UNIVERSITAS PGRI WIRANEGARA PASURUAN

Pemrograman Berorientasi Objek

KODE MK: KOMP13

SKS: 3

DOSEN PENGAMPU: RIZDANIA, ST. MKOM

Contact me

Rizdania (Nia)

Email : rizdania.uniwara@gmail.com

WA (chat only) : 08123363079

(Monday-Friday, 08:00 – 17:00)

Bobot Penilaian

Kuis	: 15 %
Tugas	: 15 %
UTS	: 25 %
UAS	: 25 %
Attitude	: 20 %
==========	
	100 %

Rules

- Berpakaian rapi dan sopan
- Dilarang merokok selama kuliah berlangsung
- Tidak ada toleransi kecurangan dalam bentuk apapun
- Attitude
- Toleransi waktu: 15 menit
- Wajib mengaktifkan kamera saat virtual meeting

Konsep Dasar OOP

PEMROGRAMAN BERORIENTASI OBJEK

Mata Kuliah Pemrograman Berorientasi Objek

Teori Pemrograman Lanjut (3 SKS)

Praktikum Pemrograman Lanjut (1 SKS)

Capaian Pembelajaran

- Mampu membuat program dengan menggunakan prinsip-prinsip OOP menggunakan bahasa pemrograman Java

Software :

- JDK
- Netbeans

Silabus

- Pengantar Konsep Dasar OOP
- Class dan Object
- Enkapsulasi
- Relasi Class
- Inheritance
- Overriding dan Overloading
- Abstract Class
- Interface
- Polimorfisme
- Unit Testing

Referensi

- Y. Daniel Liang. 2015. Introduction to Java Programming, Comprehensive Version, 10th Edition. Prentice Hall
- https://docs.oracle.com/javase/tutorial/
- Horstmann, C. S., & Cornell, G. (2007). *Core Java Volume I–Fundamentals, Eighth Edition.* Network Circle, Santa Clara: Prentice Hall.
- Horstmann, C. S., & Cornell, G. (2008). *Core Java Volume II–Advanced Features, Eighth Edition.* Network Circle, Santa Clara: Prentice Hall.
- https://www.javatpoint.com/java-oops-concepts

Dapat di download di http://libgen.io/

Objek Oriented vs Struktural

Struktural

- Program dipecah kedalam fungsi
- Perubahan fitur → kemungkinan mengganggu keseluruhan program

Object Oriented

- Program dipecah kedalam object
  - Didalamnya terdapat state dan behavior
- Perubahan fitur → tidak mengganggu keseluruhan program

Objek Oriented vs Struktural

Contoh:

Kita akan membuat program game simulasi sepeda, didalamnya ada karakter sepeda yang memiliki kecepatan, gear dan merk.

Bagaimana membangun game tersebut dengan metode **konvensional**?

- Langkah pertama kita buat variabelnya, misal kecepatan, gear, merk
- Langkah berikutnya kita buat fungsi-fungsinya, tambah kecepatan, kurangi kecepatan.
- Langkah berikutnya kita coba mengoperasikan sepeda tersebut secara sederhana, yaitu memanipulasi kecepatan, gear, merk nya, didalam fungsi main, kemudian kita cetak ke layar.

Kode program →…

```java
public class SepedaStruktural
{
    public static void main(String[] args)
    {
        String merek;
        int kecepatan, gear;

        merek = "Poligone";
        kecepatan = 10;
        gear = 1;

        kecepatan = tambahKecepatan(kecepatan, 10);

        System.out.println("Merek: " + merek);
        System.out.println("Kecepatan: " + kecepatan);
    }

    public static int tambahKecepatan(int kecepatan, int increment)
    {
        kecepatan += increment;

        return kecepatan;
    }

    public static int kurangiKecepatan(int kecepatan, int decrement)
    {
        kecepatan -= decrement;

        return kecepatan;
    }
}
```

Objek Oriented vs Struktural

Bagaimana jika ada dua sepeda di game?

- Tambahkan variabel merek2, kecepatan2, gear2
- Coba manipulasi nilai-nilai variabelnya kemudian tampilkan ke layar

Kode program → …

```java
public class SepedaStruktural
{
    public static void main(String[] args)
    {
        String merek, merek2;
        int kecepatan, kecepatan2, gear, gear2;

        merek = "Poligone";
        kecepatan = 10;
        gear = 1;

        merek2 = "Wiim Cycle";
        kecepatan2 = 15;
        gear2 = 3;

        kecepatan = tambahKecepatan(kecepatan, 10);
        kecepatan2 = tambahKecepatan(kecepatan2, 5);

        System.out.println("Merek: " + merek);
        System.out.println("Kecepatan: " + kecepatan);

        System.out.println("Merek: " + merek2);
        System.out.println("Kecepatan: " + kecepatan2);
    }

    public static int tambahKecepatan(int kecepatan, int increment)
    {
        kecepatan += increment;

        return kecepatan;
    }

    public static int kurangiKecepatan(int kecepatan, int decrement)
    {
        kecepatan -= decrement;

        return kecepatan;
    }
}
```

Objek Oriented vs Struktural

Bagaimana jika ada **sepuluh** sepeda?

- Tambahkan variabel merek3, kecepatan3, gear3 …… merek9, kecepatan9, gear9
- Cukup melelahkan…

Bagaimana dengan object oriented?

- Buat sebuah class Sepeda yang memiliki atribut merek, kecepatan, gear.
- Buat 10 object sepeda

Kode program → …

```java
public class Sepeda
{
    private String merek;
    private int kecepatan;
    private int gear;

    public Sepeda(String newMerek, int newKecepatan, int newGear)
    {
        merek = newMerek;
        kecepatan = newKecepatan;
        gear = newGear;
    }

    public void tambahKecepatan(int increment)
    {
        kecepatan += increment;
    }

    public void kurangiKecepatan(int decrement)
    {
        kecepatan -= decrement;
    }

    public void cetakStatus()
    {
        System.out.println("Merek: " + merek);
        System.out.println("Kecepatan: " + kecepatan);
        System.out.println("Gear: " + gear);
    }
}
```

```java
public class SepedaOOP
{
    public static void main(String[] args)
    {
        Sepeda spd1 = new Sepeda("Poligone", 10, 1);
        Sepeda spd2 = new Sepeda("Wiim Cycle", 15, 3);

        spd1.tambahKecepatan(10);
        spd2.tambahKecepatan(5);

        spd1.cetakStatus();
        spd2.cetakStatus();
    }

}
```

Objek Oriented vs Struktural

Dapat kita lihat bahwa dengan OOP, untuk membuat banyak sepeda, kita tidak perlu tuliskan berulang-ulang variabel merek, kecepatan, dan gear.

Kita cukup buat banyak objek sepeda saja, dari sebuah class sepeda yang sudah kita buat.

Konsep OOP

Beberapa aspek dalam OOP:

- Object
- Class
- Enkapsulasi
- Inheritance
- Polimorfisme

Object

Object adalah suatu rangkaian dalam program yang terdiri dari **state** dan **behaviour**.

Object pada software dimodelkan sedemikian rupa sehingga mirip dengan objek yang ada di dunia nyata.

Objek memiliki state dan behaviour.

State adalah ciri-ciri atau atribut dari objek tersebut.

- Misal objek Sepeda, memiliki state **merek, kecepatan, gear** dan sebagainya.

Behaviour adalah perilaku yang dapat dilakukan objek tersebut.

- Misal pada Sepeda, behaviournya antara lain, **tambah kecepatan, pindah gear, kurangi kecepatan, belok**, dan sebagainya.

Class

Class adalah blueprint atau prototype dari objek.

Ambil contoh objek sepeda.

- Terdapat berbagai macam sepeda di dunia, dari berbagai merk dan model.
- Namun semua sepeda dibangun berdasarkan blueprint yang sama, sehingga tiap sepeda memiliki komponen dan karakteristik yang sama.

Sepeda yang anda miliki dirumah, adalah hasil **instansiasi** dari **class** sepeda.

Class Sepeda

Object Sepeda

Inheritance

Memungkinkan kita untuk mengorganisir struktur program dengan natural.

Memperluas fungsionalitas program tanpa harus mengubah banyak bagian program.

Contoh di dunia nyata:

- Objek sepeda dapat diturunkan lagi ke model yang lebih luas, misal sepeda gunung (mountain bike) dan city bike.
- Masing-masing dapat memiliki komponen/fitur tambahan, misal sepeda gunung memiliki **suspensi**, yang tidak dimiliki sepeda biasa. Dan city bike memiliki keranjang di bagian depannya.
- Dalam hal ini, objek mountain bike dan road bike **mewarisi** objek sepeda.

Polimorfisme

Polimorfisme juga meniru sifat objek di dunia nyata, dimana sebuah objek dapat memiliki bentuk

- Atau menjelma menjadi bentuk-bentuk lain.

Misalkan saja objek pesawat terbang.

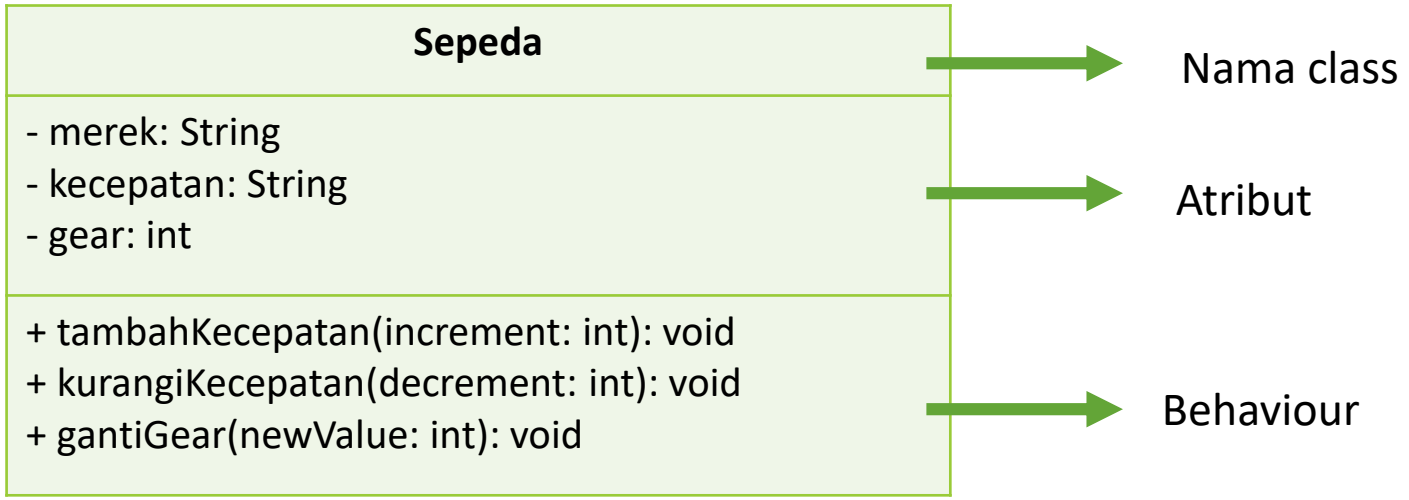
- Objek ini dapat diwariskan menjadi pesawat jet dan pesawat baling-baling.
- Keduanya memiliki kemampuan untuk menambah kecepatan.
- Namun secara teknis, metode penambahan kecepatan antara pesawat jet dengan baling-baling tentu berbeda, karena masing-masing memiliki jenis mesin yang berbeda.

UML Class Diagram

Dalam pemrograman berorientasi objek, rancangan class digambarkan dengan UML Class Diagram

- UML adalah singkatan dari Unified Modelling Language

Misal class Sepeda, yang memiliki state **merek, kecepatan, gear** dan behavior **tambahKecepatan, kurangiKecepatan, gantiGear** digambarkan dengan class diagram sebagai berikut:



Latihan

Carilah objek apa saja di dunia nyata sebanyak 5 (lima) macam.

Tuliskan state dan behavior objek tersebut. Makin banyak state dan behavior makin baik. Contoh:

- State:
  - Merek
  - Ukuran layar
  - Channel
  - Volume
- Behavior:
  - Nyalakan
  - Matikan
  - Pindah channel
  - Tambah volume
  - Kurangi volume

Terima Kasih