga dimensi

DEFINISI OVERLOADING METHOD

- Adalah keadaan dimana beberapa method yang terdapat pada suatu class memiliki nama yang sama dengan fungsionalitas yang sama



Source : AdamMukharilBachtiar, M.T

ATURAN OVERLOADING METHOD

- **The return types can be different.**

  You're free to change the return types in overloaded methods, as long as the argument lists are different.

- **You can't change ONLY the return type.**

  If only the return type is different, it's not a valid *overload*—the compiler will assume you're trying to *override* the method. And even *that* won't be legal unless the return type is a subtype of the return type declared in the superclass. To overload a method, you MUST change the argument list, although you *can* change the return type to anything.

- **You *can* vary the access levels in any direction.**

  You're free to overload a method with a method that's more restrictive. It doesn't matter, since the new method isn't obligated to fulfill the contract of the overloaded method.

CONTOH OVERLOADING METHOD

```
public class OrangTua {
    private int umur;

    public int getUmur() {
        return umur;
    }

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

    public void tampilUmur(){
        System.out.println("Umur Orang Tua : "+umur);
    }
}
```

```
public class Anak extends OrangTua {
    private int umur2;

    public int getUmur2() {
        return umur2;
    }

    public void setUmur2(int umur2) {
        this.umur2 = umur2;
    }

    //overloading
    public void tampilUmur(int a){
        System.out.println("Umur Anak : "+umur2);
    }
}
```


Source : AdamMukharilBachtiar, M.T

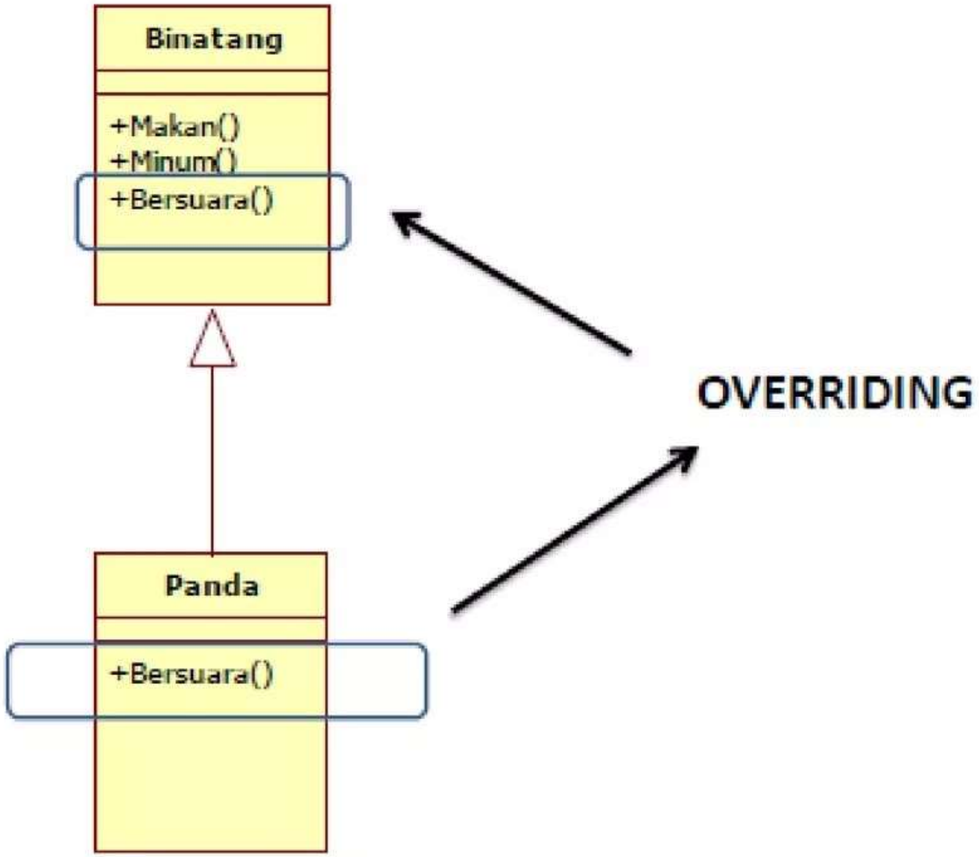
CONTOH OVERLOADING METHOD

```
public class TesterPolimorphisme {

    public static void main(String[] args) {
        Anak x = new Anak();
         x.setUmur(40);
         x.setUmur2(11);
         x.tampilUmur(1);//Pemanggilan Overloading
    }
}
```

Source : AdamMukharilBachtiar, M.T

DEFINISI OVERRIDING METHOD

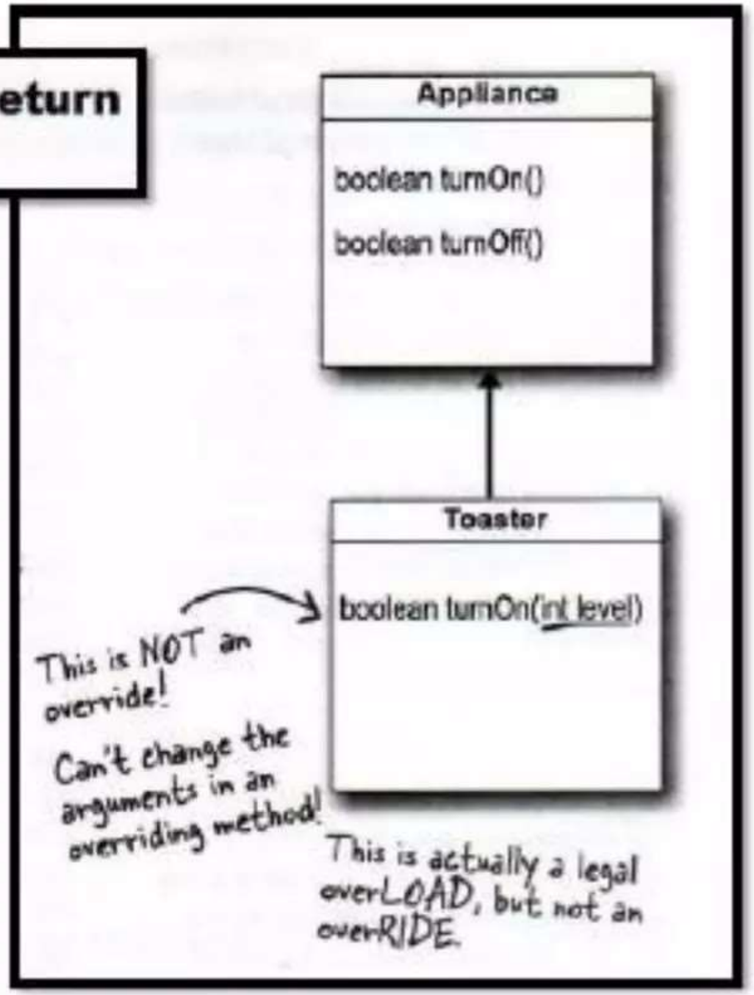
- Adalah keadaan dimana suatu method di subclass mengingkari method yang ada pada superclassnya.



Source : AdamMukharilBachtiar, M.T

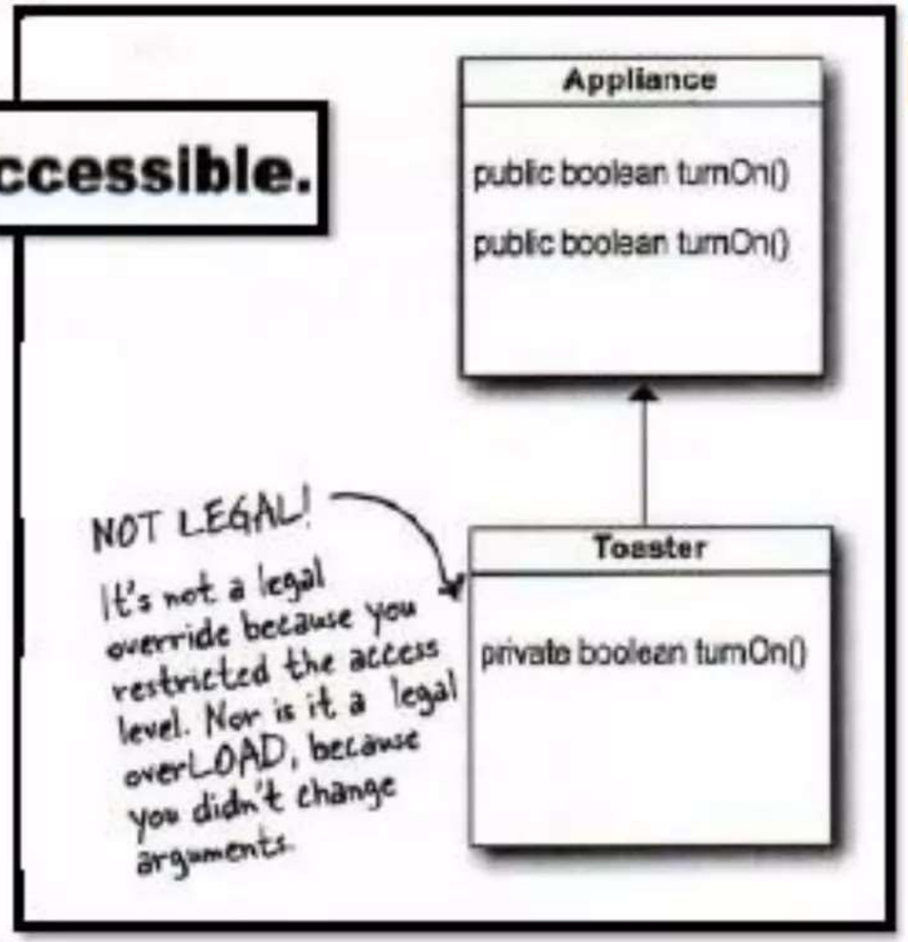
ATURAN OVERRIDING METHOD

1. **Arguments must be the same, and return types must be compatible.**



ATURAN OVERRIDING METHOD

② The method can't be less accessible.



CONTOH OVERRIDING METHOD

```
public class OrangTua {
    private int umur;

    public int getUmur() {
        return umur;
    }

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

    public void tampilUmur(){
        System.out.println("Umur Orang Tua : "+umur);
    }
}
```

```
public class Anak extends OrangTua {
    private int umur2;

    public int getUmur2() {
        return umur2;
    }

    public void setUmur2(int umur2) {
        this.umur2 = umur2;
    }

    @Override //Penanda Overriding di java
    public void tampilUmur(){
        super.tampilUmur();
        System.out.println("Umur Anak : "+umur2);
    }
}
```

Source : AdamMukharilBachtiar, M.T

CONTOH OVERRIDING METHOD

```
public class TesterPolimorphisme {

    public static void main(String[] args) {
        Anak x = new Anak();
         x.setUmur(40);
         x.setUmur2(11);
         x.tampilUmur();//Pemanggilan Overriding
    }
}
```

Source : AdamMukharilBachtiar, M.T

PERTEMUAN 11

JENIS POLYMORPHISM

Rifki Dwi Kurniawan, S.Kom | 2023

1. POLIMORPHISM AD HOC

Polymorph Ad Hoc terjadi ketika memiliki sebuah method yang akan mempunyai operasi yang berbeda ketika dipanggil dengan tipe parameter yang berbeda.

```
package pewarisan;

public class Polymorph {
    public int tambah(int x, int y){
                return x+y;
    }
    public String tambah(String x, String y){
        return x+" "+y;
    }
}
```

Class Polymorph mempunyai 2 buah method yang namanya sama yaitu "tambah", tetapi dibedakan dengan parameternya.

- Class PolymorphTester

```
package pewarisan;

public class PolymorphTester {

  public static void main(String[] args) {
        Polymorph p=new Polymorph();
        System.out.println("2 + 3 = "+p.tambah(2, 3));
        System.out.println("\"2\" + \"3\" = "+p.tambah("2", "3"));
  }
}
```

Hasil Run :

```
2 + 3 = 5
"2" + "3" = 2 3
```


Source : AndriHeryandi, M.T.

2. POLIMORPHISM SUBTYPING

Polymorph Subtyping terjadi sebuah referensi dari sebuah class diakses dengan menggunakan referensi subclassnya.

```
package pewarisan;

public class PolySubtype {
  public static void main(String[] args) {
        Binatang elang,kakaktua,kuda,sapi;
        elang=new Elang();elang.setNama("Elang");
        kakaktua=new Kakaktua();kakaktua.setNama("Kakaktua");
        kuda=new Kuda();kuda.setNama("Si Pony");
        sapi=new Sapi();sapi.setNama("Si Mooo");
        elang.bersuara();
        kakaktua.bersuara();
        kuda.bersuara();
        sapi.bersuara();
  }
```

Seluruh objek menggunakan superclass. Tetapi tetap akan mengeksekusi method masing-masing class.

Hasil Run :

```
Elang sedang bersuara.
Kakaktua sedang bersuara.
Si Pony sedang bersuara.
Si Mooo bersuara : Mooooooooo
```


Source : AndriHeryandi, M.T.

2. POLIMORPHISM SUBTYPING

Contoh Lain :

```
package pewarisan;
public class PolySubtype2 {
        static void bicara(Binatang b){
                b.bersuara();
        }
        public static void main(String[] args) {
                Elang e=new Elang();e.setNama("Hunter");
                Kakaktua k=new Kakaktua();k.setNama("Oces");
                Kuda kd=new Kuda();kd.setNama("Blacky");
                Sapi s=new Sapi();s.setNama("Moooo");
                bicara(e);
                bicara(k);
                bicara(kd);
                bicara(s);

        }
```

Method bicara membutuhkan parameter dengan class Binatang.

Pemanggilan method Bicara menggunakan parameter subclass dari Binatang.

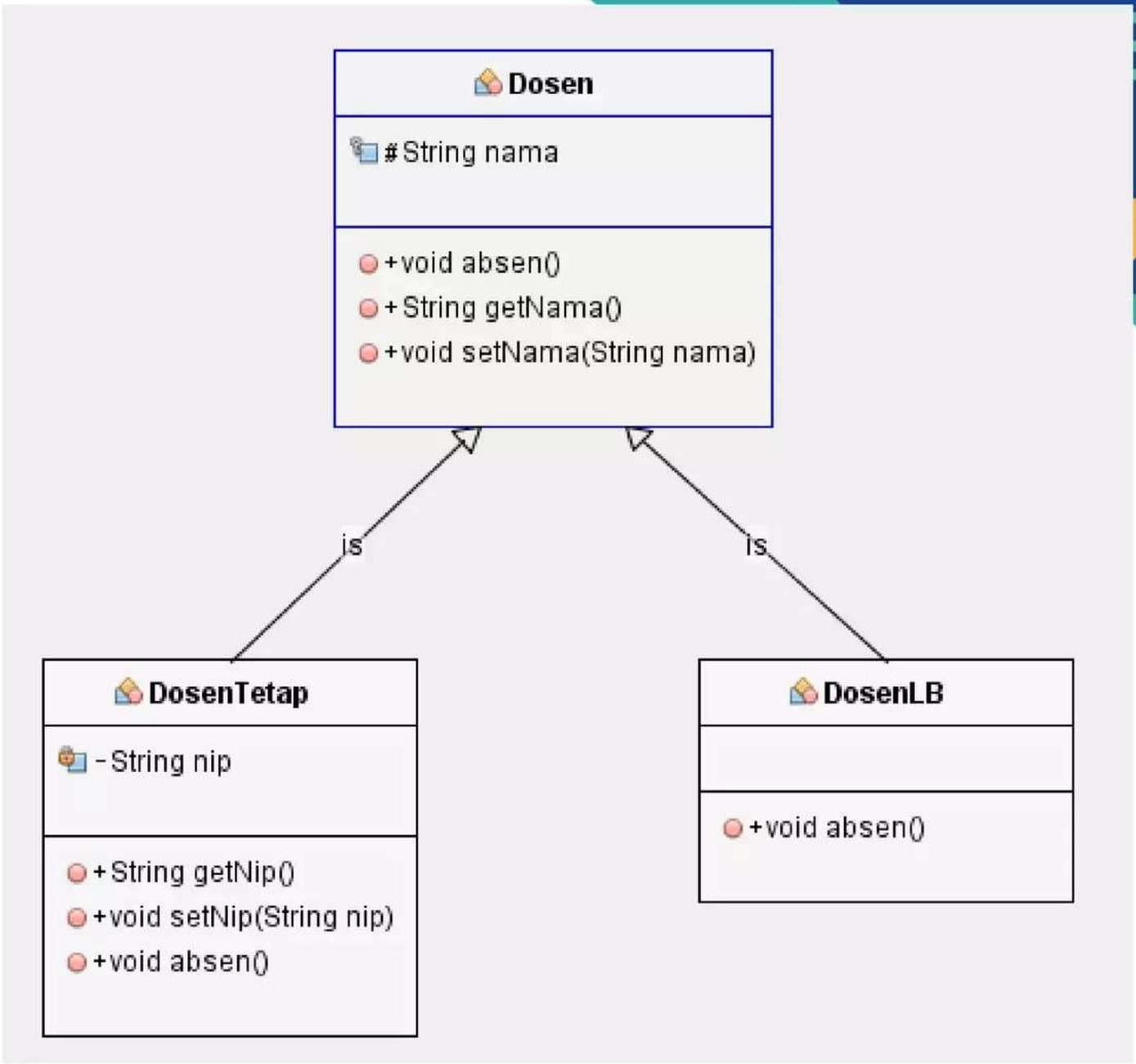
Hasil Run :

```
Hunter sedang bersuara.
Oces sedang bersuara.
Blacky sedang bersuara.
Moooo bersuara : Mooooooooo
```

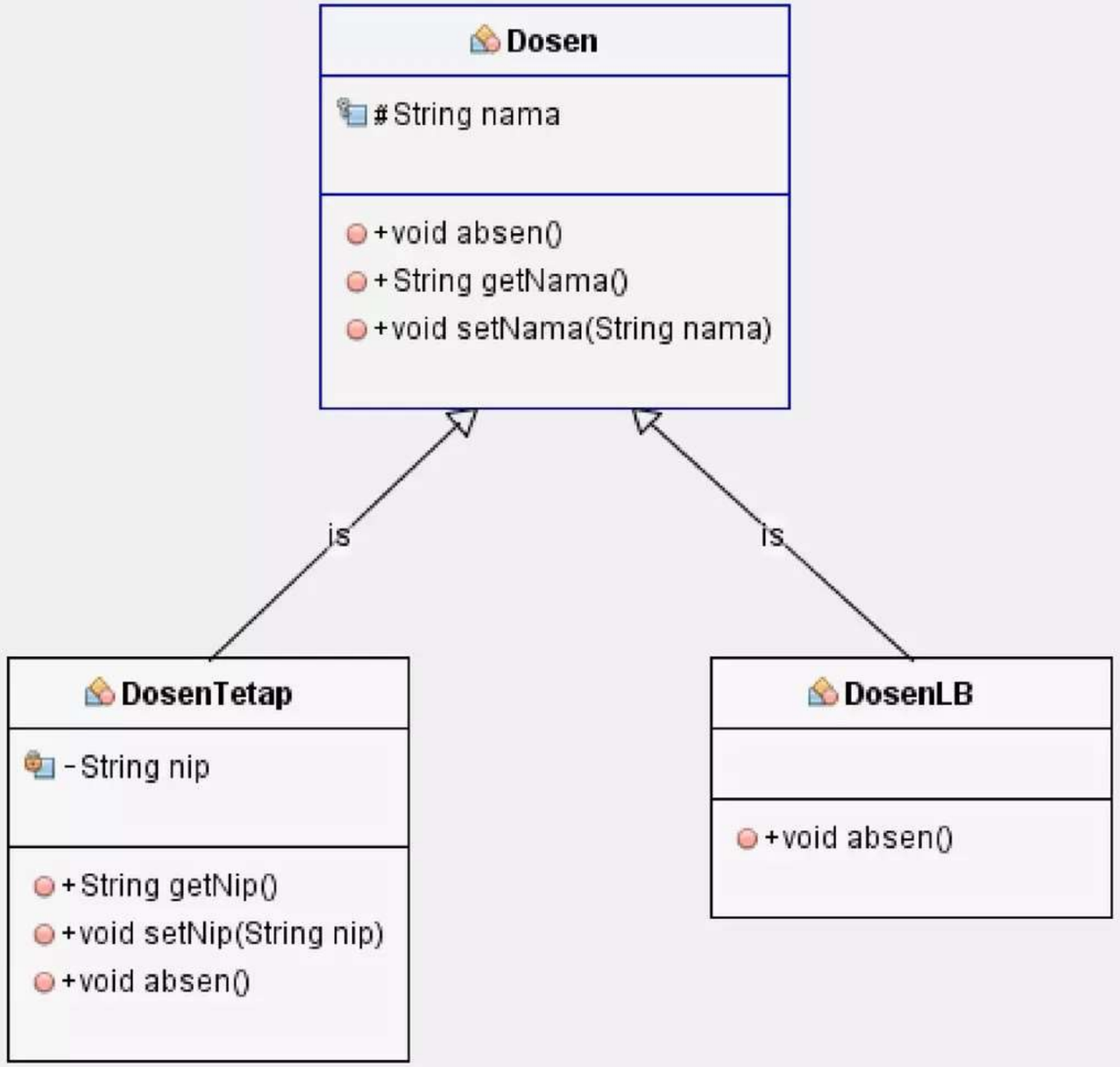

Source : AndriHeryandi, M.T.

PERTEMUAN 11

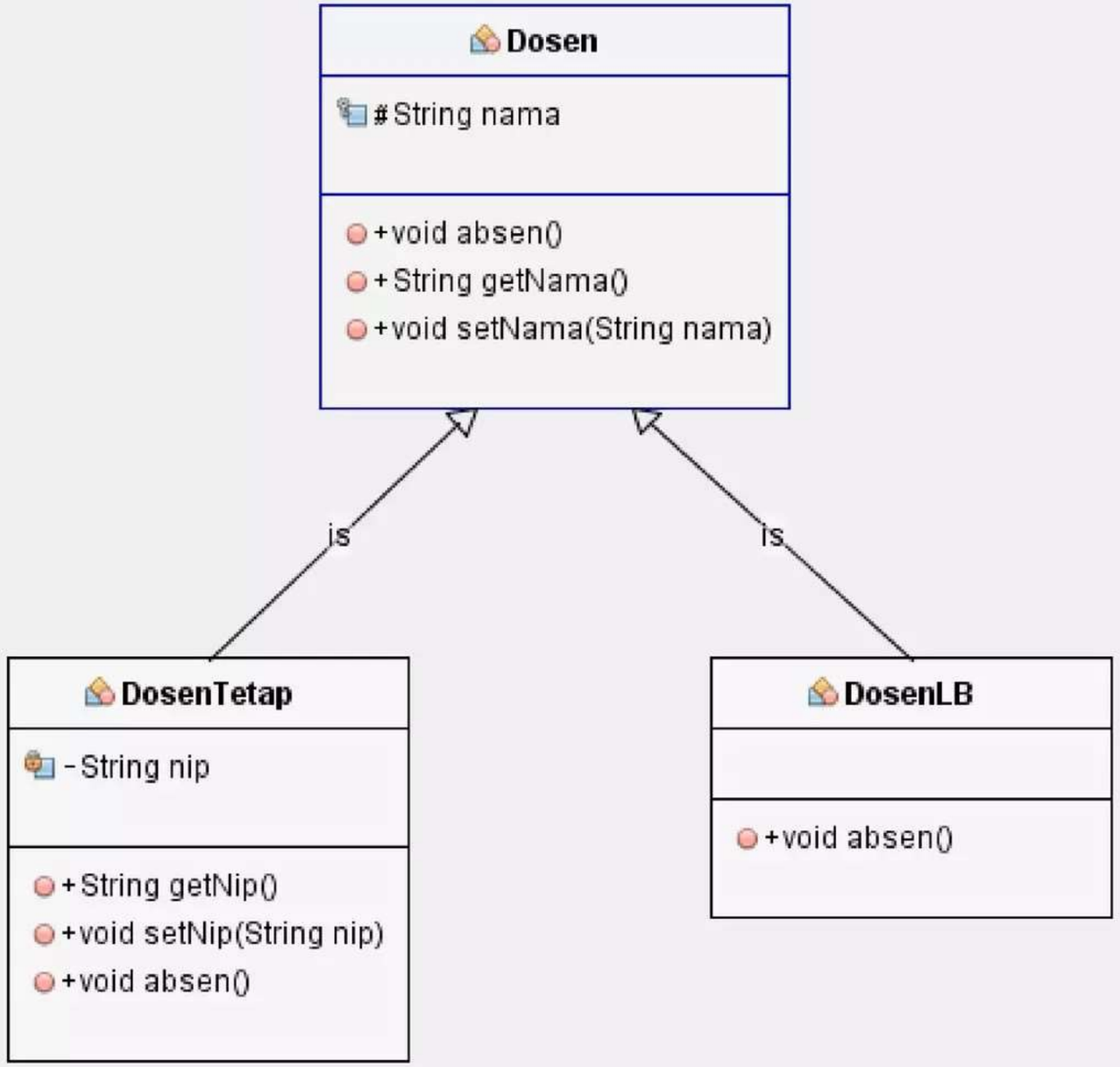
STUDI KASUS POLYMORPHISM



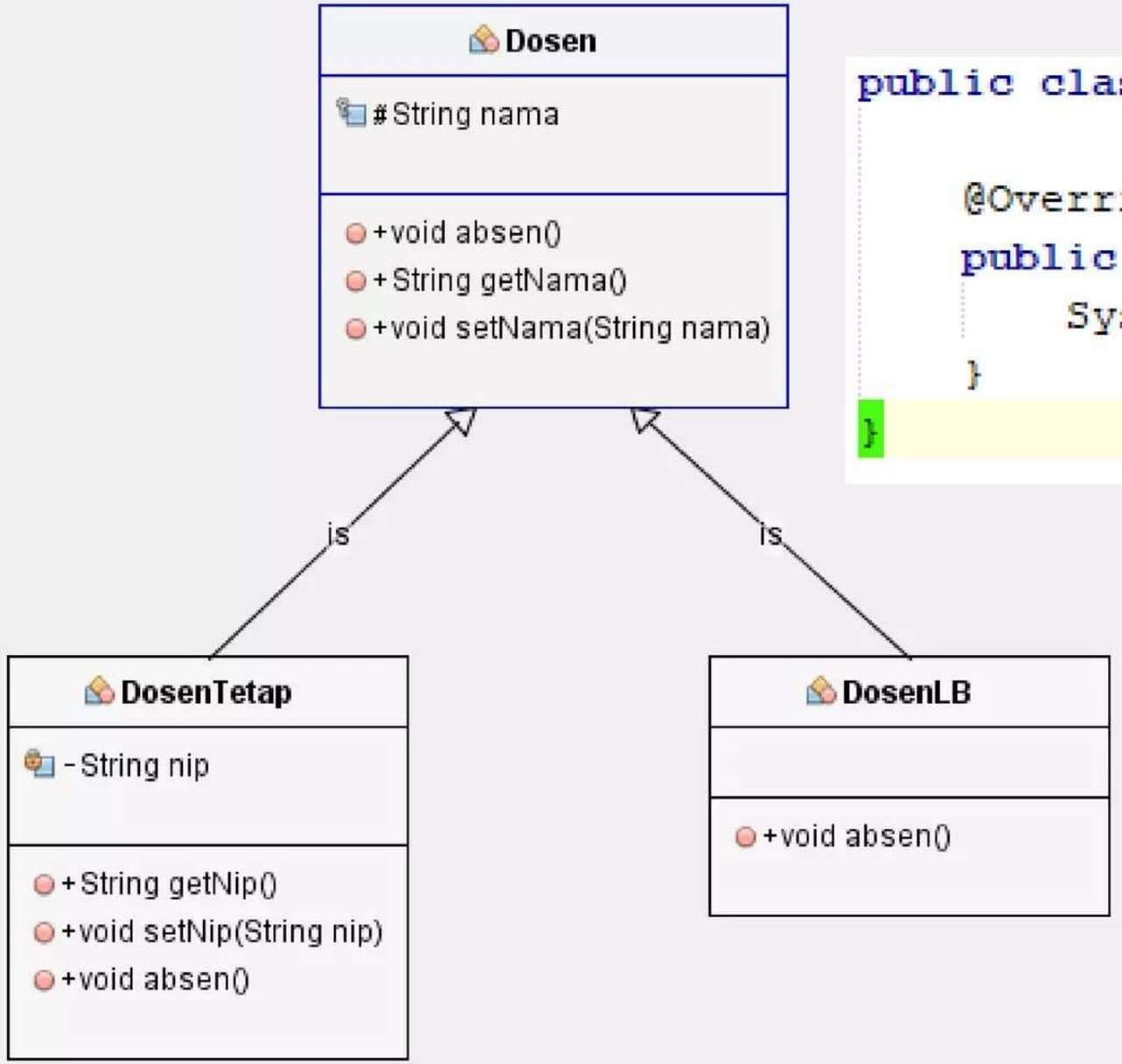
Rifki Dwi Kurniawan, S.Kom | 2023



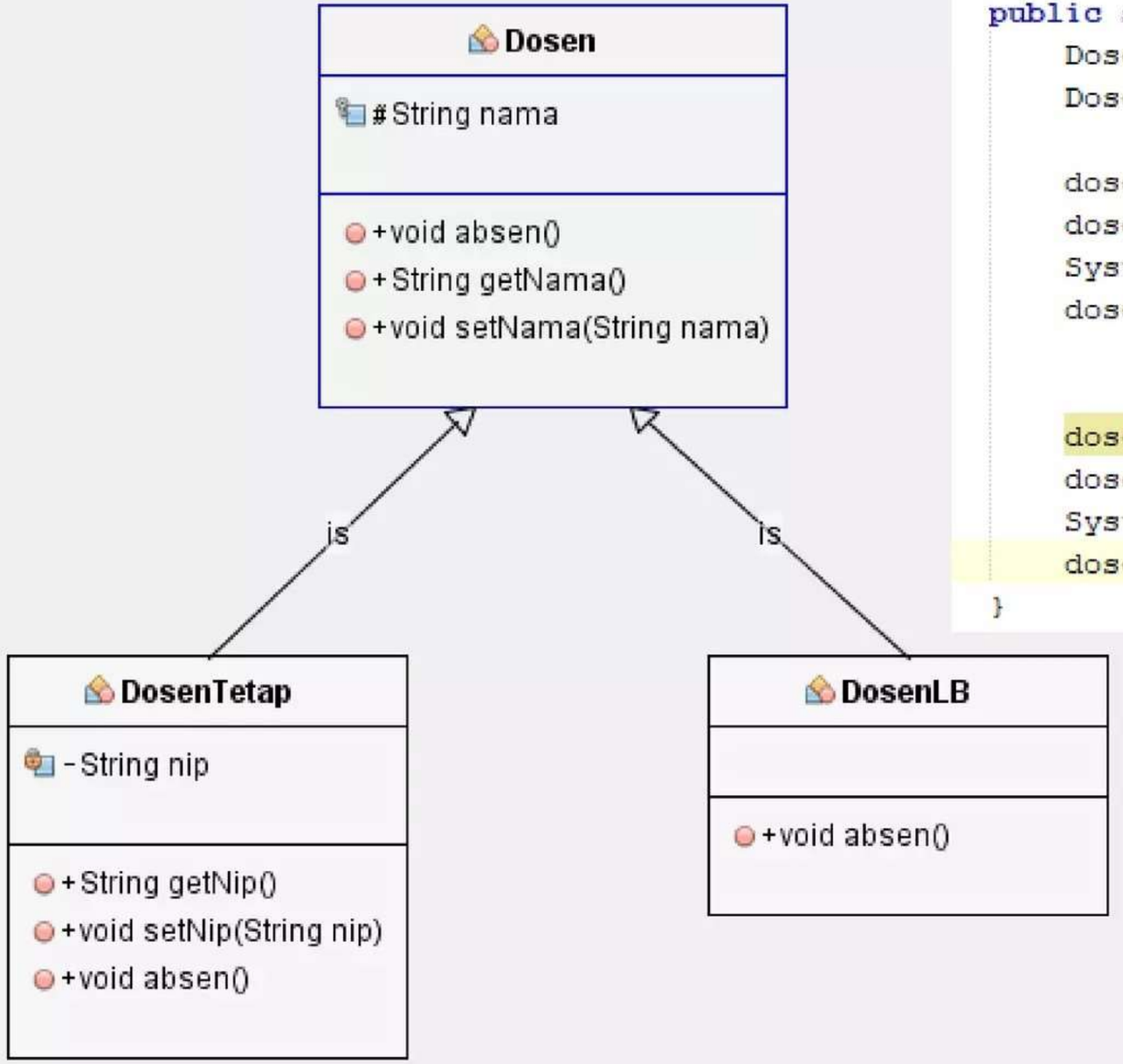

```java
public class Dosen {

    protected String nama;

    public void absen() {
        System.out.println("Melakukan Absen");
    }

    public String getNama() {
        return nama;
    }

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




```java
public class DosenTetap extends Dosen {

    private String nip;

    public String getNip() {
        return nip;
    }

    public void setNip(String nip) {
        this.nip = nip;
    }

    @Override
    public void absen() {
        System.out.println("Absen sidik jari");
    }

}
```

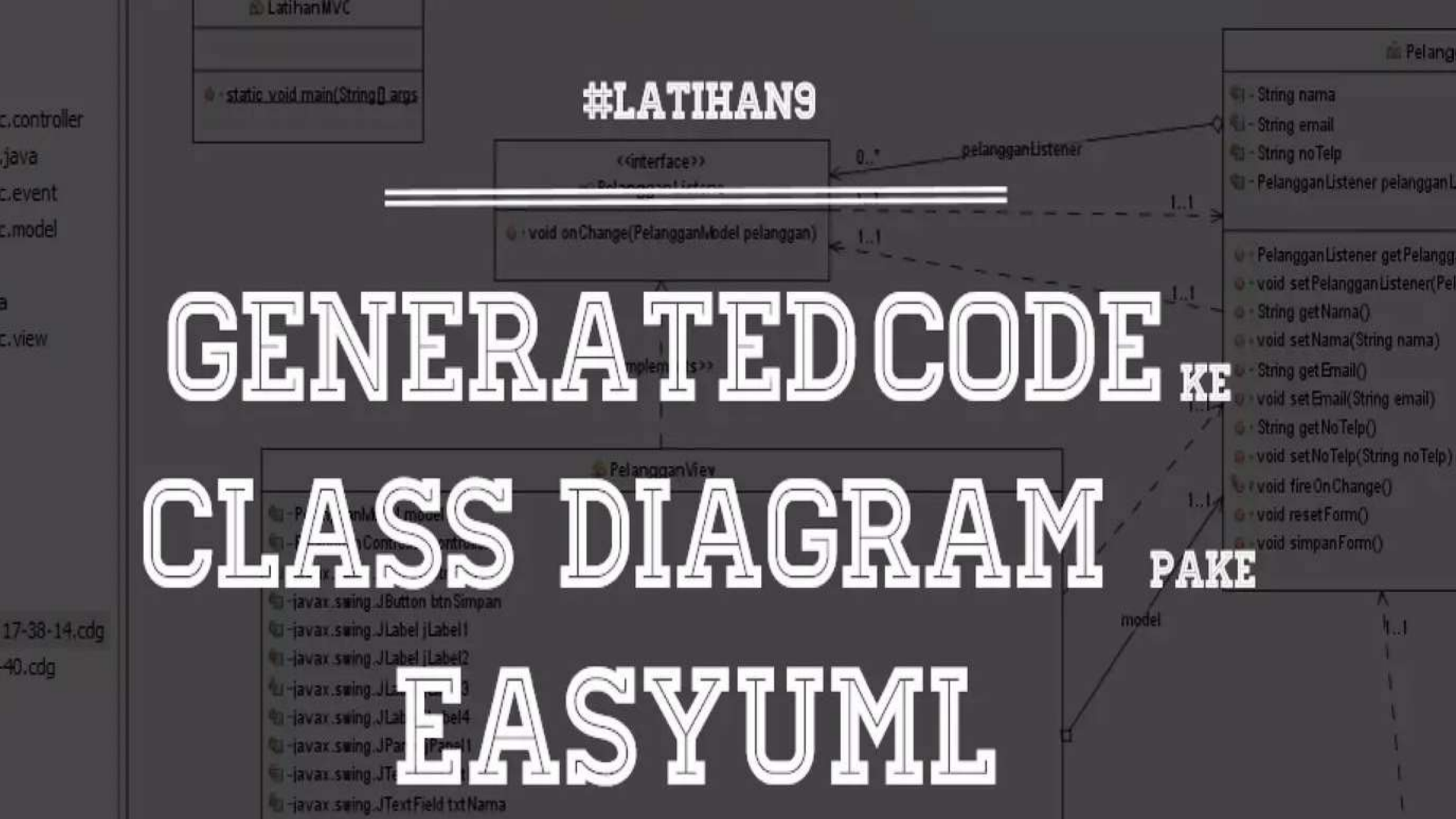



```java
public class DosenLB extends Dosen {

    @Override
    public void absen() {
        System.out.println("Absen tanda tangan");
    }
}
```




```java
public static void main(String[] args) {
    Dosen dosen1;
    Dosen dosen2;

    dosen1 = new DosenTetap();
    dosen1.setNama("Mira Kania");
    System.out.println("Dosen Tetap : " + dosen1.getNama());
    dosen1.absen();


    dosen2 = new DosenLB();
    dosen2.setNama("Rizki Adam");
    System.out.println("Dosen LB : " + dosen2.getNama());
    dosen2.absen();
}
```

```
run:
Dosen Tetap : Mira Kania
Absen sidik jari
Dosen LB : Rizki Adam
Absen tanda tangan
```


#LATIHAN9
GENERATED CODE KE
CLASS DIAGRAM PAKE
EASYUML
LatihanMVC
static void main(String[] args
<<interface>>
void onChange(PelangganModel pelanggan)
pelangganListener
0..*
1..1
String nama
String email
String noTelp
PelangganListener pelangganListener
PelangganListener getPelanggan
void setPelangganListener(Pel
String getNama()
void setNama(String nama)
String getEmail()
void setEmail(String email)
String getNoTelp()
void setNoTelp(String noTelp)
void fireOnChange()
void resetForm()
void simpanForm()
PelangganView
javax.swing.JButton btnSimpan
javax.swing.JLabel jLabel1
javax.swing.JLabel jLabel2
javax.swing.JTextField txtNama
model

PERTEMUAN 11

YUK BERLATIH ☺

Latihan Pertemuan 11

Rifki Dwi Kurniawan, S.Kom | 2023

LATIHAN 51. GAJI KARYAWAN

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:

PERTEMUAN 11

LATIHAN 51. GAJI KARYAWAN

GAJI : TunjanganJabatan +TunjanganGolongan + TunjanganKehadiran

TUNJANGAN KEHADIRAN = Kehadiran *10000

TUNJANGAN JABATAN
Untuk Manager = Rp.2.000.000,-
sedangkan untuk Kabag = Rp.1.000.000,-
Diluar itu Rp 0;

TUNJANGAN GOLONGAN
Golongan 1 memiliki Tunjangan Golongan sebesar Rp.500.000,-
Golongan 2 memiliki Tunjangan Golongan sebesar Rp.1.000.000,-
Golongan 3 memiliki Tunjangan Golongan sebesar Rp.1.500.000,-
Diluar itu Rp 0;

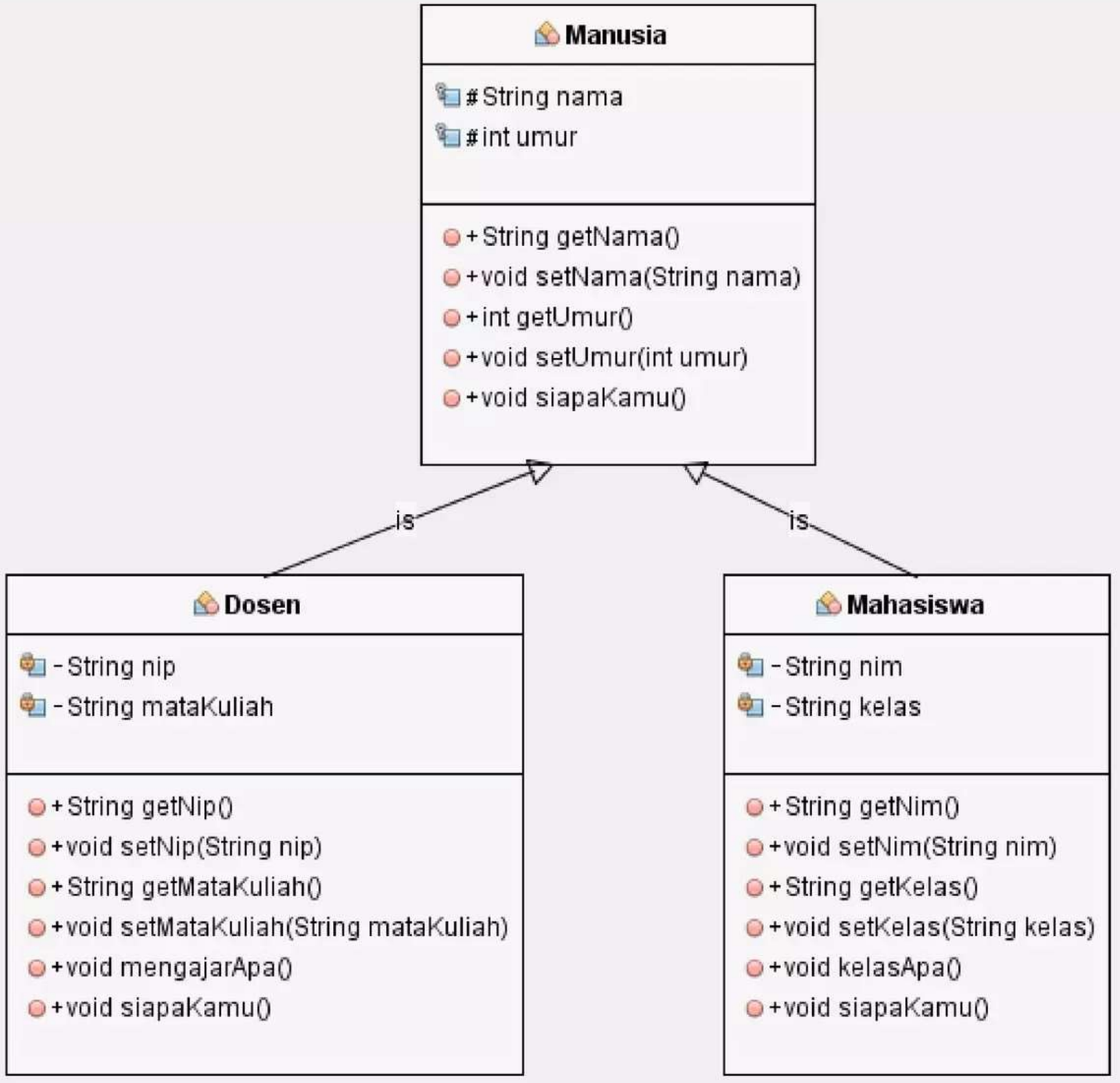
Input : dari keyboard user

```
run:
====Program Perhitungan Gaji Karyawan====
Masukkan NIK : 110110290
Masukkan Nama : Rizki Adam Kurniawan
Masukkan Golongan (1/2/3) : 2
Masukkan Jabatan (Manager/Kabag) : Manager
Masukkan Jumlah Kehadiran : 30

====Hasil Perhitungan====
NIK      : 110110290
NAMA     : Rizki Adam Kurniawan
GOLONGAN: 2
JABATAN : Manager

TUNJANGAN GOLONGAN       : 1000000.0
TUNJANGAN JABATAN        : 2000000.0
TUNJANGAN KEHADIRAN      : 300000.0

GAJI TOTAL       : 3300000.0
```

PERTEMUAN 11

LATIHAN 52. SIAPA KAMU

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:



Rifki Dwi Kurniawan, S.Kom | 2023

PERTEMUAN 11

LATIHAN 52. SIAPA KAMU

Ketentuan :

1. Kelas Manusia – Isi method siapaKamu() dengan menampilkan teks “Saya manusia”.
2. Kelas Dosen – Isi method mengajarApa() agar bisa menampilkan teks “Saya Rizki Adam Kurniawan umur 27 tahun sedang mengajar matakuliah PBO”
3. Kelas Dosen – Isi method siapaKamu() dengan menampilkan teks “Saya Dosen”.
4. Kelas Mahasiswa– Isi method kelasApa() agar bisa menampilkan teks “Saya Nindi umur 17 tahun sedang belajar di kelas PBO2”
5. Kelas Mahasiswa– Isi method siapaKamu() dengan menampilkan teks “Saya Mahasiswa”.
6. Inputan didalam coding, (tidak dari keyboard user)

Lakukan ketentuan seperti diatas, sehingga tampil output seperti berikut :

```
run:
NIP DOSEN : 41227829930
Saya Dosen
Saya Rizki Adam Kurniawan umur 27 tahun sedang mengajar matakuliah PBO

NIM MAHASISWA : 10110269
Saya Mahasiswa
Saya Nindi umur 17 tahun sedang belajar di kelas PBO2
```

PERTEMUAN 11

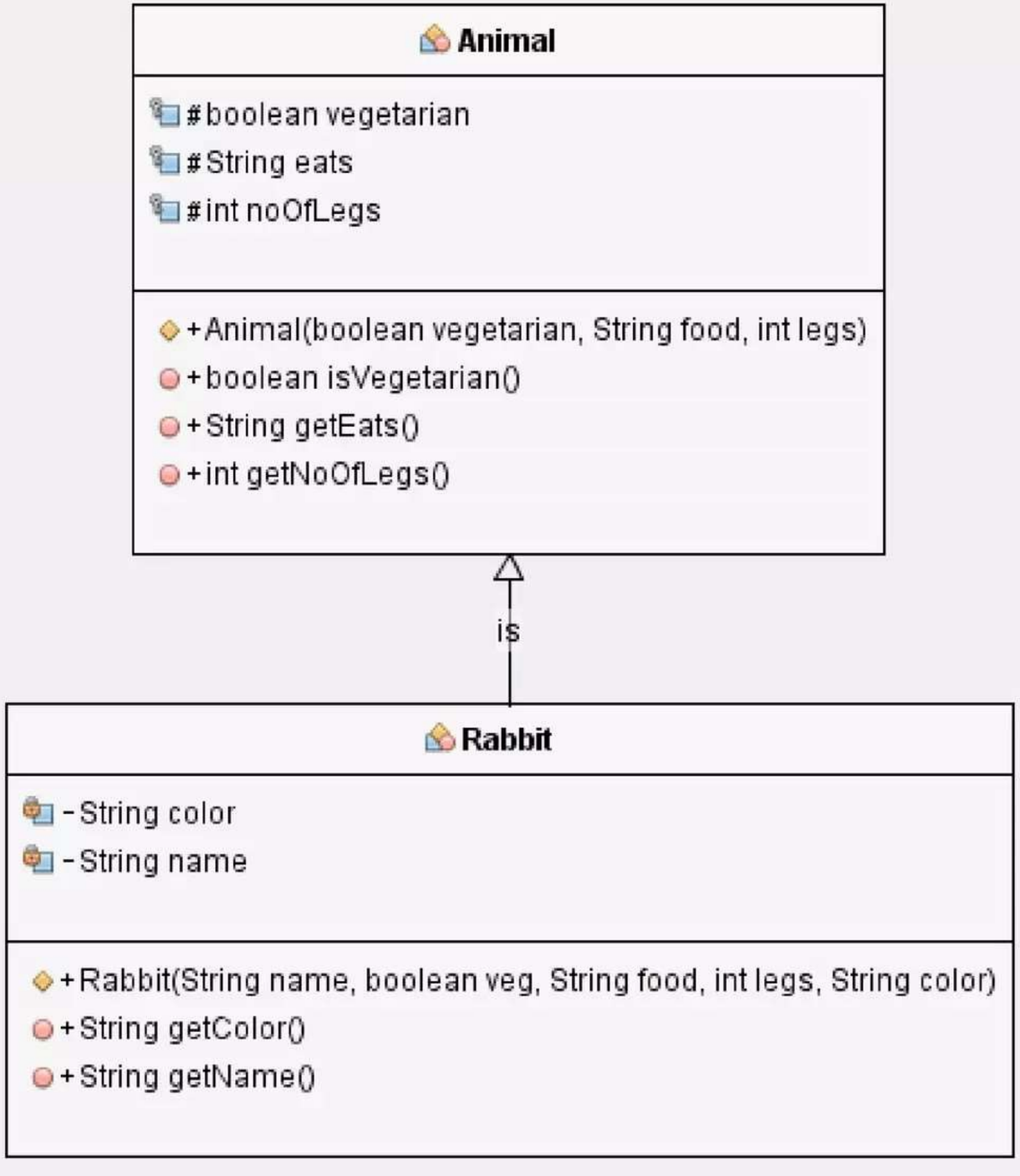
LATIHAN 53. RABBIT

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:

- Inputan didalam coding, (tidak dari keyboard user)

Output :

```
run:
Hello, His name is Peter
Peter is Vegetarian? false
Peter eats grass
Peter has 4 legs.
Peter color is grey
```



PERTEMUAN 11

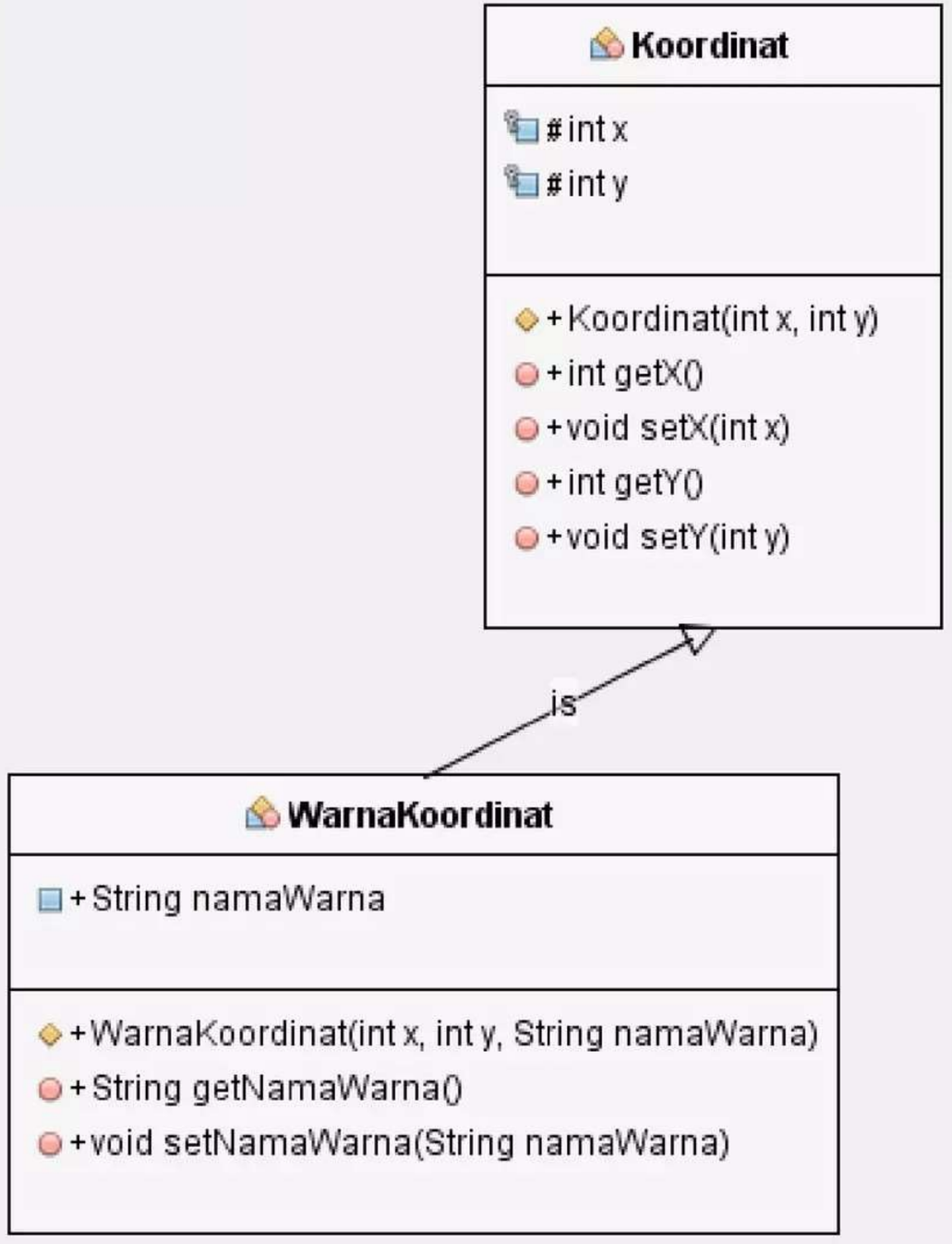
LATIHAN 54. KOORDINAT

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:

- Inputan didalam coding, (tidak dari keyboard user)

Output :

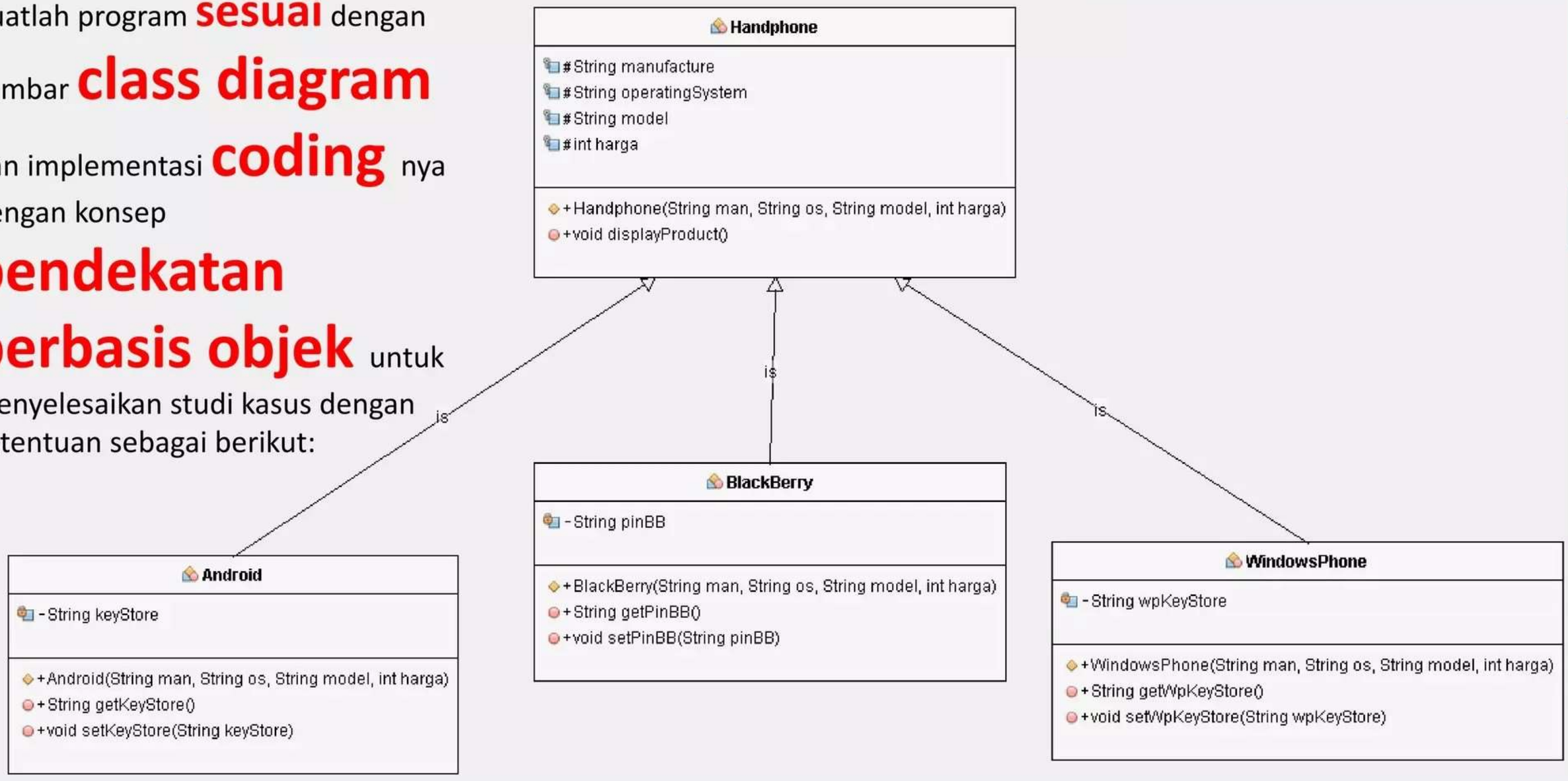
```
run:
Warna Koordinat : Jingga
Koordinat Sumbu x : 10, y : 4
```



PERTEMUAN 11

LATIHAN 55. HANDPHONE

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:



LATIHAN 55. HANDPHONE

- Inputan didalam coding, (tidak dari keyboard user)
- Method displayProduct() menampilkan tampilan ke layar untuk isi dari manufaktur, os, model, dan harga.

Output :

```
Manufaktur : Samsung
OS : Android
Model : Grand
Harga : 3000000
Android Key Store : 234ibfd3840fo

Manufaktur : BlackB
OS : RIM
Model : Curve
Harga : 2000000
PIN : BHS249

Manufaktur : Nokia
OS : Win8
Model : Lumia
Harga : 1000000
Wp Key Store : UUUQIJWORJ
```

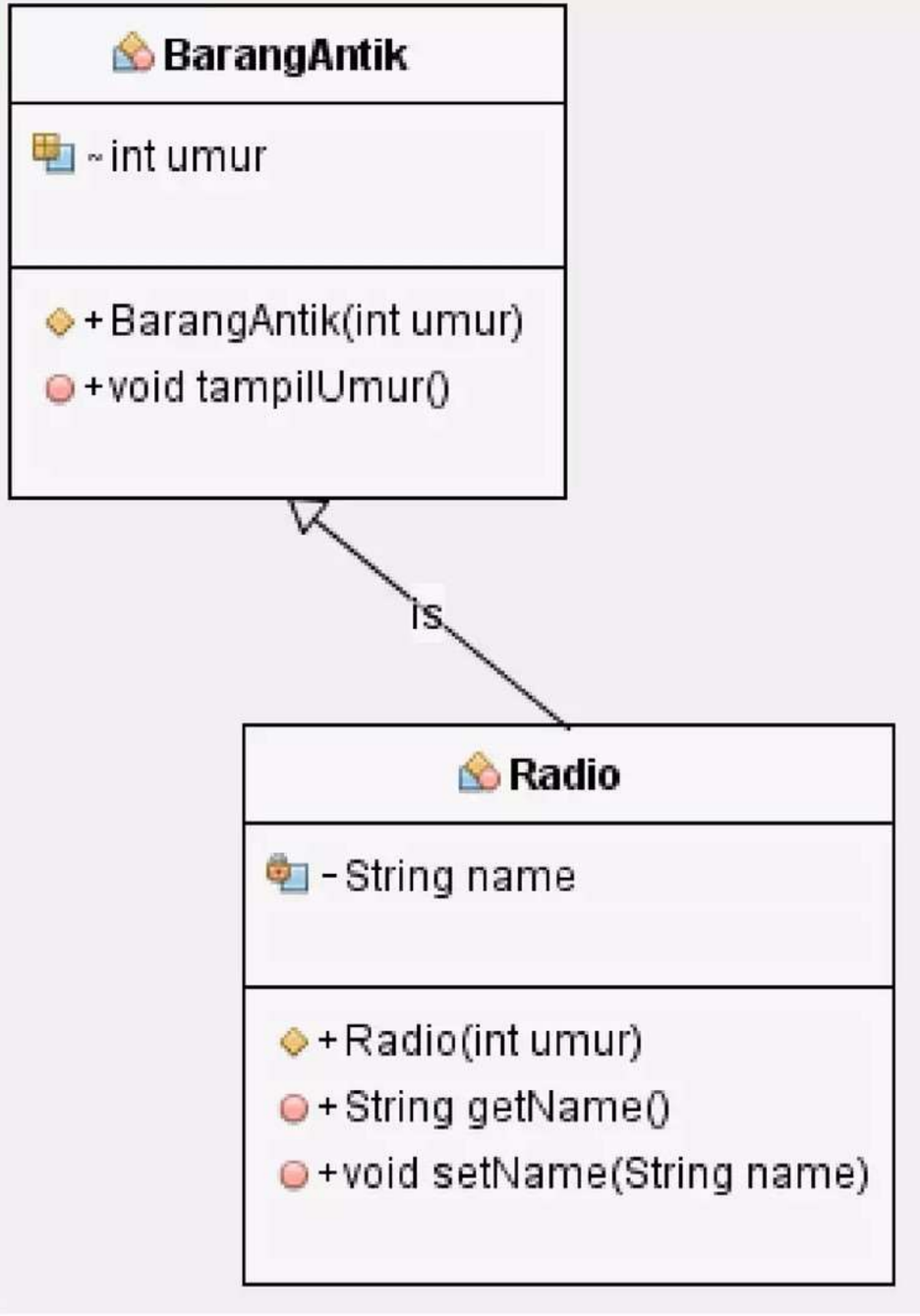
LATIHAN 56. UMUR BARANG ANTIK

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:

- Inputan didalam coding, (tidak dari keyboard user)
- Method tampilUmur() menampilkan tampilan ke layar untuk isi dari umur

Output :

```
run:
Nama barang Antik : Radio AM
Umur barang antik ini adalah : 234 tahun.
```

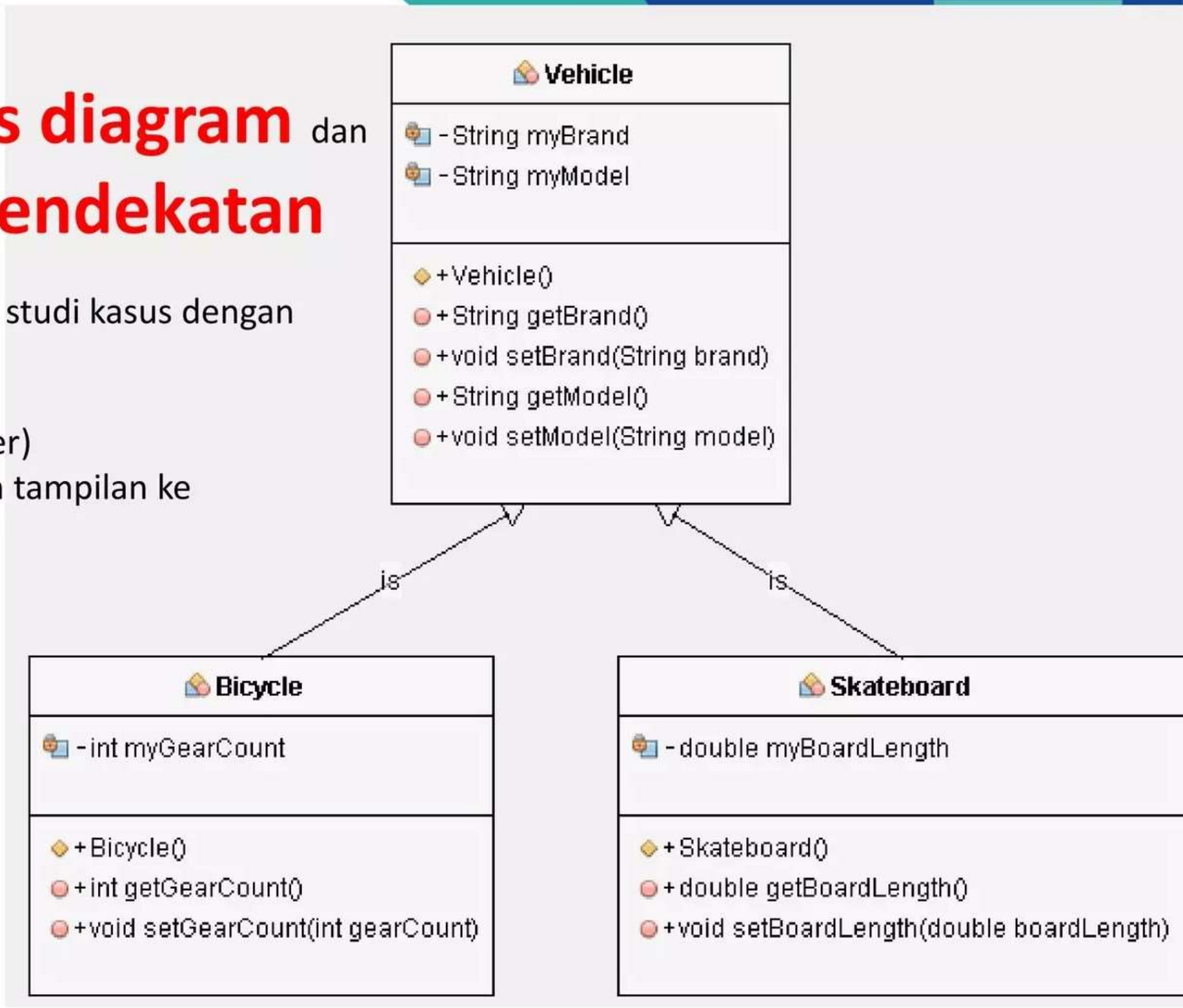


PERTEMUAN 11

LATIHAN 57. VEHICLE

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:

- Inputan didalam coding, (tidak dari keyboard user)
- Konstuktor Bicycle dan Skateboard menampilkan tampilan ke layar nama class mereka sendiri



Output :

```
run:
Bicycle
Brand : Trek Bike
Model : 7.4FX
Jumlah Gear : 23

SkateBoard
Brand : Ally Skate
Model : Rocket
Panjangnya Board: 54.5
```

PERTEMUAN 11

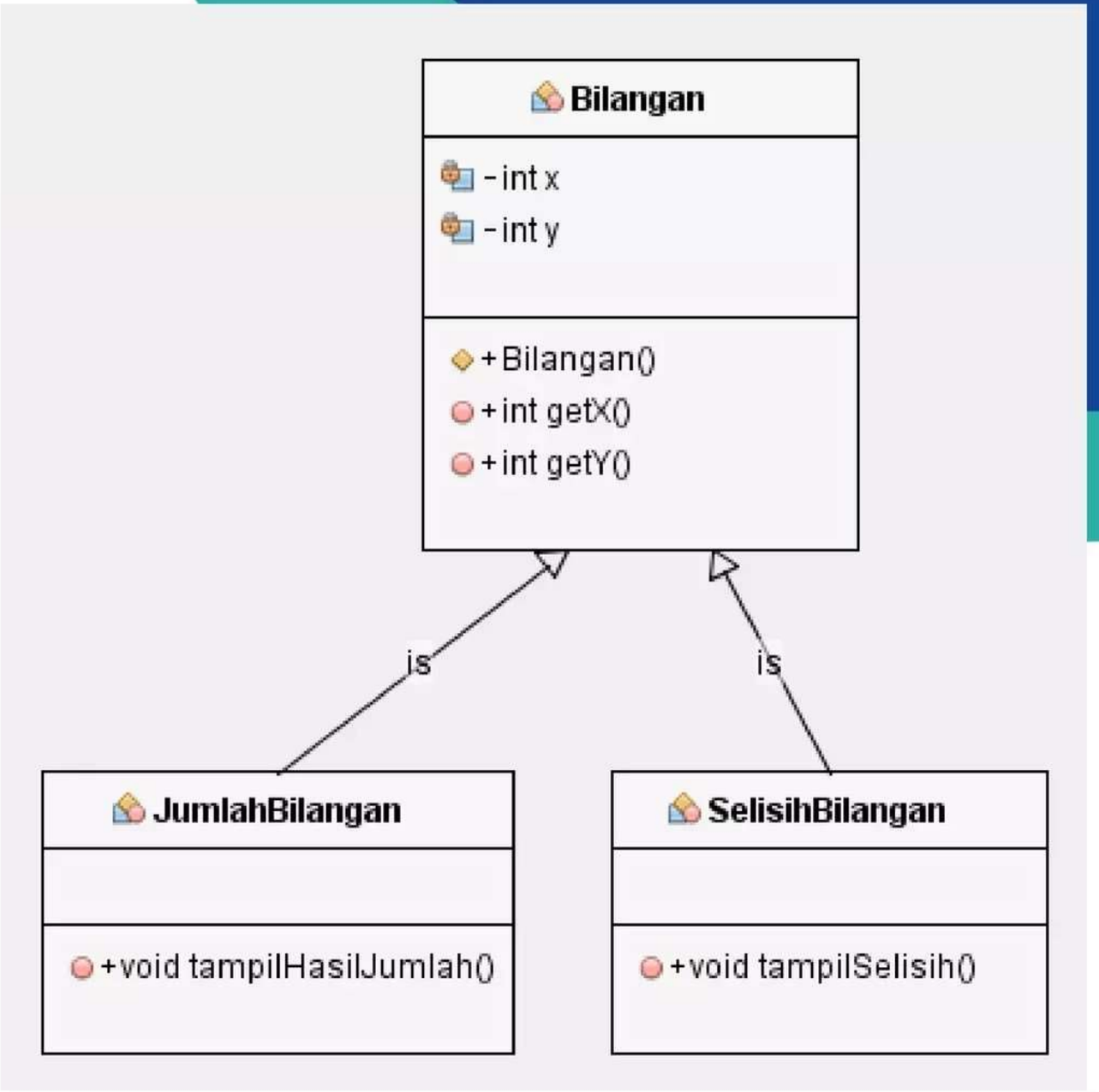
LATIHAN 58. TAMBAH KURANG

Buatlah program **sesuai** dengan gambar **class diagram** dan implementasi **coding** nya dengan konsep **pendekatan berbasis objek** untuk menyelesaikan studi kasus dengan ketentuan sebagai berikut:

- Inputan didalam coding, (tidak dari keyboard user)
- Isi konsturktor Bilangan() dengan x = 3 & y = 4

Output :

```
run:
Hasil perjumlahan = 7
Hasil Selisih 3 - 4 = -1
```



PERTEMUAN 11

LATIHAN 59. DETECTIVE CONAN

Simak baik-baik gambar berikut :

Buatlah sebuah program yang berisikan karakter dari serial anime detective conan disamping dengan pendekatan berbasis objek. Silahkan eksplorasi
Syarat nya :

1. Harus mengimplementasi konsep inheritance
2. Harus mengimplementasi konsep polimorphism
3. Buatlah Gambar Class diagram
4. Buatlah codingan & konsep output sesuai eksplorasi yang kamu buat

PERTEMUAN 11

LATIHAN 60. AKATSUKI

Simak baik-baik gambar berikut :

Buatlah sebuah program yang berisikan karakter akatsuki dari serial anime naruto disamping dengan pendekatan berbasis objek. Silahkan eksplorasi Syarat nya :

1. Harus mengimplementasi konsep inheritance
2. Harus mengimplementasi konsep polimorphism
3. Buatlah Gambar Class diagram
4. Buatlah codingan & konsep output sesuai eksplorasi yang kamu buat

TUGAS INDIVIDU – LATIHAN PERTEMUAN 11

1. Kerjakan **latihan 51-60** di pertemuan 11 sesuai dengan ketentuan.
2. Buat masing-masing project dengan format nama ***NIM-PBO-Lat51***. Ex. ***16145018-PBO-Lat51.***
3. Dalam file coding diberi comment identitas diri, seperti **nama, nim, kelas, semester, matakuliah.**
4. Upload coding setiap project latihan di akun github masing-masing.
5. Gambar class diagram dimasukkan ke dalam **pdf** dan upload ke SIA.

TERIMAKASIH