Object Oriented Programming

Week-01 Pengantar *Object Oriented Programming*

YUDA SYAHIDIN
(yudasy@gmail.com)

Kuliah OOP

Perkuliahan 14 kali pertemuan (UTS & UAS)

Penilaian :

- Keaktifan : 10 %
- Quiz + Tugas : 20 %
- UTS : 30 %
- UAS : 40%

Tugas akhir Semester (Proyek OOP)

Week-01: Pengantar OOP

Materi OOP

- Pengantar Object Oriented Programming
- Struktur Object Oriented (Class, Attribut,Method)
- Constructor
- Inheritance & Overloading
- Class Abstract
- Class Interface

Referensi

- Object Oriented Programming using Java Anban Pillay, Shcool of Computer Science, University of KwaZulu-Natal Durban, Februari 2207
- Poo, Danny. Kiong, Derek. Ashok, Swarnalatha. "Object Oriented Programming and Java, second Edition", 2008. Springer.
- The Java Reference Library, 1996,1997. O'Reillly & Associates
- J.E.N.I Pengenalan Pemrograman 1 Versi 1.2 juni 2007, Joyce Avestro, JARDIKNAS
- Budi Raharjo, Imam Heryanto, Arif Haryono., “Mudah Belajar Java”, Informatika 200

Apa OOP?

- Bahasa Java merupakan bahasa pemrograman yang berorientasi objek. Dengan menggunakan bahasa berorientasi objek, kita sebagai *programmer* dapat mengimplementasikan perancangan berorientasi objek (i.e. UML).
- Pada dasarnya, pemrograman berorientasi objek merupakan jenis pemrograman yang berdasarkan kepada suatu prinsip bahwa semua program sebenarnya merupakan simulasi berbasis komputer dari berbagai objek atau konsep yang ada di dunia nyata.
- Sebagai contoh, program-program yang digunakan untuk keperluan suatu perusahaan atau bisnis dapat dipandang sebagai sebuah simulasi dari proses-proses yang terjadi pada perusahaan atau bisnis itu sendiri, misalnya proses pemesanan, layanan konsumen, pengiriman, dan *billing*.

Kelas

- Kelas (class) adalah unit pemrograman yang membentuk suatu aplikasi berbasis Java.
- Sebuah kelas terdiri dari *methods* dan *atribut*. Suatu kelas juga dapat berisi kelas lain (*inner class*).
- **Sintaks Dasar Pembuatan Kelas:**

```
class NamaKelas {
  // deklarasi atribut (variabel)
  // methods
}
```

- **Exercise 01: PersegiPanjang.java**

```
class PersegiPanjang {
  int panjang;
  int lebar;

  int hitungLuas(){
    int area = panjang * lebar;
    return area;
  }
}
```

Kelas

- Kelas "Persegi Panjang" di atas memiliki:
  - 2 atribut (variabel) yaitu "panjang" dan "lebar",
  - 1 methods yaitu "hitungLuas()".
- **Atribut** adalah suatu lokasi di memori yang diberi nama yang berfungsi untuk menyimpan suatu nilai dengan tipe tertentu.
  - Misalnya, kelas "PersegiPanjang" tersebut memiliki dua buah atribut (variabel) yaitu "panjang" dan "lebar" yang bertipe integer.
- **Method** merupakan suatu aksi atau perilaku tertentu yang dapat dipanggil untuk melakukan suatu fungsi atau tindakan tertentu.
  - Misalnya, kelas "PersegiPanjang" memiliki sebuah method yang bernama "hitungLuas" yang berfungsi mengalikan nilai panjang dan lebar untuk menghasilkan luas dari persegi panjang tersebut.

Constructor

- Biasanya, setiap kelas memiliki suatu method khusus yang disebut dengan nama "constructor". Method ini disebut khusus karena:
  - memiliki nama yang sama dengan nama kelasnya, serta
  - tidak memiliki *return* type.
  - Method ini akan dipanggil setiap kali kita membuat objek dari kelas tersebut.
- **Exercise 02: PersegiPanjang2.java (class + constructor-nya)**

```
class PersegiPanjang2 {
   int panjang;
   int lebar;

   public PersegiPanjang2 (int pj, int lb){ //constructor
        panjang = pj;
        lebar = lb;
   }

   int hitungLuas(){
        int luas = panjang * lebar;
        return luas;
   }
}
```

Proses membuat objek dari suatu kelas: declare-instantiate-bind

- Langkah-langkah pembuatan suatu objek di OOP dikenal dengan:

  Declare → Instantiate → Bind

- Declare:

```
PersegiPanjang p1;
```

- Instantiate:

```
new PersegiPanjang( );
```

- Bind:

```
=
```

- Declare-instantiate-bind:

```
PersegiPanjang p1 = new PersegiPanjang ( );
```

Proses membuat objek dari suatu kelas: declare-instantiate-bind

- Berdasarkan kelas yang telah dibuat tersebut, kita dapat membuat objek dengan cara melakukan proses intansiasi yaitu menggunakan perintah "new" disertai dengan nama *construtor* yang telah didefinisikan sebelumnya.
- Perhatikan sekali lagi bahwa nama *constructor* sama dengan nama kelas.

- **Sintaks:** NamaKelas namaObjek = new Nama_Kelas_atau_Constructor();

- **Exercise 03: HitungLuasPP.java**

```
public class HitungLuasPP {
   public static void main(String[] args){
      // Instansiasi objek:
      PersegiPanjang2 p = new PersegiPanjang2 (2, 5);
      // Panggil method "hitungLuas" dari objek tsb:
      int luas = p.hitungLuas();
      System.out.println("Luas Persegi Panjang = " + luas);
   }
}
```

That's all for today, folks!

- Congratulations!
- You've learnt:
  - Kelas
  - Field/Atribut/Variabel
  - Method
  - Instansiasi