MAKALAH, PERCOBAAN, LATIHAN, DAN TUGAS
MODUL PRAKTIKUM 6

Disusun sebagai salah satu tugas
mata kuliah PBO I

Patricia Joanne
140810160065

Dikumpulkan tanggal
3 Oktober 2017

PROGRAM STUDI S-1 TEKNIK INFORMATIKA
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS PADJADJARAN
2017

Tugas Pendahuluan

1. Apakah yang dimaksud dengan overloading?
2. Apakah yang dimaksud dengan overriding?
3. Jelaskan aturan tentang overriden!
4. Apakah yang dimaksud dengan polimorfisme?
5. Jelaskan proses terjadinya Virtual Method Invocation!
6. Apakah yang dimaksud dengan polymorphic arguments?
7. Apakah kegunaan kata kunci instanceof?
8. Apa perbedaan antara Abstract dan Interface?

---

1. Overloading adalah suatu keadaan dimana beberapa method sekaligus dapat mempunyai nama yang sama, akan tetapi mempunyai fungsionalitas yang berbeda. Contoh penggunaan overloading dilihat di bawah ini:

`Gambar(Titik t1)` → 1 parameter titik, untuk menggambar titik

`Gambar(Titik t1, Titik t2)` → 2 parameter titik, untuk menggambar garis

`Gambar(Titik t1,Titik t2,Titik t3)` → 3 parameter titik, untuk menggambar segitiga

`Gambar(Titik t1,Titik t2,Titik t3,Titik t4)` → 4 parameter titik, untuk menggambar persegi empat

Overloading ini dapat terjadi pada class yang sama atau pada suatu parent class dan subclass-nya. Overloading mempunyai ciri-ciri sebagai berikut:

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

2. Overriding adalah suatu keadaan dimana method pada subclass menolak method pada parent class-nya. Overriding mempunyai ciri-ciri sebagai berikut:

- Nama method harus sama
- Daftar parameter harus sama
- Return type harus sama

3. Method yang terkena override (overridden method) diharuskan tidak boleh mempunyai modifier yang lebih luas aksesnya dari method yang meng-override (overriding method).

4. Polimorfisme adalah kemampuan untuk mempunyai beberapa bentuk class yang berbeda. Polimorfisme ini terjadi pada saat suatu obyek bertipe parent class, akan tetapi pemanggilan constructornya melalui subclass.

5. Virtual Method Invocation (VMI) bisa terjadi jika terjadi polimorfisme dan overriding. Pada saat obyek yang sudah dibuat tersebut memanggil overridden method pada parent class, kompiler Java akan melakukan invocation (pemanggilan) terhadap overriding method pada subclass, dimana yang seharusnya dipanggil adalah overridden method.

6. Polymorphic arguments adalah tipe suatu parameter yang menerima suatu nilai yang bertipe subclass-nya.

7. Pernyataan instanceof sangat berguna untuk mengetahui tipe asal dari suatu polymorphic arguments. Seringkali pemakaian instanceof diikuti dengan casting object dari tipe parameter ke tipe asal.

8. Perbedaan Abstract dan Interface:

Perbedaan	Abstract	Interface
Attribute	Tipe data apa saja	Hanya berupa konstanta
Method	Boleh deklarasi, boleh method lengkap	Berupa deklarasi
Syntax	Sebagian abstract	Seluruhnya abstract

Bab 6

Overloading, Overriding, dan Polimorfisme

Percobaan 1: Melakukan overloading pada method

```
import java.awt.Point;

public class Segiempat {
    int x1 = 0;
    int y1 = 0;
    int x2 = 0;
    int y2 = 0;

    public void buatSegiempat(int x1, int y1, int x2, int y2){
    this.x1 = x1;
    this.y1 = y1;
    this.x2 = x2;
    this.y2 = y2;
    }

    public void buatSegiempat(Point topLeft, Point bottomRight){
        x1 = topLeft.x;
        y1 = topLeft.y;
        x2 = bottomRight.x;
        y2 = bottomRight.y;
    }

    public void buatSegiempat(Point topLeft, int w, int h){
        x1 = topLeft.x;
        y1 = topLeft.y;
        x2 = (x1 + w);
        y2 = (y1 + h);
    }

    void cetakSegiempat(){
        System.out.print("Segiempat: <" + x1 + ", " + y1);
        System.out.println(", " + x2 + ", " + y2 + ">");
    }
```

```
    public static void main(String[] args){
        Segiempat rect = new Segiempat();
        System.out.println("Buat segiempat dengan koordinat (25,25) dan (50,50)");
        rect.buatSegiempat(25, 25, 50, 50);
        rect.cetakSegiempat();
        System.out.println();

        System.out.println("Buat segiempat dengan point (10,10) dan Point (20,20)");
        rect.buatSegiempat(new Point(10,10), new Point(20,20));
        rect.cetakSegiempat();
        System.out.println();

        System.out.println("Buat segiempat dengan 1 point(10,10), koodinat (50,50)");
        rect.buatSegiempat(new Point(10,10), 50, 50);
        rect.cetakSegiempat();
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Segiempat.java

E:\DOCS\task.bbr\Programming\Java>java Segiempat
Buat segiempat dengan koordinat (25,25) dan (50,50)
Segiempat: <25, 25, 50, 50>

Buat segiempat dengan point (10,10) dan Point (20,20)
Segiempat: <10, 10, 20, 20>

Buat segiempat dengan 1 point(10,10), koodinat (50,50)
Segiempat: <10, 10, 60, 60>
```

Percobaan 2: Memahami proses terjadinya Virtual Method Invocation

```
class Parent {
    int x = 5;

    public void Info(){
        System.out.println("Ini class Parent");
    }
}
```

```
class Child extends Parent {
    int x = 10;
    public void Info(){
        System.out.println("Ini class Child");
    }
}
```

```java
public class Tes {
    public static void main(String args[]){
        Parent tes=new Child();
        System.out.println("Nilai x = "+tes.x);
        tes.Info();
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Tes.java

E:\DOCS\task.bbr\Programming\Java>java Tes
Nilai x = 5
Ini class Child
```

Latihan 1: Overriding

Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan!

```java
class Base {
    private void amethod(int iBase){
        System.out.println("Base.amethod");
    }
}
```

```java
class Over extends Base {
    public static void main(String args[]){
        Over o = new Over();
        int iBase = 0;
        o.amethod(iBase);
    }

    public void amethod(int iOver){
        System.out.println("Over.amethod");
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Over.java

E:\DOCS\task.bbr\Programming\Java>java Over
Over.amethod
```

Latihan 2: Overloading

Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan!

```
class MyParent {
    int x, y;

    MyParent(int x, int y){
        this.x = x;
        this.y = y;
    }

    public int addMe(int x, int y){
        return this.x + x + y + this.y;
    }

    public int addMe(MyParent myPar){
        return addMe(myPar.x, myPar.y);
    }
}
```

```
class MyChild extends MyParent {
    int z;
    MyChild (int x, int y, int z){
        super(x,y);
        this.z = z;
    }

    public int addMe(int x, int y, int z){
        return this.x + x + this.y + y + this.z + z;
    }

    public int addMe(MyChild myChi){
        return addMe(myChi.x, myChi.y, myChi.z);
    }

    public int addMe(int x, int y){
        return this.x + x + this.y + y;
    }
}
```

```
public class MySomeOne {
    public static void main(String args[]){
        MyChild myChi = new MyChild(10, 20, 30);
        MyParent myPar = new MyParent(10, 20);
        int x = myChi.addMe(10, 20, 30);
        int y = myChi.addMe(myChi);
        int z = myPar.addMe(myPar);
        System.out.println(x + y + z);
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac MySomeOne.java

E:\DOCS\task.bbr\Programming\Java>java MySomeOne
300
```

Latihan 3: Overloading

Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan!

```java
class MyClass {
    void myMethod(int i){
        System.out.println("int version");
    }
    void myMethod(String s){
        System.out.println("String version");
    }
    public static void main(String args[]){
        MyClass obj = new MyClass();
        char ch = 'c';
        obj.myMethod(ch);
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac MyClass.java

E:\DOCS\task.bbr\Programming\Java>java MyClass
int version
```

Latihan 4: Mengimplementasikan UML class diagram dalam program

```java
public class Orang {
    protected String nama;
    protected int umur;

    public Orang(String nama){
        this.nama = nama;
    }

    public Orang(String nama, int umur){
        this.nama = nama;
        this.umur = umur;
    }
}
```

```java
public class Dosen extends Orang {
    private int nip;

    public Dosen(String nama){
        super(nama);
    }

    public Dosen(String nama, int nip){
        super(nama);
        this.nip = nip;
    }

    public Dosen(String nama, int nip, int umur){
        super(nama,umur);
        this.nip = nip;
    }

    public void Info(){
        System.out.println("Nama = "+this.nama);
        if(this.nip==0) System.out.println("NIP = -");
        else System.out.println("NIP = "+this.nip);
        if(this.umur==0) System.out.println("Umur = -");
        else System.out.println("Umur = "+this.umur);
    }
}
```

```java
public class TesLatihan {
    public static void main(String args[]){
        System.out.println("Patricia Joanne 1408101160065");
        System.out.println("Memasukkan identitas dosen 1: Agus");
        Dosen dosen1 = new Dosen("Agus");
        System.out.println("Memasukkan identitas dosen 2: Budi, NIP 1458");
        Dosen dosen2 = new Dosen("Budi",1458);
        System.out.println("Memasukkan identitas dosen 3: Iwan, NIP 1215, Umur 47");
        Dosen dosen3 = new Dosen("Iwan",1215,47);

        System.out.println();
        dosen1.Info();
        System.out.println();
        dosen2.Info();
        System.out.println();
        dosen3.Info();
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac TesLatihan.java

E:\DOCS\task.bbr\Programming\Java>java TesLatihan
Patricia Joanne 1408101160065
Memasukkan identitas dosen 1: Agus
Memasukkan identitas dosen 2: Budi, NIP 1458
Memasukkan identitas dosen 3: Iwan, NIP 1215, Umur 47
```

```
Nama = Agus
NIP = -
Umur = -

Nama = Budi
NIP = 1458
Umur = -

Nama = Iwan
NIP = 1215
Umur = 47
```

Latihan 5. Apa yang terjadi bila kode di bawah ini dikompile dan dijalankan jika sebelumnya Base.java belum dikompile? Jelaskan!

```java
//Filename: SuperclassX.java
package packageX;

public class SuperclassX {
    int superclassVarX;
    protected void superclassMethodX(){
    }
}
```

```java
//Filename: SubclassX.java
package packageX.packageY;

public class SubclassY extends SuperclassX {
    SuperclassX objX = new SubclassY();
    SubclassY objY = new SubclassY();

    void subclassMethodY(){
        objY.superclassMethodX();
        int i;
        i = objY.superclassVarX;
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac SuperclassX.java

E:\DOCS\task.bbr\Programming\Java>javac SubclassY.java
SubclassY.java:4: error: cannot find symbol
public class SubclassY extends SuperclassX {
                               ^
  symbol: class SuperclassX
SubclassY.java:5: error: cannot find symbol
        SuperclassX objX = new SubclassY();
        ^
  symbol:   class SuperclassX
  location: class SubclassY
SubclassY.java:9: error: cannot find symbol
                objY.superclassMethodX();
                    ^
  symbol:   method superclassMethodX()
  location: variable objY of type SubclassY
SubclassY.java:11: error: cannot find symbol
                i = objY.superclassVarX;
                         ^
  symbol:   variable superclassVarX
  location: variable objY of type SubclassY
4 errors
```

Latihan 6. Apa yang tampil di layar jika kode di bawah ini dijalankan? Jelaskan!

```java
class Basee {
    int i = 99;
    Basee(){
        amethod();
    }
    public void amethod(){
        System.out.println("Base amethod()");
    }
}
```

```java
public class Derived extends Basee {
    int i = -1;
    public static void main(String argv[]){
        Basee b = new Derived();
        System.out.println(b.i);
        b.amethod();
    }
    public void amethod(){
        System.out.println("Derived.amethod()");
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Derived.java

E:\DOCS\task.bbr\Programming\Java>java Derived
Derived.amethod()
99
Derived.amethod()
```

Latihan 7: Apa yang tampil di layar jika kode dibawah ini dijalankan? Jelaskan!

```
class Parentz {
    private void method1(){
        System.out.println("Parent's method1()");
    }
    public void method2(){
        System.out.println("Parent's method2()");
        method1();
    }
}
```

```
class Childz extends Parentz {
    public void method1(){
        System.out.println("Child's method1()");
    }
    public static void main(String args[]){
        System.out.println("Patricia Joanne 1480160065");
        Parentz p = new Childz();
        p.method2();
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Childz.java

E:\DOCS\task.bbr\Programming\Java>java Childz
Patricia Joanne 1480160065
Parent's method2()
Parent's method1()
```

Latihan 8: Mengimplementasikan UML class diagram dalam program

```
class Pegawai {
    public String nama;
    public Pegawai(){
    }
    public Pegawai(String nama){
        this.nama = nama;
    }
    public String getNama(){
        return nama;
    }
}
```

```
public class Kurir extends Pegawai {
    public int gaji;
    public Kurir(String nama){
        super(nama);
    }
    public Kurir(String nama, int gaji){
        super(nama);
        this.gaji = gaji;
    }
    public int getGaji(){
        return gaji;
    }
}
```

```
public class Manajer extends Pegawai {
    public int tunjangan;
    public Manajer(String nama){
        super(nama);
    }
    public Manajer(String nama, int tunjangan){
        super(nama);
        this.tunjangan = tunjangan;
    }
    public int getTunjangan(){
        return tunjangan;
    }
}
```

```
class Latihan8 {
    public static void Proses(Pegawai peg){
        if(peg instanceof Manajer){
            Manajer man = (Manajer) peg;
            System.out.println("Nama manajer: "+man.nama);
            System.out.println("Tunjangan: "+man.tunjangan);
        }
```

```
        else if(peg instanceof Kurir){
            Kurir kur = (Kurir) peg;
            System.out.println("Nama kurir: "+kur.nama);
            System.out.println("Gaji: "+kur.gaji);
        }
    }
```

```
    public static void main(String[] args){
        System.out.println("Patricia Joanne 140810160065");
        Manajer peg1 = new Manajer("Emma Watson",90000);
        Proses(peg1);
        Kurir peg2 = new Kurir("Emma Stone",60000);
        Proses(peg2);
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Latihan8.java

E:\DOCS\task.bbr\Programming\Java>java Latihan8
Patricia Joanne 140810160065
Nama manajer: Emma Watson
Tunjangan: 90000
Nama kurir: Emma Stone
Gaji: 60000
```

Tugas 1: Mengimplementasikan UML class diagram dalam program

```
class RerataNilai {
    public RerataNilai(){
    }
    public int average(int a, int b){
        return ((a+b)/2);
    }
    public double average(double a, double b){
        return ((a+b)/2);
    }
    public int average(int a, int b, int c){
        return ((a+b+c)/3);
    }
}
```

```
class Tugas1 {
    public static void main(String[] args){
        System.out.println("Patricia Joanne 140810160065");
        RerataNilai hasil1 = new RerataNilai();
        System.out.println("Rerata nilai 21 dan 13 adalah: "+hasil1.average(21,13));
        RerataNilai hasil2 = new RerataNilai();
        System.out.println("Rerata nilai 19.3 dan 39.5 adalah: "+hasil2.average(19.3,39.5));
        RerataNilai hasil3 = new RerataNilai();
        System.out.println("Rerata nilai 123, 567, dan 744 adalah: "+hasil1.average(123,567,744));
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Tugas1.java

E:\DOCS\task.bbr\Programming\Java>java Tugas1
Patricia Joanne 140810160065
Rerata nilai 21 dan 13 adalah: 17
Rerata nilai 19.3 dan 39.5 adalah: 29.4
Rerata nilai 123, 567, dan 744 adalah: 478
```

Tugas 2: Mengimplementasikan UML class diagram dalam program

```
public class Katak {
    int umur;
    String nama;
    String gerak;
    public Katak(int umur, String nama){
        this.umur = umur;
        this.nama = nama;
    }
    public String caraBergerak(){
        return gerak;
    }
    public String getNama(){
        return nama;
    }
    public int getUmur(){
        return umur;
    }
}
```

```
class Kecebong extends Katak {
    double panjangEkor;
    public Kecebong(int umur, String nama, double panjangEkor){
        super(umur,nama);
        this.panjangEkor = panjangEkor;
    }
    public double getEkor(){
        return panjangEkor;
    }
}
```

```
class Tugas2 {
    public static String cariGerak(Katak kat){
        String gerak = kat.caraBergerak();
        if(kat instanceof Kecebong) gerak = "Berenang";
            else gerak = "Melompat";
        return gerak;
    }
```

```java
    public static void main(String[] args){
        System.out.println("Patricia Joanne 140810160065\n");
        Katak q1 = new Katak(5, "Froggy");
        Kecebong q2 = new Kecebong(2, "Junior Frog", 10);
        System.out.println("Objek \t| Umur \t| Nama \t\t| Ekor \t| caraBergerak");
        System.out.println("-----------------------------------------------------");
        System.out.print("O1 \t| " + q1.getUmur() + " \t| " + q1.getNama() + " \t|\t| ");
        System.out.println(cariGerak(q1));
        System.out.print("O2 \t| " + q2.getUmur() + " \t| " + q2.getNama() + " \t| "+ q2.getEkor() +" \t| ");
        System.out.println(cariGerak(q2));
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Tugas2.java

E:\DOCS\task.bbr\Programming\Java>java Tugas2
Patricia Joanne 140810160065

Objek   | Umur  | Nama          | Ekor  | caraBergerak
-----------------------------------------------------
O1      | 5     | Froggy        |       | Melompat
O2      | 2     | Junior Frog   | 10.0  | Berenang
```

Tugas 3: Mengimplementasikan UML class Diagram dalam program

```java
class Pegawai {
    protected String nama;
    protected int gaji;
    public Pegawai(String nama, int gaji){
        this.nama = nama;
        this.gaji = gaji;
    }
    public int infoGaji(){
        return gaji;
    }
}
```

```java
class Manajer extends Pegawai {
    private int tunjangan;
    public Manajer(String nama, int gaji, int tunjangan){
        super(nama, gaji);
        this.tunjangan = tunjangan;
    }
    public int infoGaji(){
        return gaji;
    }
    public int infoTunjangan(){
        return tunjangan;
    }
}
```

```java
class Programmer extends Pegawai {
    private int bonus;
    public Programmer(String nama, int gaji, int bonus){
        super(nama, gaji);
        this.bonus = bonus;
    }
    public int infoGaji(){
        return gaji;
    }
    public int infoBonus(){
        return bonus;
    }
}
```

```java
public class Bayaran {
    public int hitungBayaran(Pegawai peg){
        int uang = peg.infoGaji();
        if (peg instanceof Manajer) uang += ((Manajer) peg).infoTunjangan();
        else if (peg instanceof Programmer) uang += ((Programmer)peg).infoBonus();
        return uang;
    }
    public static void main(String[] args){
        System.out.println("Patricia Joanne 140810160065");
        Manajer man = new Manajer("Agus", 800, 50);
        Programmer prog = new Programmer("Budi", 600, 30);
        Bayaran hr = new Bayaran();

        System.out.println("Bayaran untuk Manajer: " +hr.hitungBayaran(man));
        System.out.println("Bayaran untuk Programmer: " +hr.hitungBayaran(prog));
    }
}
```

```
E:\DOCS\task.bbr\Programming\Java>javac Bayaran.java

E:\DOCS\task.bbr\Programming\Java>java Bayaran
Patricia Joanne 140810160065
Bayaran untuk Manajer: 850
Bayaran untuk Programmer: 630
```