

MODUL 1

FUNDAMENTAL OOP

TUJUAN :

1. Mengerti konsep dasar *object oriented programming*,
2. Dapat membandingkan antara pemrograman prosedural dengan pemrograman berorientasi object,
3. Dapat menerapkan *object oriented programming* pada program yang dibuat.

DASAR TEORI :

Object Oriented Programming (OOP) atau Pemrograman Berorientasi Objek adalah suatu metode yang digunakan dalam pemrograman yang menyelesaikan masalah program dengan menyediakan objek-objek (terdiri dari *attribute* dan *method*) yang saling berkaitan dan disusun kedalam sebuah kelas (*class*).

Pada modul ini kita akan membahas Fundamental OOP, yaitu :

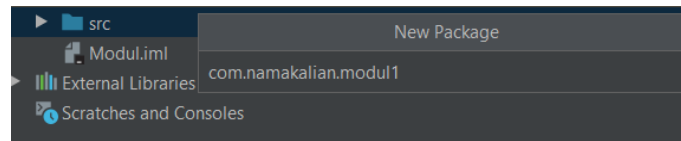
1. Class
2. Object
3. Method
4. Constructor

CLASS

Class adalah cetak biru (rancangan) dari objek. Ini berarti kita bisa membuat banyak objek dari satu macam class. Kelas mendefinisikan sebuah tipe dari objek. Di dalam Class kita dapat mendeklarasikan variabel dan menciptakan object (instansiasi). Sebuah Class mempunyai anggota (member) yang terdiri atas atribut dan method. Atribut adalah semua field identitas yang kita berikan pada suatu Class, misal Class manusia memiliki field atribut berupa nama, maupun umur. Method dapat kita artikan sebagai semua fungsi ataupun prosedur yang merupakan perilaku (behaviour) dari suatu Class. Dikatakan fungsi bila method tersebut melakukan suatu proses dan mengembalikan suatu nilai (return value), dan dikatakan prosedur bila method tersebut hanya melakukan suatu proses dan tidak mengembalikan nilai (void).

CODELAB CLASS

Buatlah project baru dan buatlah sebuah package baru sesuai format dibawah ini :



Lalu buatlah sebuah kelas baru dengan nama “Kelas” dan tulis kode program seperti dibawah ini :

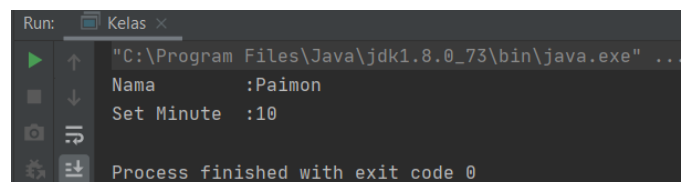
```
/*
 * Code was written by Bagus Bayu Sasongko
 */
public class Kelas {

    private static String nama = "Paimon";

    private static int SetMinute(){
        int minute = 10;
        return minute;
    }

    public static void main(String[] args) {
        System.out.println("Nama\t\t:" + nama);
        System.out.println("Set Minute\t:" + SetMinute());
    }
}
```

Setelah melengkapinya lalu jalankan programnya, maka output dari program tersebut akan seperti ini.



OBJECT

Object (objek) secara lugas dapat diartikan sebagai instansiasi atau hasil ciptaan dari suatu kelas yang telah dibuat sebelumnya. Dalam pengembangan program orientasi objek lebih lanjut, sebuah objek dapat dimungkinkan terdiri atas objek-objek lain. Seperti halnya objek mobil terdiri atas mesin, ban, kerangka mobil, pintu, karoseri dan lain-lain. Atau, bisa jadi sebuah objek merupakan turunan dari objek lain sehingga mewarisi sifat-sifat induknya. Misal motor dan mobil merupakan kendaraan bermotor, sehingga motor dan mobil mempunyai sifat-sifat yang dimiliki oleh kelas kendaraan bermotor dengan spesifikasi sifat-sifat tambahan sendiri. Contoh object yang lain : Komputer, TV, mahasiswa, ponsel, lbuku, dan lainnya.

CODELAB OBJECT

Lalu buatlah sebuah kelas baru dengan nama “Objek” dan tulis kode program seperti dibawah ini :

```
/*
 * Code was written by Bagus Bayu Sasongko
 */

public class Objek {
    public static void main(String[] args) {
        //Deklarasi object tanpa parameter
        Mahasiswa mahasiswaTanpa = new Mahasiswa();
        mahasiswaTanpa.setName("Bagus");
        mahasiswaTanpa.setNim(18104005);

        //Output
        System.out.println("Mahasiswa 1");
        System.out.println("Nama\t: " + mahasiswaTanpa.getName());
        System.out.println("Nim\t: " + mahasiswaTanpa.getNim());

        //Deklarasi object dengan parameter
        Mahasiswa mahasiswa = new Mahasiswa("Paimon", 803303533);

        //Output
        System.out.println("Mahasiswa 2");
        System.out.println("Nama\t: " + mahasiswa.getName());
        System.out.println("Nim\t: " + mahasiswa.getNim());
    }
}
```

Lalu buatlah sebuah class baru dengan nama “Mahasiswa”, dan tulis kode program seperti dibawah ini.

```
/*
 * Code was written by Bagus Bayu Sasongko
 */

public class Mahasiswa{
    String nama;
    int nim;

    public Mahasiswa() {}

    public Mahasiswa(String nama, int nim) {
        this.nama = nama;
        this.nim = nim;
    }

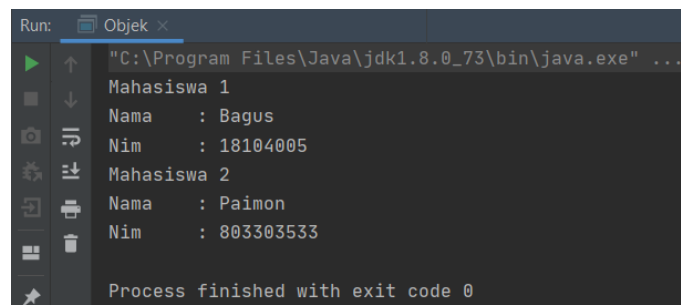
    public String getNama() {
        return nama;
    }

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

    public int getNim() {
        return nim;
    }

    public void setNim(int nim) {
        this.nim = nim;
    }
}
```

Setelah melengkapinya lalu jalankan programnya, maka output dari program tersebut akan seperti ini.



```
Run: Objek x
"C:\Program Files\Java\jdk1.8.0_73\bin\java.exe" ...
Mahasiswa 1
Nama : Bagus
Nim : 18104005
Mahasiswa 2
Nama : Paimon
Nim : 803303533
Process finished with exit code 0
```

METHOD

Method dikenal juga sebagai suatu function dan procedure. Dalam OOP, method digunakan untuk memodularisasi program melalui pemisahan tugas dalam suatu kelas. Pemanggilan method menspesifikasikan nama method dan menyediakan informasi (parameter) yang diperlukan untuk melaksanakan tugasnya. Ada dua cara melewati argumen ke method, yaitu:

1. Melewatkan secara Nilai (Pass by Value)

Digunakan untuk argumen yang mempunyai tipe data primitif (byte, short, int, long, float, double, char, dan boolean). Prosesnya adalah compiler hanya menyalin isi memori (pengalokasian suatu variable), dan kemudian menyampaikan salinan tersebut kepada method. Isi memory ini merupakan data “Sesungguhnya” yang akan dioperasikan. Karena hanya berupa salinan isi memory, maka perubahan yang terjadi pada variable akibat proses di dalam method tidak akan berpengaruh pada nilai variable asalnya.

2. Melewatkan secara Referensi (Pass by Reference)

Digunakan pada array dan objek. Prosesnya isi memory pada variable array dan objek merupakan penunjuk ke alamat memory yang mengandung data sesungguhnya yang akan dioperasikan. Dengan kata lain, variable array atau objek menyimpan alamat memory bukan isi memory. Akibatnya, setiap perubahan variable di dalam method akan mempengaruhi nilai pada variable asalnya.

CODELAB METHOD

Buatlah sebuah class baru dengan nama “TestPass”, dan tulis kode program seperti dibawah ini.

```
/*
 * Code was written by Bagus Bayu Sasongko
 */
public class TestPass {
    int nomer1, nomer2;

    public TestPass(int nomer1, int nomer2) {
        this.nomer1 = nomer1;
        this.nomer2 = nomer2;
    }

    //Passed by value
    void calculate (int m, int n){
        m = m * 10;
        n = n / 2;
    }

    //Passed by reference
    void calculate (TestPass pass){
        pass.nomer1 = pass.nomer1 * 10;
        pass.nomer2 = pass.nomer2 / 2;
    }
}
```

Lalu buatlah sebuah class baru dengan nama “Passed”, dan tulis kode program seperti dibawah ini.

```
/*
 * Code was written by Bagus Bayu Sasongko
 */

public class Passed {
    public static void main(String[] args) {
        int nomer1, nomer2;

        TestPass pass = new TestPass(50,100);
        nomer1 = 10;
        nomer2 = 20;

        //Passed by value
        System.out.println("Nilai sebelum passed by value\t: ");
        System.out.println("Nomer1\t: " + nomer1);
        System.out.println("Nomer2\t: " + nomer2);

        pass.calculate(nomer1, nomer2);
        System.out.println("Nilai setelah passed by value\t: ");
        System.out.println("Nomer1\t: " + nomer1);
        System.out.println("Nomer2\t: " + nomer2);
        System.out.println();

        //Passed by reference
        System.out.println("Nilai sebelum passed by reference\t: ");
        System.out.println("passed.nomer1\t: " + pass.nomer1);
        System.out.println("passed.nomer2\t: " + pass.nomer2);

        pass.calculate(pass);
        System.out.println("Nilai sesudah passed by reference\t: ");
        System.out.println("passed.nomer1\t: " + pass.nomer1);
        System.out.println("passed.nomer2\t: " + pass.nomer2);
    }
}
```

Setelah melengkapinya lalu jalankan programnya, maka output dari program tersebut akan seperti ini.

```
Run: Passed x
"C:\Program Files\Java\jdk1.8.0_73\bin\java.exe" ...
Nilai sebelum passed by value :
x = 10
y = 20
Nilai sesudah passed by value :
x = 10
y = 20
Nilai sebelum passed by reference :
z.i = 50
z.j = 100
Nilai sesudah passed by reference :
z.i = 500
z.j = 50
Process finished with exit code 0
```

Pada saat pemanggilan method “calculate()” dengan metode pass by value, hanya nilai dari variable x dan y saja yang dilewatkan ke variable m dan n , sehingga perubahan pada variable m dan n tidak akan mengubah nilai dari variable x dan y . Sedangkan pada saat pemanggilan method “calculate()” dengan metode *pass by reference* yang menerima parameter bertipe kelas Test. Pada waktu kita memanggil method “calculate()”, nilai dari variable z yang berupa referensi ke obyek sesungguhnya dilewatkan ke variable a ,

sehingga variable *a* menunjukkan ke obyek yang sama dengan yang ditunjuk oleh variable *z* dan setiap perubahan pada objek tersebut dengan menggunakan variable *a* akan terlihat efeknya pada variable *z* yang terdapat pada kode yang memanggil method tersebut.

CONSTRUCTOR

Constructor adalah tipe khusus method yang digunakan untuk menginstansiasi atau menciptakan sebuah objek. Nama constructor adalah sama dengan nama kelasnya. Selain itu, constructor tidak bisa mengembalikan suatu nilai (not return value) bahkan void sekalipun. Defaultnya, bila kita tidak membuat constructor secara eksplisit, maka Java akan menambahkan constructor default pada program yang kita buat secara implisit. Constructor default ini tidak memiliki parameter masukan sama sekali. Namun, bila kita telah mendefinisikan minimal satu buah constructor, maka Java tidak akan menambah constructor default. Constructor juga dimanfaatkan untuk membangun suatu objek dengan langsung mengeset atribut-atribut yang disandang pada objek yang dibuat tersebut. Oleh karena itu, constructor jenis ini haruslah memiliki parameter masukan yang akan digunakan untuk mengeset nilai atribut. Access modifier yang dipakai pada constructor selayaknya adalah public karena constructor tersebut akan diakses di luar kelasnya (walaupun kita juga bisa memberikan access modifier pada constructor dengan private—artinya kita tidak bisa memanggil constructor tersebut di luar kelasnya). Cara memanggil constructor adalah dengan menambahkan keyword new. Keyword new dalam deklarasi ini artinya kita mengalokasikan pada memory sekian blok memory untuk menampung objek yang baru kita buat.

CODELAB CONSTRUCTOR

Buatlah sebuah class baru dengan nama “Manusia”, dan tulis kode program seperti dibawah ini.

```
/*
 * Code was written by Bagus Bayu Sasongko
 */

public class Manusia {
    private String nama;
    private int umur;

    //Definisi Constructor
    //Constructor pertama = default tanpa parameter
    public Manusia(){}

    //Constructor kedua
    public Manusia(String nama){
        this.nama = nama;
    }

    //Constructor ketiga
    public Manusia(String nama, int umur){
        this.nama = nama;
        this.umur = umur;
    }
}
```

```
//Definisi method
public String getName() {
    return nama;
}

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

public int getUmur() {
    return umur;
}

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

Lalu buatlah sebuah class baru dengan nama “DemoManusia”, dan tulis kode program seperti dibawah ini.

```
/*
 * Code was written by Bagus Bayu Sasongko
 */

public class DemoManusia {
    public static void main(String[] args) {
        Manusia arrayManusia[] = new Manusia[3];

        Manusia manusia1 = new Manusia(); //Constructor Pertama
        manusia1.setName("Jean");
        manusia1.setUmur(20);

        Manusia manusia2 = new Manusia("Fischl"); //Constructor Kedua
        Manusia manusia3 = new Manusia("Barbara",18); //Constructor Ketiga

        arrayManusia[0] = manusia1;
        arrayManusia[1] = manusia2;
        arrayManusia[2] = manusia3;

        for (Manusia x : arrayManusia){
            System.out.println("Character ");
            System.out.println("Nama\t: " + x.getName());
            System.out.println("Umur\t: " + x.getUmur());
        }
    }
}
```

Setelah melengkapinya lalu jalankan programnya, maka output dari program tersebut akan seperti ini.

```
Run: DemoManusia x
"C:\Program Files\Java\jdk1.8.0_73\bin\java.exe" ...
Character
Nama : Jean
Umur : 20
Character
Nama : Fischl
Umur : 0
Character
Nama : Barbara
Umur : 18

Process finished with exit code 0
```


LATIHAN

1. Buatlah program menghitung konversi suhu, $C \rightarrow F$; $F \rightarrow K$; $K \rightarrow R$; $R \rightarrow C$ (Menggunakan Method)
2. Buatlah program Pendaftaran Mahasiswa Baru (Menggunakan Constructor)
3. Buatlah program detail Karyawan (Menggunakan Kelas)