Praktikum Pemrograman Berbasis Objek

Pertemuan 2

Review Time 

1. Struktur Kode Java
2. Membuat Objek
3. Input dan Output
4. Statements dalam Java
5. Compile dan Run File Java
6. Built-in Methods dalam Class Java
7. Seputar assignment atau lainnya

Materi Pertemuan 2

Gambaran Materi Pembelajaran Hari Ini

Materi Pertemuan 2

01

Program Modular Java

Membuat program modular dengan file java lebih dari satu

02

Array of Object

Penggunaan array of object dalam java

03

Enkapsulasi

Mengenal konsep dan penggunaan enkapsulasi

04

Overloading

Mengenal konsep dan penggunaan overloading

01

Program Modular Java

Modularitas Java

Dalam bahasa pemrograman berorientasi objek apapun termasuk Java dan C#, konsep modularitas menjadi sebuah standar untuk membuat setiap modul bersifat individual. Hal ini mengizinkan komunikasi antar modul yang lebih mudah.

Library yang sering kita sertakan ketika membuat suatu program juga dapat disebut sebagai module. Maka dari itu, modularitas ini sangat berguna terutama untuk bahasa pemrograman berorientasi objek.

Di Java sendiri, terdapat sebuah keyword bernama `package` yang berfungsi untuk mengorganisir sekumpulan `` `class` `` dan `` `interface` `` dalam suatu direktori tertentu.

Package

```java
package child1;

public class File1 {
    String data;

    public File1(String data) {
        this.data = data;
    }

    public String getData() {
        return this.data;
    }

    public void setData(String data) {
        this.data = data;
    }
}
```

← Deklarasi package

← Isi dari class

Package (lanjutan)

```java
package child1;

public class File1 {
    String data;

    public File1(String data) {
        this.data = data;
    }

    public String getData() {
        return this.data;
    }

    public void setData(String data) {
        this.data = data;
    }
}
```

Contoh File1.java

```java
package child2;

import child1.File1;

public class File2 {
    public static void main(String[] args) {
        File1 test = new File1("Tes package!");

        System.out.println(test.getData());
    }
}
```

Contoh File2.java

Package (lanjutan)

Struktur folder dan file nya dapat dilihat di gambar disamping. Workspace yang sedang digunakan bernama `parent`, memiliki folder `child1` dan `child2` dengan isi file yang ada di slide sebelumnya.

Current directory harus berada di folder `parent` dan cara mengcompile nya dapat dilihat dibawah.

```
s_v\asprak\pbo\sandbox\parent
❯ javac .\child2\File2.java; java child2/File2
Tes package!
```

02

Array of Object

Membuat Array dari Object

```java
import child1.File1;

public class Test {
    public static void main(String[] args) {
        File1[] tes = new File1[5];

        for (int i = 0; i < 5; i++) {
            tes[i] = new File1("Ini tes ke-"+i);
        }

        for (File1 f : tes) {
            System.out.println(f.getData());
        }
    }
}
```

← Instansiasi array of object

← Pengisian array of object melalui constructornya

← Print isi dari array of object

Output dari kode →

```
s_v\asprak\pbo\sandbox\parent
❯ javac Test.java; java Test
Ini tes ke-0
Ini tes ke-1
Ini tes ke-2
Ini tes ke-3
Ini tes ke-4
```

03

Enkapsulasi

Access Modifier

Berdasarkan apakah class dapat diakses oleh class lain

	class yang satu package	class beda package	subclass yang satu package	subclass beda package
public	✅	✅	✅	✅
protected	✅	❌	✅	✅
package (default)	✅	❌	✅	❌
private	❌	❌	❌	❌

Enkapsulasi

Enkapsulasi merupakan salah satu konsep fundamental PBO yaitu membungkus suatu kode dan data menjadi satu kesatuan.

Enkapsulasi ini dapat disebut juga sebagai **data hiding**. Salah satu keuntungan dari enkapsulasi adalah membuat suatu class hanya dapat diakses melalui class itu sendiri.

Syarat enkapsulasi meliputi :

1. Mendeklarasi variabel sebagai `private`.
2. Membuat method `public` `setter` dan `getter` untuk memodifikasi dan mengakses variabel tadi.

Enkapsulasi (lanjutan)

```
public class Rekening {
    private float money;

    public float retrieveMoney() {
        // for retrieving money
    }

    public void setMoney(float amount) {
        // for setting money value
    }
}
```

Apa yang terjadi jika money tidak diberi access modifier private / tidak menggunakan konsep enkapsulasi?

Enkapsulasi (contoh)

```java
package child1;

public class File1 {
    private String data;

    public File1(String data) {
        this.data = data;
    }

    public String getData() {
        return this.data;
    }

    public void setData(String data) {
        this.data = data;
    }
}
```

Ketika variabel `data` dijadikan `private`...

```java
package child2;

import child1.File1;

public class File2 {
    public static void main(String[] args) {
        File1 test = new File1("Tes package!");

        System.out.println(test.data);

        System.out.println(test.getData());
    }
}
```

Pemanggilan variabel `data` pada class lain akan menghasilkan error

Manfaat Enkapsulasi

1. **Data hiding**

   Program lebih *secure*.

   Tidak perlu tahu bagaimana cara variable dibuat atau disimpan. Begitupun dengan method.

2. **Flexibility**

   Bisa diatur apakah variable Read-only, atau Read-Write. Lebih maintainable.

3. **Reusability**

   Method bisa dipanggil dimanapun berkali-kali secara efektif (dalam arti mudah dibaca dan digunakan)

04

Overloading

Overloading (constructor)

Overloading sebuah constructor berarti membuat banyak konstruktor yang memiliki parameter yang berbeda-beda

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

    public Manusia() {
        nama = null;
        umur = 0;
    }

    public Manusia(String nama, int umur) {
        this.nama = nama;
        this.umur = umur;
    }

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

← Constructor default (tanpa parameter)

← Constructor dengan paramater `String`, `int`

← Constructor dengan paramater `int`, `String`

Overloading (method)

Overloading method sama halnya dengan overloading constructor karena constructor merupakan (special) method.

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

public void setUmur(String umur) {
this.umur = Integer.parseInt(umur);
}
```

← Method setter dengan parameter `int`

← Method setter dengan parameter `String`

Exercise!

Mari kita berlatih langsung agar cepat bisa

Exercise `Main.java`

Buat sebuah program untuk mencetak selisih terdekat dari tiga angka. Sebuah angka diinstansiasi pada sebuah objek.

Angka pertama dalam argumen method akan selalu menunjuk angka terkecil. Angka kedua dalam argumen method akan selalu menunjuk angka terbesar, namun bisa tidak dituliskan. Jika tidak dituliskan, angka terbesar akan selalu bernilai `100`.

Kembalikan selisih terkecil dari angka tengah dengan angka terkecil dan/ atau angka terbesar.

INPUT

- a : int (angka tengah)
- lo : int (angka terkecil)
- hi : int (optional - angka terbesar, default `100`)

OUTPUT

int (selisih terkecil antara angka tengah dengan angka terkecil atau angka terbesar)

CONTOH

Sampel Input & Output 1

```
26
15
```

```
11
```

Sampel Input & Output 2

```
62
31 88
```

```
26
```

Assignment!

Seperti biasa akan setiap selesai praktikum pasti ada tugas

Assignment 2 Soal 1 `soal1/Test.java`

Buat sebuah program **kalkulator sederhana** yang menggunakan 2 package. Package pertama bernama “io”, package kedua bernama “util”.

Dalam package “io” buat sebuah class yang memiliki method-method untuk **input dan output.**

Dalam package “util” buat sebuah class yang memiliki method-method untuk **operasi aritmatika** (tambah, kurang, kali, bagi) bilangan bulat.

INPUT

num1 : int (operan pertama)
num2 : int (operan kedua)
op : char (operator “+”,” -”, “*”, atau “/”)

OUTPUT

Hasil perhitungan antara num1 dan num2 menggunakan op

CONTOH

Sampel Input

```
8
4
/
```

Sampel Output

```
2
```

Assignment 2 Soal 2 `soal2/Test.java`

Menggunakan class Manusia dari praktikum sebelumnya, buatlah array objek Manusia berukuran **5** dengan properti-properti yang diberikan oleh user input. Kemudian output-kan nama dan umur setiap objek tersebut ke console.

INPUT

Untuk setiap manusia, ada input sebagai berikut secara konsekutif

- *nama* : string
- *umur* : int

OUTPUT

Untuk setiap manusia, ada output sebagai berikut secara konsekutif

- *"Orang ke i"*, dimana i adalah index
- *"Nama: nama"*, dimana nama adalah nama Manusia ke i
- *"Umur: umur"*, dimana umur adalah umur Manusia ke i

Assignment 2 Soal 3 `soal3/Test.java`

Buat class “Rekening” yang mempunyai default balance = 0 dan 2 (dua) public method (selain constructor)

- getBalance() : *float*
- addBalance() : *void*

Class lain tidak boleh mengakses member yang lain kecuali constructor dan 2 method diatas.

Program berjalan dan menerima input terus menerus (*looping*) sampai user keluar dari program.

INPUT

op : int (1 = get, 2 = add, 0 = exit)
addValue : int (hanya jika op = 2)

OUTPUT

Jika op = 1 ATAU op = 2 → Jumlah balance saat ini.
Jika op = 0 → Keluar dari program.

CONTOH

Sampel Input

```
2
2500
```

Sampel Output

```
2500
```

Assignment 2 Soal 4 `soal4/Test.java`

Buat 2 buah method untuk menambahkan 2 buah bilangan. Gunakan overloading untuk tipe data Integer dan Float.

INPUT

num1 : int
num2 : int
num3 : float
num4 : float

OUTPUT

result1 : int
result2 : float

CONTOH

Sampel Input

```
1
2
2.5
1.5
```

Sampel Output

```
3
4.0
```

Teknis Pengumpulan

Pengerjaan dan pengumpulan tugas akan dilakukan di **Github Classroom**

Kelas A:

Link Tugas Kelas A

Kelas B:

Link Tugas Kelas B

Accept assignment terlebih dahulu lalu link akun Github dengan slot nama yang sesuai di Github Classroom

Teknis Pengumpulan

Format setiap file `.java` didahulukan dengan Nama, NPM, Kelas, Tanggal, dan Deskripsi

Cara menambah comment di java

```
// untuk single line
/* untuk multiple line */
```

Contoh Format

```
/*
  Nama  : Jane Doe
  NPM       : 99
  Kelas     : A
  Tanggal   : 1 September 2021
  Deskripsi : Class jawaban exercise-01 soal-01
*/
```

Deadline Pengumpulan ⌚

Kelas A:

19 September 2022, 23:59 WIB

Kelas B:

20 September 2022, 23:59 WIB

Waktu yang dilihat adalah waktu last commit.
Jika ada yang commit melewati deadline walaupun sudah commit sebelumnya akan dianggap telat

Terima Kasih!

Do you have any questions? Please use respective class discussion channel on Discord.

Semangat terus menjalani kuilahnya!! 🔥🔥🔥