LAPORAN RESMI

Praktikum 12 Overloading dan Overriding

Mata Kuliah: Praktek Pemrograman Berbasis Objek

Dosen Pengampu: Andhik Ampuh Yunanto S.Kom., M.Kom.

Disusun oleh:

M. Ainur Ramadhan (3122500047)

2 D3 Teknik Informatika B

PROGRAM STUDI D3 TEKNIK INFORMATIKA

POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2023/2024

A. TUGAS PENDAHULUAN

1. Memahami tentang overloading

   Jawab:

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

   Gambar(int t1) → 1 parameter titik, untuk menggambar titik
   Gambar(int t1,int t2) → 2 parameter titik, untuk menggambar garis
   Gambar(int t1,int t2,int t3)→3 parameter titik, untuk menggambar segitiga
   Gambar(int t1,int t2,int t3,int 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:
   1. Nama method harus sama
   2. Daftar parameter harus berbeda
   3. Return type boleh sama, juga boleh berbeda

2. Memahami tentang overriding

   Jawab:

   Overriding adalah suatu keadaan dimana method pada subclass menolak method pada parent class-nya. Overriding mempunyai ciri-ciri sebagai berikut :
   1. Nama method harus sama
   2. Daftar parameter harus sama
   3. Return type harus sama

   Berikut ini contoh terjadinya overriding dimana method Info() pada class Child meng-override method Info() pada class parent:

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

3. Memahami aturan tentang overridden

   Jawab:

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

B. LATIHAN

Latihan 1. Overriding

Apa yang terjadi bila program berikut ini dikompile dan dijalnakn? Jelaskan?

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

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

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

Jawab:

Base.java

```
package Praktikum12.Latihan.Latihan1;

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

Over.java

```
package Praktikum12.Latihan.Latihan1;

public class Over extends Base{
    public static void main(String argv[]){
        Over o = new Over();
        int iBase=0;
        o.amethod(iBase);
        }
        public void amethod(int iOver){
        System.out.println("Over.amethod");
        }
}
```

Output

```
PS C:\Users\USER\OOP> javac -cp . Praktikum12/Latihan/Latihan1/*.java

PS C:\Users\USER\OOP> java Praktikum12.Latihan.Latihan1.Over
Over.amethod
```

Penjelasan:
Kode tersebut berjalan dengan baik, karena disini kode Over.java tidak menggunakan method yang ada pada kelas Base melainkan menggunakan methodnya sendiri yang dioverriding. Sehingga kode Over.java berhasil untuk dijalankan.

Latihan2.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);
      }
}
```

Jawab:
Output:

```
PS C:\Users\USER\OOP> javac -cp . Praktikum12/Latihan/Latihan2/*.java
PS C:\Users\USER\OOP> java Praktikum12.Latihan.Latihan2.MySomeOne
300
PS C:\Users\USER\OOP>
```

Penjelasan:
Kode tersebut adalah contoh pewarisan (inheritance) dan penggunaan metode overload di dalam kelas-kelas Java. Ini menciptakan hubungan antara kelas **MyParent** dan **MyChild**.

1. Kelas **MyParent** memiliki dua atribut **x** dan **y**, serta dua metode **addMe**. Metode pertama menerima dua parameter **int x** dan **int y**, dan menghitung jumlah dari **x**, **this.x**, **y**, dan **this.y**. Metode kedua menerima objek **MyParent** sebagai parameter dan memanggil metode pertama dengan parameter dari objek yang diberikan.
2. Kelas **MyChild** adalah turunan dari **MyParent** dan menambahkan atribut **z**. Kelas ini memiliki tiga metode **addMe**. Metode pertama menerima tiga parameter **int x**, **int y**, dan **int z**, dan menghitung jumlah dari atribut **x**, **this.x**, **y**, **this.y**, **z**, dan **this.z**. Metode kedua menerima objek **MyChild** sebagai parameter dan memanggil metode pertama dengan parameter dari objek yang diberikan. Metode ketiga adalah overload dari metode **addMe** di kelas **MyParent**, yang menerima dua parameter **int x** dan **int y**.
3. Dalam metode **main** di kelas **MySomeOne**, membuat objek **MyChild** (**myChi**) dan objek **MyParent** (**myPar**) serta menginisialisasi atribut-atributnya. Kemudian, melakukan pemanggilan beberapa metode **addMe** dan menjumlahkan hasilnya dalam variabel **x**, **y**, dan **z**.
4. Output yang dihasilkan adalah penjumlahan dari variabel **x**, **y**, dan **z** yaitu 300.

Latihan3 Overloading
Apa yang terjadi bila program berikut ini dikompile dan dijalankan? Jelaskan !

```
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);
        }
}
```

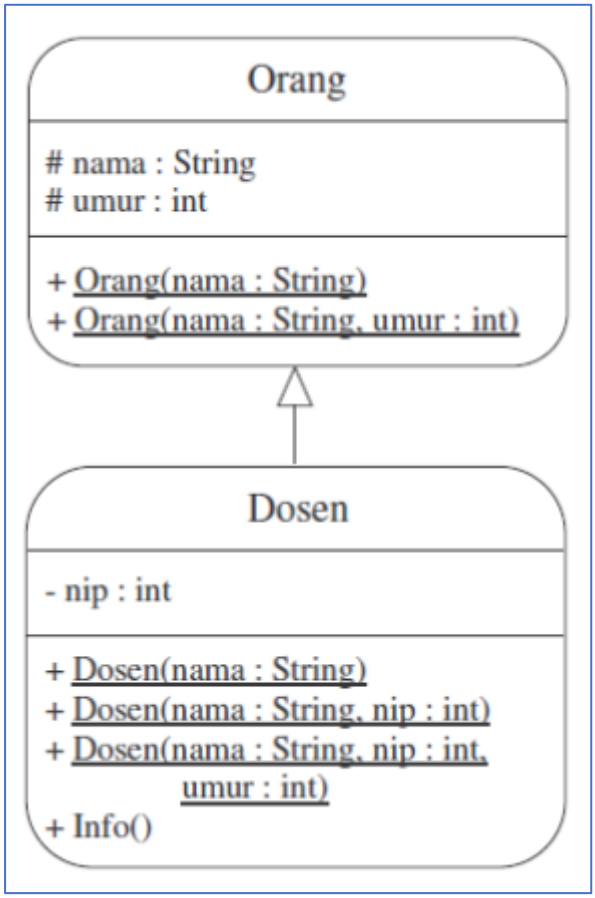
Jawab:
Output:

```
PS C:\Users\USER\OOP> javac -cp . Praktikum12/Latihan/Latihan3/*.java
PS C:\Users\USER\OOP> java Praktikum12.Latihan.Latihan3.MyClass
int version
PS C:\Users\USER\OOP>
```

Penjelasan:
Dalam kode tersebut, metode yang dipilih adalah **myMethod** yang memiliki parameter bertipe data **int**. Kode berjalan lancar karena Java secara otomatis mengonversi karakter **char** menjadi tipe data **int**, dan kemudian memilih metode yang sesuai berdasarkan konversi tersebut.

1. Dalam metode **main**, objek **obj** dari kelas **MyClass** dibuat.
2. Karakter **char ch = 'c';** didefinisikan.
3. Kemudian, **obj.myMethod(ch);** dipanggil, yang seharusnya merupakan pemanggilan metode dengan argumen karakter **ch**.
4. Namun, karena tidak ada metode yang memiliki parameter bertipe karakter (**char**) dalam kelas **MyClass**, Java mencoba mengonversi karakter **ch** menjadi tipe data **int**, dan ini berhasil dilakukan.
5. Java memilih metode **myMethod** yang memiliki parameter bertipe data **int** untuk pemanggilan ini.
6. Hasilnya, output yang dicetak ke layar adalah "int version."

Latihan4 Mengimplementasikan UML class diagram dalam program



Transformasikan class diagram diatas ke dalam bentuk program? Tulislah listing program berikut ini sebagai pengetesan
package Praktikum12.Latihan.Latihan4;

```
public class TesLatihan {
    public static void main(String args[]) {
        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();
    }
}
```

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.

```
Masukkan identitas dosen 1 : Agus
Masukkan identitas dosen 2 : Budi, NIP. 1458
Masukkan identitas dosen 3 : Iwan, NIP. 1215, umur 47

Nama : Agus
NIP : -
Umur : -

Nama : Budi
NIP : 1458
Umur : -

Nama : Iwan
NIP : 1215
Umur : 47
```

Jawab:

Orang.java

```
package Praktikum12.Latihan.Latihan4;

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

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

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

Dosen.java

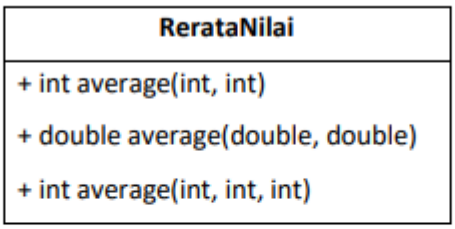
```java
package Praktikum12.Latihan.Latihan4;

public class Dosen extends Orang{
    private int nip;

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

    Dosen(String nama, int nip){
        super(nama);
        this.nip=nip;
    }
    Dosen(String nama, int nip, int umur){
        super(nama,umur);
        this.nip=nip;
    }
    void Info(){
        System.out.println("Nama : " + nama);
        System.out.println("NIP : " + (nip==0 ? "-" : nip));
        System.out.println("Umur : " + (umur==0 ? "-" :umur));
    }
}
```

Output:

```
PS C:\Users\USER\OOP> javac -cp . Praktikum12/Latihan/Latihan4/*.java
PS C:\Users\USER\OOP> java Praktikum12.Latihan.Latihan4.TesLatihan
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
PS C:\Users\USER\OOP>
```

C. TUGAS

Tugas1 Mengimplementasikan UML class diagram dalam program



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

```
public class TesTugas1{
    public static void main(String args[]){
        RerataNilai rn = new RerataNilai();
        System.out.println("Rerata nilai 21 dan 13 adalah : " +
                           rn.average(21, 13));
        System.out.println("Rerata nilai 19.3 dan 39.5 adalah : " +
                           rn.average(19.3, 39.5));
        System.out.println("Rerata nilai 123, 567 dan 744
                           adalah : " + rn.average(123, 567, 744));
    }
}
```

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.

```
Rerata nilai 21 dan 13 adalah : 27
Rerata nilai 19.3 dan 39.5 adalah : 39.05
Rerata nilai 123, 567 dan 744 adalah : 938
```

Jawab:

RerataNilai.java

```
package Praktikum12.Tugas.Tugas1;

public class RerataNilai {
    int x;
    int y;
    int z;
    double a;
    double b;
```

```
    public int average(int x,int y){
        this.x = x;
        this.y = y;
        return((x+y)/2);
    }

    public double average(double a, double b) {
        this.a = a;
        this.b = b;
        return (a + b) / 2.0;
    }

    public int average(int x,int y, int z){
        this.x = x;
        this.y = y;
        this.z = z;
        return (x+y+z)/3;
    }
}
```

TesTugas1.java

```
package Praktikum12.Tugas.Tugas1;

public class TesTugas1 {
    public static void main(String args[]) {
        RerataNilai rn = new RerataNilai();
        System.out.println("Rerata nilai 21 dan 13 adalah : " +
rn.average(21, 13));
        System.out.println("Rerata nilai 19.3 dan 39.5 adalah : " +
rn.average(19.3, 39.5));
        System.out.println("Rerata nilai 123, 567 dan 744 adalah : " +
rn.average(123, 567, 744));
    }
}
```

Output:

```
PS C:\Users\USER\OOP> javac -cp . Praktikum12/Tugas/Tugas1/*.java
PS C:\Users\USER\OOP> java Praktikum12.Tugas.Tugas1.TesTugas1
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
PS C:\Users\USER\OOP>
```

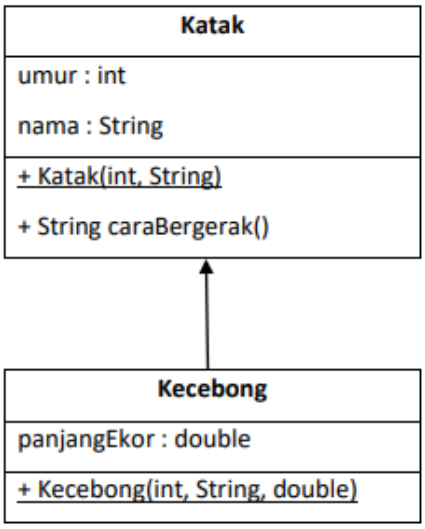
Penjelasan:

Kode tersebut merupakan penggunaan overloading metode untuk menghitung rata-rata dari dua atau tiga angka berbeda. Dalam kelas **RerataNilai**, terdapat tiga versi metode **average**:

1. **average(int x, int y)**: Menghitung rata-rata dari dua angka bulat.
2. **average(double a, double b)**: Menghitung rata-rata dari dua angka floating-point (desimal).
3. **average(int x, int y, int z)**: Menghitung rata-rata dari tiga angka bulat.

Kemudian, dalam kelas **TesTugas1**, objek dari kelas **ReRataNilai** digunakan untuk menghitung rata-rata dari beberapa angka yang berbeda dan mencetak hasilnya.
Program ini akan mencetak rata-rata dari angka-angka yang diberikan dalam tiga pemanggilan metode **average**, dan hasilnya akan ditampilkan di layar.
Namun hasil yang didapat berbeda dengan perintah soal, karena hasil yang didapat pada perintah soal terdapat kesalahan dalam perhitungan rata ratanya, sehingga hasilnya berbeda.

Tugas2 Mengimplementasikan UML class diagram dalam program



Transformasikan class diagram diatas ke dalam bentuk program? Buatlah TesTugas2.java yang membuat obyek dengan karakteristik sebagaimana Tabel 13.1. Selanjutnya tampilkan informasi nama, umur dan cara bergerak obyek-obyek tersebut!

Tabel 13.1 Karakteristik obyek

Obyek	umur	nama	panjangEkor	caraBergerak
O1	5	Froggy		melompat
O2	2	Junior Frog	10	berenang

Jawab:

Dalam kode ini, terdapat dua kelas, yaitu **Katak** dan **Kecebong**, yang merupakan class yang memiliki hubungan pewarisan.

1. Kelas **Katak** memiliki atribut **umur** dan **nama**, serta metode **caraBergerak**. Metode **caraBergerak** mengembalikan string "Melompat."

2. Kelas **Kecebong** adalah turunan dari **Katak** dan memiliki atribut tambahan yaitu **panjangEkor**. Selain itu, kelas **Kecebong** meng-override metode **caraBergerak** yang diwarisi dari kelas **Katak** untuk mengembalikan string "Berenang."
3. Dalam metode **main** di kelas **TesTugas2**, objek **Katak** (**O1**) dan objek **Kecebong** (**O2**) dibuat, menginisialisasi atribut-atributnya, dan mencetak informasi tentang keduanya, termasuk nama, umur, panjang ekor (hanya untuk **Kecebong**), dan cara bergerak.
4. Output yang dihasilkan mencerminkan perilaku pewarisan dan overriding metode. Objek **Katak** akan mengikuti metode **caraBergerak** yang diwarisi dari kelas **Katak**, sementara objek **Kecebong** akan menggunakan metode yang di-override di dalam kelas **Kecebong**.

D. KESIMPULAN

Overloading dan overriding adalah konsep yang penting dalam pemrograman berorientasi objek yang memungkinkan untuk menciptakan fleksibilitas dan modularitas dalam kode. Overloading digunakan untuk menyediakan metode dengan nama yang sama yang berbeda dalam hal parameter, sementara overriding digunakan untuk mengubah perilaku metode yang diwarisi dalam kelas turunan.