Modul Praktikum

Pemrograman Berorientasi Objek

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Lampung

2022
Semester Ganjil

Modul 7. Method

Method adalah sebuah blok program yang hanya akan dijalankan ketika dipanggil. Anda dapat mengirim data, yang dikenal sebagai parameter, ke dalam method. Method digunakan untuk melakukan tugas tertentu, dalam berbagai sumber method juga disebut sebagai fungsi. Anda dapat menggunakan method untuk menggunakan kembali kode yang sudah dibuat.

Capaian Pembelajaran

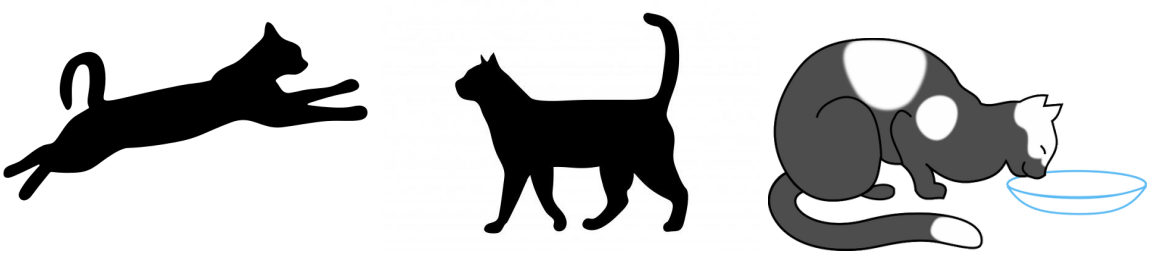
1. Mahasiswa mampu membuat method dalam sebuah class
2. Mahasiswa mampu mengirimkan nilai ke dalam method melalui parameter
3. Mahasiswa mampu membuat class yang menerapkan method overloading
4. Mahasiswa mampu membuat class yang memiliki constructor, setter dan getter
5. Mahasiswa mampu memanggil dan menggunakan method yang telah ada

Materi

Konsep Method

Method dalam sebuah class merupakan implementasi Behavior dari sebuah Objek. Yaitu apa saja yang bisa dilakukan oleh sebuah objek.

Sebagai contoh dalam kehidupan sehari-hari seekor Kucing memiliki warna mata, warna bulu dan jenis yang berbeda-beda. Kucing juga bisa mengeong, melompat, berjalan, makan, dan lain-lain.



Behavior Kucing

Dalam kasus tersebut apabila kita implementasikan ke dalam sebuah class, Warna mata, warna bulu dan jenis merupakan atribut dari class Kucing. Sedangkan mengeong, melompat, berjalan, makan, dan lainnya merupakan behaviour atau method.

Contoh lainnya adalah misal jika kita membuat sebuah class Lingkaran. Lingkaran memiliki jari-jari dan diameter. Selain itu, kita dapat menghitung Luas dan Keliling dari sebuah lingkaran. Dalam kasus ini jari-jari dan diameter dapat dijadikan Atribut dari class Lingkaran. Sedangkan Luas dan Keliling merupakan Behavior atau Method di Class Lingkaran.

Membuat Method

Sebuah method harus dideklarasikan di dalam sebuah kelas. Method didefinisikan dengan nama method, diikuti oleh tanda kurung ().

Java menyediakan beberapa predefined method, seperti System.out.println(), tetapi Anda juga dapat membuat metode sendiri untuk melakukan hal tertentu:

```
public class Main {
  accessModifier dataType namaMethod() {
    // code to be executed
```

```
  }
}
```

Memanggil Method

Untuk memanggil method di Java, tulis nama method diikuti oleh dua tanda kurung () dan titik koma. Untuk memanggil method yang ada di dalam sebuah objek, gunakan dot operator (.) sama seperti saat anda memanggil variabel.

Dalam contoh berikut, terdapat class Kucing yang memiliki method meong() digunakan untuk mencetak teks “Miaw!” dimana method meong() dapat dianggap sebagai behavior dari objek Kucing:

```
class Kucing {
  public void meong() {
    System.out.println("Miaw!");
  }
}

public class Main{
  public static void main(String[] args) {
  /*
      Langkah-langkah memanggil method:
      - Instansiasi objek
      - Panggil method dengan menggunakan dot operator (.)
  */
    Kucing moza = new Kucing() //Instansiasi objek
    moza.meong();              //Memanggil method meong di objek kucing
  }
}
```

Parameters and Arguments

Data dapat diteruskan ke method sebagai parameter. Parameter bertindak sebagai variabel di dalam method.

Parameter ditentukan setelah nama method, di dalam tanda kurung. Anda dapat menambahkan parameter sebanyak yang Anda inginkan, cukup pisahkan dengan koma.

Contoh berikut memiliki metode yang mengambil String yang disebut nama sebagai parameter. Saat method dipanggil, anda dapat menggunakan variabelnya untuk mencetak string yang diinginkan:

```
class Kucing {
  public void meong(String nama) {
    System.out.println("Miaw " + nama);
  }
}

public class Main{
  public static void main(String[] args) {
    Kucing moza = new Kucing() //Instansiasi objek
    moza.meong("Ervan");       //Memanggil method dengan parameter
  }
}
// Output : Miaw Ervan
```

Return Values

Kata 'void', yang digunakan dalam contoh di atas, menunjukkan bahwa method tidak boleh mengembalikan nilai. Jika Anda ingin method mengembalikan nilai, Anda dapat menggunakan tipe data primitif (seperti int, char, dll.) alih-alih void, dan gunakan kata kunci return di dalam method:

```
class Segiempat {
      public int luas(int sisi) {
      return sisi * sisi;
      }
}

public class Main {
  public static void main(String[] args) {
      Segiempat s = new Segiempat();
      System.out.println(s.luas(4));
  }
}

// Output : 16
```

Method Overloading

Dengan method overloading, beberapa method dapat memiliki nama yang sama dengan parameter atau return type yang berbeda:

```
class Segiempat {
    public int luas(int sisi) {
      return sisi * sisi;
      }
    public double luas(double panjang, double lebar){
      return panjang * lebar;
    }
}

public class Main {
  public static void main(String[] args) {
      Segiempat s = new Segiempat();
      System.out.println(s.luas(4));
      System.out.println(s.luas(4.5, 2));
  }
}
```

Output:

```
16
9.0
```

Java Constructors

Constructor adalah method khusus yang akan dieksekusi pada saat pembuatan objek (instance). Biasanya method ini digunakan untuk inisialisasi atau mempersiapkan data untuk objek.

Saat membuat sebuah objek dari class, constructor akan otomatis dipanggil untuk menyiapkan data-data pada objek tersebut.

Getter and Setter

Dari materi sebelumnya bahwa variabel private hanya dapat diakses di dalam kelas yang sama (kelas luar tidak memiliki akses ke sana). Namun, dimungkinkan untuk mengaksesnya jika anda menyediakan method getter dan setter yang bersifat public.

Method get mengembalikan nilai variabel, dan metode set menetapkan nilai.

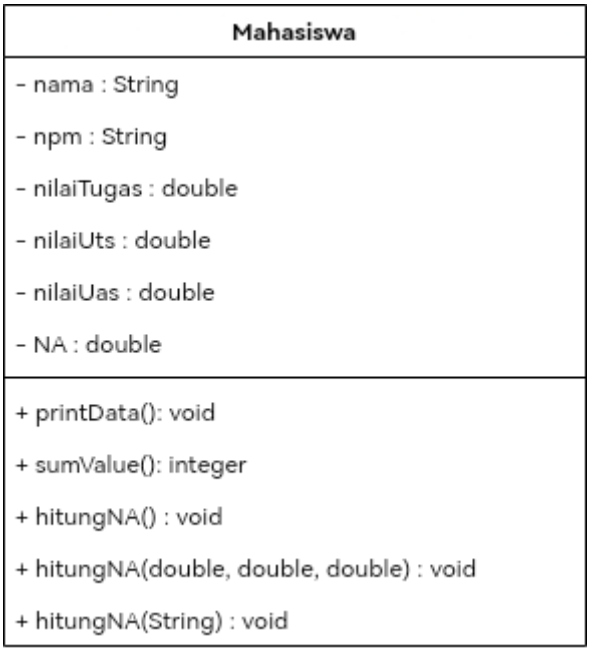
Langkah Praktikum

Membuat Project

1. Buka Netbeans
2. Buat Project dengan nama **nama_Praktikum7**

Studi Kasus

Pada pertemuan ini anda akan diminta untuk membuat sebuah class Mahasiswa yang memiliki atribut **NPM, Nama, NIlaiTugas, NilaiUTS, NilaiUAS** dan **NA**. Selain itu dalam class tersebut akan terdapat beberapa method seperti **printData()**, **hitungNA()**, **Constructor, Setter** dan **Getter**.



Membuat Class Mahasiswa

1. Buatlah class Mahasiswa yang memiliki atribut **npm, nama, nilaiTugas, nilaiUAS,** dan **nilaiUTS**.

```
public class Mahasiswa {
    private String nama;
    private String npm;
    private double nilaiTugas;
    private double nilaiUas;
    private double nilaiUts;
    private double NA;
}
```

2. Kemudian buat sebuah object dari class Mahasiswa di class utama.

```
public static void main(String[] args) {
    Mahasiswa mhs = new Mahasiswa();
}
```

Membuat Method

1. Buatlah sebuah method **printData** pada class **Mahasiswa**. Method ini digunakan untuk menampilkan data mahasiswa.

```
public void printData(){
    System.out.println("NPM\t:" + this.npm);
    System.out.println("Nama\t:" + this.nama);
}
```

Panggil method tersebut di class utama untuk menampilkan hasilnya.

2. Selanjutnya buatlah sebuah method **sumValue** dalam class Mahasiswa. Method ini memiliki return value **integer**. Method ini mengembalikan hasil dari penjumlahan dua buah nilai.

```
public int sumValue(){
    return 5+3;
}
```

Panggil method tersebut dan simpan nilainya ke dalam sebuah variabel

atau langsung cetak nilainya.

```
public static void main(String[] args) {
    Mahasiswa mhs = new Mahasiswa();

    int result = mhs.sumValue();

    System.out.println(result);
}
```

3. Modifikasi method **printData**. Tambahkan sebuah parameter **String jurusan** pada method tersebut. Kemudian tampilkan nilainya.

```
public void printData(String jurusan){
    System.out.println("NPM\t: " + this.npm);
    System.out.println("Nama\t: " + this.nama);
    System.out.println("Jurusan\t: " + jurusan);
}
```

Panggil method tersebut di fungsi main dan jangan lupa untuk menambahkan parameternya.

```
...
mhs.printData("Ilmu Komputer");
...
```

4. Modifikasi method **sumValue** dengan menambahkan **dua** parameter **int x** dan **y**. Method ini akan mengembalikan hasil penjumlahan nilai **x** dan **y**.

```
public int sumValue(int x, int y){
    return x+y;
}
```

Panggil method tersebut dan jangan lupa untuk menambahkan parameternya. Kemudian simpan atau langsung cetak nilainya.

```
...
int result = mhs.sumValue(5, 8);
System.out.println(result);
...
```

Method Overloading

1. Buatlah method **hitungNA** dalam class Mahasiswa. Method ini akan mengisi nilai dari atribut **NA** berdasarkan hasil penghitungan antara atribut **nilaiTugas, nilaiUTS,** dan **nilaiUAS**, dengan persentase :
   a. Tugas : 30%
   b. UTS : 35%
   c. UAS : 35%

```
public void hitungNA(){
    this.NA = (this.nilaiTugas*0.3) + (this.nilaiUts*0.35) + (this.nilaiUas*0.35);
}
```

2. Buatlah kembali method **hitungNA** dalam class Mahasiswa, kali ini berikan tiga buah parameter dengan tipe data **double** sebagai **nilaiTugas, nilaiUTS,** dan **nilaiUAS**.

```
public void hitungNA(double nilaiTugas, double nilaiUts, double nilaiUas){
    this.NA = (nilaiTugas*0.3) + (nilaiUts*0.35) + (nilaiUas*0.35);
}
```

3. Buat kembali method **hitungNA**. Kali ini berikan **satu** parameter **String pesan**. Kemudian isi statementnya dengan menampilkan pesan **"Nilai tidak Valid"**

```
public void hitungNA(String pesan){
    System.out.println("Nilai tidak valid");
}
```

4. Panggil class tersebut di class main dan coba sesuaikan parameternya.

Constructor

1. Buatlah sebuah **Constructor** pada class Mahasiswa. Constructor ini memiliki 5 Parameter sesuai dengan atribut yang ada pada class

Mahasiswa.

```
public Mahasiswa(
    String nama, String npm, double nilaiTugas,
    double nilaiUas, double nilaiUts
){
    this.nama = nama;
    this.npm = npm;
    this.nilaiTugas = nilaiTugas;
    this.nilaiUas = nilaiUas;
    this.nilaiUts = nilaiUts;
}
```

2. Buatlah sebuah Objek baru dari class Mahasiswa di class utama dengan menggunakan constructor baru. Kemudian panggil method **printData**

```
...
Mahasiswa mhs2 = new Mahasiswa("Ervan Chodry", "2017051001", 80, 85, 88);
mhs2.printData("Ilmu Komputer");
...
```

Setter dan Getter

1. Buatlah **Setter** dari Atribut **Nama**. Setter ini merupakan sebuah method void dengan sebuah parameter untuk mengisi nilai dari atribut nama.

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

2. Kemudian Buat sebuah **Getter** dari atribut **Nama**. Getter merupakan sebuah method yang mengembalikan nilai dari atribut dalam class agar dapat diakses diluar class tersebut.

```
public String getNama() {
    return nama;
}
```

3. Cobalah mengubah nilai dari atribut nama pada class main menggunakan setter. Kemudian panggil nilainya menggunakan getter.

```
...
mhs2.setNama("Bukan Ervan");
System.out.println("Setelah diubah : " + mhs2.getNama());
...
```

4. Buatlah setter dan juga getter untuk atribut lainnya.

Tugas

1. Lanjutkan class Mahasiswa yang telah dibuat.
2. Buatlah sebuah method **hitungTugas** yang memiliki sebuah parameter **array nilai** dengan tipe data **double**. Method ini akan mengisi atribut **nilaiTugas** dengan hasil penghitungan rata-rata dari **array nilai**.
3. Modifikasi method **printData** dengan parameter **String jurusan** yang ada pada class Mahasiswa. Tambahkan code agar method ini dapat menampilkan data mahasiswa beserta seluruh nilai yang dimiliki.
4. Menggunakan konsep Overloading buat kembali method **printData** tanpa parameter. Method ini akan menampilkan data Mahasiswa beserta seluruh nilainya.
5. Submit class Mahasiswa yang telah dibuat melalui Vclass dan ikuti instruksinya.