

MODUL 13
CLASS, OBJECT, AND OVERLOADING



Adam Arthur Faizal

M3119001

TI A

PROGRAM STUDI TEKNIK INFORMATIKA
SEKOLAH VOKASI
UNIVERSITAS SEBELAS MARET
SURAKARTA
2020

MODUL 13. CLASS, OBJECT, OVERLOADING

Capaian Pembelajaran Praktikum:

- Menerapkan konsep class, object & method
- Menerapkan overloading konstruktor/method

Tools:

- Java Development Kit (JDK)
- Eclipse

Terminologi:

Isikan terminology yang sesuai untuk definisi dibawah ini:

[this] An optional keyword used to access the members and methods of a class.

[garbage collector] A built-in function of the Java VM that frees memory as objects are no longer needed or referenced.

[Finalizers] An optional method that is called just before an object is removed by the garbage collector.

[Overloading Constructors] Having more than one constructor or method with the same name but different arguments.

[Access modifiers] Used to specify accessibility for variables, methods, and classes.

[default constructor] A constructor that does not have any parameters

[variable arguments method] A way to call a method with a variable number of arguments.

TRY IT / SOLVE IT:

1. Identifikas bagian dari class java berikut. Beri tanda asterisk (*) untuk semua instance variabel. Tandai dengan box pada masing-masing konstruktor. Lingkari signature method nya. Tandai dengan segitiga di sekitar parameternya. Garis bawah tipe nilai kembalian dari method.

```
public class Animal {
    int weight, height;
    double speed;
    Animal() {
        weight = 50;
        height = 4;
        speed = 2; //miles per hour
    }
    Animal(int w, int h, int s ) {
        weight = w;
        h = height;
        speed = s
    }
    public double getTime(double miles) {
        return miles/speed;
    }
    public int getWeight() {
        return weight;
    }
    public int getHeight() {
        return height;
    }
    public double getSpeed() {
        return speed;
    }
}
```

J *Animal.java

```
1  /**
4  package com.latihan6;
5
7  * @author adam
10 public class Animal {
11     private int weight, height; // Inisialisasi variabel
12     private double speed;
13
14     public Animal() { // Ini adalah default constructor
15         this.weight = 50; // Sedangkan constructor-constructor dibawah
16         this.height = 4; // adalah overloading constructor
17         this.speed = 2;
18     }
19
20     public Animal(int weight, int height, int speed) {
21         this.weight = weight;
22         this.height = height;
23         this.speed = speed;
24     }
25
26     public double getTime(double miles) {
27         return miles / this.speed;
28     }
29
31     * @return the weight
33     public int getWeight() {
34         return this.weight;
35     }
36
38     * @return the height
40     public int getHeight() {
41         return this.height;
42     }
43
45     * @return the speed
47     public double getSpeed() {
48         return this.speed;
49     }
50 }
51
```

2. Tambahkan method main pada program pada nomor 1 untuk membuat 2 objek dengan menggunakan semua konstruktor yang disediakan (nilai parameter sembarang). Kemudian tambahkan kode untuk mencetak ke layar sbb:
- Animal #1 memiliki kecepatan (isian sesuai dengan nilai parameter)
 - Animal #2 memiliki kecepatan (isian sesuai dengan nilai parameter)

```
61 class UjiAnimal {
62
63     public static void main(String[] args) {
64         Animal cheetah = new Animal("Cheetah", 70, 0.9, 58);
65         Animal kijang = new Animal("Kijang", 35, 0.9, 54);
66
67         System.out.printf("---- %s ----\n", cheetah.getNama());
68         System.out.printf("%s memiliki berat %.1f kilogram, ", cheetah.getNama(), cheetah.getWeight());
69         System.out.printf("tinggi %.1f meter, ", cheetah.getHeight());
70         System.out.printf("dan kecepatan lari %.1f mil/jam\n", cheetah.getSpeed());
71
72         System.out.printf("---- %s ----\n", kijang.getNama());
73         System.out.printf("%s memiliki berat %.1f kilogram, ", kijang.getNama(), kijang.getWeight());
74         System.out.printf("tinggi %.1f meter, ", kijang.getHeight());
75         System.out.printf("dan kecepatan lari %.1f mil/jam\n", kijang.getSpeed());
76
77     }
78 }
79
```

```
Console
<terminated> UjiAnimal [Java Application] /usr/lib/jvm/java-13-openjdk-amd64/bin/java (May 11, 2020, 7:49:55 PM)
---- Cheetah ----
Cheetah memiliki berat 70.0 kilogram, tinggi 0.9 meter, dan kecepatan lari 58.0 mil/jam
---- Kijang ----
Kijang memiliki berat 35.0 kilogram, tinggi 0.9 meter, dan kecepatan lari 54.0 mil/jam
```

3. Buatlah sebuah class public dengan nama Fish dengan fitur sbb:

- Class *Fish* memiliki 2 instance variabel yakni *typeOffish* bertipe String dan *friendliness* bertipe integer.
- Buatlah konstruktor default, dimana body konstruktor akan memberikan nilai "Unknown" untuk variabel *typeOffish* dan nilai 3 untuk variabel *friendliness*.
- Tambahkan konstruktor yang lain dengan 2 parameter yakni String *t* dan integer *f*. Dalam body konstruktor nilai parameter *t* akan mengisi variabel *typeoffish* dan parameter *f* akan mengisi variabel *friendliness*.
- Tambahkan method setter dan getter untuk masing-masing instance variable.
- Tulis segment code untuk membuat 2 objek fish yakni Amber dan James dengan definisi sbb:
 - Amber -- Type: AngelFish, Frindliness level: 5
 - James - Type: Guppy, Friendliness level: 3
- Buat functional method *nicestFish* dengan parameter berupa 2 objek fish yang membandingkan level *friendliness* dari 2 fish tersebut dan akan mengembalikan objek fish yang memiliki level *friendliness* paling tinggi. Uji method ini dengan objek fish pada poin e. (*Friendliness scale*: 1 mean, 2 not friendly, 3 neutral, 4 friendly, 5 very friendly)
Hint: *fishName.getFriendliness()* akan mengambil nilai *friendliness* dari *fishName*.
- Tambahkan method *nicestFish* yang lain untuk membandingkan lebih dari 2 objek fish sebagai berikut:

```
Fish nicestFish(Fish ... a){
    Fish temp = a[0];
    for(int i = 1; i < a.length; i++)
        if(temp.getFriendliness() < a[i].getFriendliness())
            temp = a[i];
    return temp;
}
```

- Uji method pada poin g dengan membandingkan 3 objek. (Sebelumnya anda tambahkan satu objek fish dengan parameter yang berbeda dari objek amber dan james). Jelaskan cara kerjanya.



```
1 /**
4 package com.latihan6;
5
7 * @author adam
10 public interface SeaAnimal {
11     public void berenang();
12 }
13
```

```

J Fish.java x J Animal.java J SeaAnimal.java
10 /**
4 package com.latihan6;
5
7 * @author adam
10 public class Fish extends Animal implements SeaAnimal {
11     private String typeOfFish;
12     private int friendLiness;
13
14     public Fish() { // default constructor
15         this.typeOfFish = null;
16         this.friendLiness = 3;
17     }
18
19     public Fish(String nama, String typeOfFish, int friendLiness) {
20         this.setNama(nama);
21         this.typeOfFish = typeOfFish;
22         this.friendLiness = friendLiness;
23     }
24
26     * @param nama
31     public Fish(String nama, int weight, double height, int speed, String typeOfFish, int friendLiness) {
32         super(nama, weight, height, speed);
33         this.typeOfFish = typeOfFish;
34         this.friendLiness = friendLiness;
35     }
36
38     * @return the typeOfFish
40     public String getTypeOfFish() {
41         return this.typeOfFish;
42     }

```

```

J Fish.java x J Animal.java J SeaAnimal.java
45     * @param typeOfFish the typeOfFish to set
47     public void setTypeOfFish(String typeOfFish) {
48         this.typeOfFish = typeOfFish;
49     }
50
52     * @return the friendLiness
54     public int getFriendLiness() {
55         return this.friendLiness;
56     }
57
59     * @param friendLiness the friendLiness to set
61     public void setFriendLiness(int friendLiness) {
62         this.friendLiness = friendLiness;
63     }
64
65     public static Fish nicestFish(Fish fish1, Fish fish2) {
66         if (fish1.getFriendLiness() < fish2.getFriendLiness())
67             return fish1;
68         return fish2;
69     }
70
71     public static Fish nicestFish(Fish ...fishs) {
72         Fish temp = fishs[0];
73         for (int i = 0; i < fishs.length; i++) {
74             if (temp.getFriendLiness() < fishs[i].getFriendLiness()) {
75                 temp = fishs[i];
76             }
77         }
78         return temp;
79     }

```



```
J Fish.java x J Animal.java J SeaAnimal.java
72     Fish temp = fishs[0];
73     for (int i = 0; i < fishs.length; i++) {
74         if (temp.getFriendliness() < fishs[i].getFriendliness()) {
75             temp = fishs[i];
76         }
77     }
78     return temp;
79 }
80
81 @Override
82 public void berenang() {
83     System.out.println("Yeah I'm Pro Player.");
84 }
85 }
86
87 class UjiFish {
88
89     public static void main(String[] args) {
90         Fish Amber = new Fish("Amber", "AngelFish", 5);
91         Fish James = new Fish("James", "Guppy", 3);
92         Fish Arthur = new Fish("Arthur", "MbahFish", 5);
93
94         Fish hasil1 = Fish.nicestFish(Amber, James);
95         System.out.printf("%s memiliki friendliness yang paling tinggi\n", hasil1.getNama());
96         Fish hasil2 = Fish.nicestFish(Amber, James, Arthur);
97         System.out.printf("%s memiliki friendliness yang paling tinggi\n", hasil2.getNama());
98     }
99 }
100 }
```

```
Console x
<terminated> UjiFish [Java Application] /usr/lib/jvm/java-13-openjdk-amd
James memiliki friendliness yang paling tinggi
Amber memiliki friendliness yang paling tinggi
```


LATIHAN:

4. Buat program class Mahasiswa dengan atribut NIM, Nama, Prodi, IPK, Status. Kemudian buatlah 3 method dengan nama yang sama *cetakMahasiswa()* untuk menerapkan overloading, yakni:
 - Cetak nama dan NIM
 - Cetak nama, NIM dan prodi
 - Cetak nama, NIM, prodi, IPK dan status.

Anda uji method yang anda buat dengan membuat beberapa objek Mahasiswa.

```
J Mahasiswa.java J MahasiswaInformatika.java ✖
10 /**
4 package com.latihan6;
5
70 * @author adam
10 public interface MahasiswaInformatika {
11     public void Ngoding();
12 }
```

```
J Mahasiswa.java ✖ J MahasiswaInformatika.java
10 /**
4 package com.latihan6;
5
70 * @author adam
10 public class Mahasiswa implements MahasiswaInformatika {
11     private String nama, nim, prodi;
12     private double ipk;
13     private boolean status;
14
15     public Mahasiswa(String nama, String nim, String prodi, double ipk, boolean status) {
16         this.nama = nama;
17         this.nim = nim;
18         this.prodi = prodi;
19         this.ipk = ipk;
20         this.status = status;
21     }
22
23     public static String cetakMahasiswa(String nama, String nim){
24         return String.format("Mahasiswa ini bernama %s dan memiliki nim %s\n",nama, nim);
25     }
26
27     public static String cetakMahasiswa(String nama, String nim, String prodi) {
28         return String.format("Mahasiswa ini bernama %s, memiliki nim %s, dan prodi %s\n", nama, nim, prodi);
29     }
30
31     public static String cetakMahasiswa(String nama, String nim, String prodi, double ipk, boolean status) {
32         return String.format("Mahasiswa ini bernama %s, memiliki nim %s, prodi %s, ipk %f, dan statusnya %b\n", nama, nim, prodi, ipk, status);
33     }
34
36     * @return the nama
38     public String getNama() {
39         return this.nama;
40     }
41 }
```

```
43  * @param nama the nama to set
44
45  public void setName(String nama) {
46      this.nama = nama;
47  }
48
49
50  * @return the nim
51
52  public String getNim() {
53      return this.nim;
54  }
55
56
57  * @param nim the nim to set
58
59  public void setNim(String nim) {
60      this.nim = nim;
61  }
62
63
64  * @return the prodi
65
66  public String getProdi() {
67      return this.prodi;
68  }
69
70
71  * @param prodi the prodi to set
72
73  public void setProdi(String prodi) {
74      this.prodi = prodi;
75  }
76
77
78  * @return the ipk
79
80  public double getIpk() {
81      return this.ipk;
82  }
83
84
85  * @param ipk the ipk to set
86
87  public void setIpk(double ipk) {
88      this.ipk = ipk;
89  }
```

```

Mahasiswa.java  MahasiswaInformatika.java
87  public void setIpk(double ipk) {
88      this.ipk = ipk;
89  }
90
91  * @return the status
92  public boolean isStatus() {}
93
94  * @param status the status to set
95  public void setStatus(boolean status) {
96      this.status = status;
97  }
98
99  @Override
100 public void Ngoding() {
101     System.out.println("Yeah I'm a Pro Player.");
102 }
103
104
105 class UjiMahasiswa {
106
107     public static void main(String[] args) {
108         Mahasiswa m1 = new Mahasiswa("Mbah Putih", "M3119000", "Teknik Informatika", 7.63, true);
109         Mahasiswa m2 = new Mahasiswa("Arthur", "M3119001", "Teknik Informatika", 3.67, true);
110
111         System.out.println(Mahasiswa.cetakMahasiswa(m1.getNama(), m1.getNim()));
112         System.out.println(Mahasiswa.cetakMahasiswa(m1.getNama(), m1.getNim(), m1.getProdi()));
113         System.out.println(Mahasiswa.cetakMahasiswa(m1.getNama(), m1.getNim(), m1.getProdi(), m1.getIpk(), m1.isStatus()));
114
115         System.out.println(Mahasiswa.cetakMahasiswa(m2.getNama(), m2.getNim()));
116         System.out.println(Mahasiswa.cetakMahasiswa(m2.getNama(), m2.getNim(), m2.getProdi()));
117         System.out.println(Mahasiswa.cetakMahasiswa(m2.getNama(), m2.getNim(), m2.getProdi(), m2.getIpk(), m2.isStatus()));
118     }
119 }
120
121
122
123
124
125
126

```

Setelah sesi praktikum SELESAI, laporan praktikum harus dikirim/diupload ke google classroom pada hari yang ditentukan.