

# Fungsi/Method

Hafidz Mulia  
Teosofi Hidayah Agung

Departemen Matematika  
Institut Teknologi Sepuluh Nopember

28 Oktober 2024



## Masalah

*Copy-paste* memanglah hal yang sangat mudah dilakukan namun juga melelahkan jika yang akan di-*copas* terlalu banyak. Misalkan kita memiliki program yang memiliki 100 baris kode dan kita ingin mengulangnya sebanyak 10 kali. Tentu saja kita tidak akan menulis ulang program tersebut sebanyak 10 kali. Oleh karena itu, kita memerlukan suatu cara agar program tersebut dapat dijalankan berkali-kali tanpa harus menulis ulang program tersebut.

# Daftar isi

java.util.Date;

1 Fungsi

2 Method

- Struktur Method
- Parameter
- Pemanggilan Method

3 Rekursif

```
Date date = new Date();
oos.writeObject(date);
oos.flush();
oos.close();
fos.close();
```

4 Latihan

# Fungsi

## Definisi

**Fungsi** secara umum didefinisikan sebagai hubungan/pemetaan antara himpunan input dan himpunan output.

## Contoh

Misalkan  $A = \{\text{Fuad}, \text{Disty}, \text{Yusuf}\}$  dan  $B = \{\text{D}, \text{F}, \text{Y}\}$ . Kemudian Didefinisikan fungsi  $f : A \rightarrow B$  dengan aturan sebagai berikut:

►  $f :=$  Huruf pertama dari nama di  $A$

- Fuad  $\mapsto$  F
- Disty  $\mapsto$  D
- Yusuf  $\mapsto$  Y

►  $f :=$  Huruf terakhir dari nama di  $A$

- Fuad  $\mapsto$  D
- Disty  $\mapsto$  Y
- Yusuf  $\mapsto$  F

# Fungsi

## Perhatikan

Input yang diberikan kepada fungsi haruslah sesuai dengan definisi fungsi parameter tersebut. Jika tidak, maka akan terjadi error. Misalkan saja kita berikan input berupa angka pecahan kepada fungsi diatas maka jelas sekali si fungsi akan bingung.

# Daftar isi

---> java.util.Date;

1 Fungsi `public class SaveDate {`

2 Method

- Struktur Method
- Parameter
- Pemanggilan Method

3 Rekursif `Date date = new Date();`

`oos.writeObject(date);`

`oos.flush();`

`oos.close();`

`fos.close();`

# Method

## Definisi

**Method** adalah blok kode yang dirancang untuk melakukan tugas tertentu dan dapat dipanggil kapan saja dalam program. Penggunaan method membuat kode lebih terstruktur, dapat digunakan kembali (reusable), dan lebih mudah dibaca.

## Method Java

Berdasarkan asal atau sumbernya bisa dibagi menjadi dua jenis:

- ▶ **Built-in Method** adalah method yang sudah disediakan oleh Java. Contoh:  
`System.out.println(), Math.abs(), Math.sqrt(), dll.`
- ▶ **User-defined Method** adalah method yang dibuat oleh pengguna. Inilah yang akan kita pelajari.

# Method

## Struktur Method

### Kode: Struktur Method

```
1 <access_modifier> <return_type> <method_name>(list_of_parameters) {  
2     // Body of the method  
3     // Statements to be executed  
4     return <value>;  
5 }
```

- ▶ access\_modifier adalah kata kunci yang menentukan tingkat akses method.  
Contoh: public, private, protected, dll.
- ▶ return\_type adalah tipe data yang akan dikembalikan oleh method. Jika method tidak mengembalikan nilai, gunakan tipe data void.
- ▶ method\_name adalah nama method yang akan digunakan.
- ▶ parameters adalah variabel yang diperlukan oleh method.

# Method

## Struktur Method

### Catatan

Yang wajib dibuat dalam method adalah `return_type` dan `method_name`. Sedangkan `parameters` dan `access_modifier` adalah opsional.

### Contoh:

```
1 public static void hello() {  
2     System.out.println("Hello, World!");  
3 }
```

bisa diubah menjadi

```
1 static void hello() {  
2     System.out.println("Hello, World!");  
3 }
```

# Method

## Parameter

### Definisi

**Parameter** adalah variabel yang didefinisikan di dalam tanda kurung saat kita membuat sebuah method. Parameter digunakan untuk menerima data atau input dari luar method, sehingga method tersebut bisa bekerja berdasarkan nilai yang diberikan. Dengan kata lain, parameter memungkinkan kita untuk mengirimkan data yang berbeda setiap kali memanggil method yang sama, sehingga method tersebut bisa melakukan tugas yang lebih fleksibel.

# Method

## Parameter

Kode: Parameter pada Method

```
1  returnType methodName(parameterType parameterName) {  
2      // kode di dalam method  
3 }
```

## Penjelasan

- ▶ `parameterType` adalah tipe data dari parameter yang diterima method, seperti `int`, `double`, `String`, dll.
- ▶ `parameterName` adalah Nama variabel untuk parameter, yang akan digunakan sebagai referensi dalam method.

Paramater bisa lebih dari satu, dipisahkan dengan tanda koma.

# Method

## Parameter

Kode: Contoh Method berparameter

```
1 public static void greet(String name) {  
2     System.out.println("Hello, " + name);  
3 }
```

```
1 public static void f(double a, char b) {  
2     // Blok kode  
3     return int_type;  
4 }
```

# Method

## Pemanggilan Method

### Memanggil Method

Method yang telah dibuat tidak akan berjalan jika tidak dipanggil. Untuk memanggil method, kita cukup menuliskan nama method tersebut di dalam method `main()` dan jangan lupa untuk memberikan tanda kurung "()" di belakang nama method.

### Perbedaan

Membedakan antara method dan variabel cukup mudah yaitu method selalu diakhiri dengan tanda kurung "()". Sedangkan variabel tidak.

# Method

## Pemanggilan Method

Kode: Pemanggilan Method

```
1  public class Method {  
2      public static void main(String[] args) {  
3          hello();  
4          hello();  
5          hello();  
6      }  
7      static void hello() {  
8          System.out.println("Hello, World!");  
9      }  
10 }
```

```
Hello, World!  
Hello, World!  
Hello, World!
```

# Method

## Pemanggilan Method

Disinilah kita bisa menggunakan *syntax return* secara maksimal.

Kode: Penjumlahan Dua Bilangan

```
1 public class Method {  
2     public static void main(String[] args) {  
3         int x = 10, y = 20, z = add(x, y);  
4         System.out.println("Hasil " +x +" + " +y +" = " +z);  
5     }  
6     public static int add(int a, int b) {  
7         return a + b;  
8     }  
9 }
```

# Method

## Pemanggilan Method

### Pertanyaan

- Apa yang terjadi jika kita menghilangkan `return` pada method `add()`?
- Bisakah method `add()` langsung dipanggil di dalam `println`?
- Jika kita hanya menuliskan `add(x, y)` tanpa menyimpannya ke dalam variabel, apakah akan terjadi error?

# Daftar isi

java.util.Date;

1 Fungsi  
public class SaveDate {

2 Method

- Struktur Method
- Parameter
- Pemanggilan Method

3 Rekursif  
Date date = new Date();

oos.writeObject(date);

oos.flush();

oos.close();

fos.close();

# Rekursif

## Definisi

**Rekrusif** adalah teknik pemrograman dimana suatu fungsi memanggil dirinya sendiri. Rekursi adalah cara menyelesaikan masalah dengan memecah masalah besar menjadi sub-masalah yang lebih kecil dan lebih sederhana hingga mencapai kondisi dasar (atau base case) yang dapat diselesaikan secara langsung tanpa rekursi lebih lanjut.

Kode: Salah satu struktur Rekrusif

```
1 return_type method_name(parameters) {  
2     // Statements  
3     return method_name(another_parameters);  
4 }
```

## Infinite Recursive

Rekursif pada dasarnya adalah pengulangan yang dilakukan oleh method itu sendiri. Oleh karena itu, dalam menggunakannya perlu diperhatikan agar tidak terjadi *infinite recursive*.

### Kode: Infinite Recursive

```
1  public static void infinite() {  
2      System.out.println("Infinite Recursive");  
3      infinite();  
4  }
```

## Contoh

Program untuk menghitung faktorial dari suatu bilangan.

### Kode: Faktorial

```
1 public class Method {  
2     public static void main(String[] args) {  
3         int n = 5;  
4         System.out.println("Faktorial dari " +n +" adalah " +  
5             factorial(n));  
6     }  
7     public static int factorial(int n) {  
8         if (n == 0 || n == 1) return 1;  
9         return n * factorial(n - 1);  
10    }  
11 }
```

# Daftar isi

java.util.Date;

1 Fungsi  
public class SaveDate {

2 Method

- Struktur Method
- Parameter
- Pemanggilan Method

3 Rekursif  
Date date = new Date();

oos.writeObject(date);

oos.flush();

oos.close();

fos.close();

## Latihan 1

Buatlah program untuk menentukan mana diantara 2 bilangan yang terbesar ( $\max(a, b)$ ) menggunakan method.

## Latihan 2

Buatlah program yang menerima input  $n$  bilangan asli dan memunculkan output berupa segitiga pascal.

## Latihan 3

Buatlah program untuk menghitung jumlahan faktor dari bilangan bulat positif  $n$ .  
Contoh:  $n = 6$ , maka faktornya adalah 1, 2, 3, 6. Jumlahnya adalah 12.