Algoritma dan Pemrograman Komputer 1

Bab 9: Fungsi/Method

Aslam Pandu Tasminto – 5002241025
M. Ma'ruf Qomaruddin Kafi – 5002241095

November 2024

Departemen Matematika
Fakultas Sains dan Analitika Data
Institut Teknologi Sepuluh Nopember

Daftar Isi

Pengenalan Fungsi/Method

Konsep Dasar Fungsi/Method

Apa itu Fungsi/Method?

Fungsi/Method adalah kumpulan beberapa pernyataan yang dikelompokkan untuk melakukan suatu operasi tertentu.

- Seperti **mesin kecil** dalam program yang melakukan tugas spesifik
- Dapat dipanggil berulang kali dari berbagai bagian program
- Meningkatkan **reusability**, **readability**, **modularity**, dan **maintainability** kode

Konsep Fundamental Method

Dalam memahami method, terdapat 3 konsep fundamental yang harus dikuasai:

1. **Parameter** - Input data yang diproses method
2. **Statement** - Logika dan operasi yang dilakukan
3. **Return Type/Value** - Output hasil pemrosesan

Penting: Meskipun secara teknis parameter bisa kosong dan return type bisa void,

Struktur Fungsi/Method

Anatomi Fungsi/Method

6 Komponen Utama Fungsi/Method

1. **Modifier** - Akses kontrol (`public static`)
2. **Return Type** - Tipe data kembalian
3. **Nama Method** - Identifier fungsi
4. **Parameter** - Data input method
5. **Exception** - Penanganan error
6. **Statement** - Badan method

Contoh Struktur

```
1 public static boolean isBigger(int a, int b) {
2     if (a > b) {
3         return true;    // Return Value
4     } else {
5         return false;   // Return Value
6     }
7 }
```

Aturan Penamaan

Konvensi Penamaan Method

Pattern	Contoh dan Penjelasan
`camelCase`	`calculateArea()`, `getUserName()`
Awalan huruf kecil	`make()`, `processData()`, `isEmpty()`
Kata kerja deskriptif	`convertToMeter()`, `validateInput()`
Konsisten	`getX()`, `setX()`, `isX()`, `hasX()`

Tabel 1: Konvensi Penamaan Method yang Direkomendasikan

Best Practice

- Gunakan nama yang menjelaskan **apa yang dilakukan** method
- Hindari nama singkat yang tidak jelas (`proc()`, `calc()`)
- Untuk boolean, gunakan awalan `is`, `has`, `can`

Jenis-jenis Method

Method Tanpa Parameter

Method Tanpa Input Parameter

```
1  class MethodSederhana {
2      public static void sapa() {
3          System.out.println("Halo! Selamat belajar Java!");
4      }
5
6      public static void main(String[] args) {
7          sapa();  // Memanggil method
8          sapa();  // Bisa dipanggil berkali-kali
9      }
10 }
```

Output Program

```
Halo!  Selamat belajar Java!

Halo!  Selamat belajar Java!
```

Method Dengan Parameter

Method dengan Input Parameter

```java
class MethodParameter {
    public static void sapaNama(String nama) {
        System.out.println("Halo, " + nama + "!");
    }

    public static void hitungLuas(int panjang, int lebar) {
        int luas = panjang * lebar;
        System.out.println("Luas: " + luas);
    }

    public static void main(String[] args) {
        sapaNama("Budi");
        hitungLuas(5, 3);
    }
}
```

Output Program

```
Halo, Budi!
Luas:  15
```

Method Dengan Return Value

Method yang Mengembalikan Nilai

```java
public class MethodReturn {
    public static int tambah(int a, int b) {
        return a + b;
    }

    public static boolean adalahGenap(int angka) {
        return angka % 2 == 0;
    }

    public static void main(String[] args) {
        int hasil = tambah(10, 5);
        boolean cek = adalahGenap(hasil);

        System.out.println("10 + 5 = " + hasil);
        System.out.println(hasil + " genap? " + cek);
    }
}
```

Output Program

```
10 + 5 = 15
15 genap?  false
```

Method Rekursif

Contoh Method Rekursif (Faktorial)

```
1  class MethodRekursif {
2      public static int faktorial(int n) {
3          if (n == 0 || n == 1) {
4              return 1;
5          } else {
6              return n * faktorial(n - 1);
7          }
8      }
9
10     public static void main(String[] args) {
11         int angka = 5;
12         int hasil = faktorial(angka);
13         System.out.println(angka + "! = " + hasil);
14     }
15 }
```

Output Program

```
5!  = 120
```

Parameter vs Argument

Perbedaan Parameter dan Argument

Parameter	Argument
Variabel dalam deklarasi method	Nilai aktual yang diberikan saat pemanggilan
`String nama` dalam deklarasi	`"Budi"` saat memanggil method
Seperti variabel yang didapat	Seperti nilai yang diisi
Ditentukan saat menulis method	Diberikan saat menggunakan method

Tabel 2: Perbedaan Konseptual Parameter dan Argument

Contoh

```java
// Parameter: 'nama' dan 'umur'
public static void perkenalan(String nama, int umur) {
    System.out.println("Nama: " + nama + ", Umur: " + umur);
}

// Argument: "Alice" dan 20
perkenalan("Alice", 20);
```

Method Overloading

Konsep Method Overloading

Apa itu Overloading?
Membuat beberapa method dengan **nama sama** tetapi **parameter berbeda** (jumlah atau tipe data).

Contoh Overloading

```
1  public class Calculator {
2      // Versi 1: dua parameter integer
3      public static int tambah(int a, int b) {
4          return a + b;
5      }
6
7      // Versi 2: tiga parameter integer
8      public static int tambah(int a, int b, int c) {
9          return a + b + c;
10     }
11
12     // Versi 3: parameter double
13     public static double tambah(double a, double b) {
14         return a + b;
15     }
16 }
```

Penggunaan Method Overloading

Implementasi dalam Program

```
1  public class Calculator {
2      public static void main(String[] args) {
3          // Memanggil versi berbeda berdasarkan parameter
4          int hasil1 = tambah(5, 3);
5          int hasil2 = tambah(1, 2, 3);
6          double hasil3 = tambah(2.5, 3.7);
7
8          System.out.println("5 + 3 = " + hasil1);
9          System.out.println("1 + 2 + 3 = " + hasil2);
10         System.out.println("2.5 + 3.7 = " + hasil3);
11     }
12 }
```

Output Program

```
5 + 3 = 8

1 + 2 + 3 = 6

2.5 + 3.7 = 6.2
```

Studi Kasus

Studi Kasus 1: Kalkulator

Implementasi Kalkulator dengan Method

```java
public class Kalkulator {
    public static double tambah(double a, double b) {
        return a + b;
    }

    public static double kurang(double a, double b) {
        return a - b;
    }

    public static double kali(double a, double b) {
        return a * b;
    }

    public static double bagi(double a, double b) {
        if (b == 0) {
            System.out.println("Error: Pembagian dengan nol!");
            return 0;
        }
        return a / b;
    }

    public static void main(String[] args) {
        System.out.println("10 + 5 = " + tambah(10, 5));
        System.out.println("10 - 5 = " + kurang(10, 5));
        System.out.println("10 * 5 = " + kali(10, 5));
        System.out.println("10 / 5 = " + bagi(10, 5));
    }
}
```

Studi Kasus 2: Validasi Data

Method untuk Validasi Input

```
1  public class Validator {
2      public static boolean isValidEmail(String email) {
3          return email.contains("@") && email.contains(".");
4      }
5
6      public static boolean isValidAge(int age) {
7          return age >= 0 && age <= 150;
8      }
9
10     public static boolean isStrongPassword(String password) {
11         return password.length() >= 8 &&
12                !password.equals(password.toLowerCase());
13     }
14
15     public static void main(String[] args) {
16         System.out.println("Email valid? " + isValidEmail("user@example.com"));
17         System.out.println("Umur valid? " + isValidAge(25));
18         System.out.println("Password kuat? " + isStrongPassword("Pass1234"));
19     }
20 }
```

Keuntungan Method

Manfaat Penggunaan Method

Keuntungan	Deskripsi
Reusability	Dapat dipakai berulang tanpa menulis ulang kode
Modularity	Memecah program kompleks menjadi bagian kecil
Maintainability	Mudah di-maintain dan debug
Readability	Kode lebih terstruktur dan mudah dibaca
Testing	Mudah di-test secara terpisah
Collaboration	Memungkinkan kerja tim yang lebih baik

Tabel 3: Keuntungan Menggunakan Method dalam Pemrograman

Best Practices

Best Practices Penggunaan Method

Tips dan Rekomendasi

- **Satu tugas satu method** - Setiap method harus melakukan satu tugas spesifik
- **Nama deskriptif** - Nama method harus jelas menjelaskan fungsinya
- **Parameter minimal** - Gunakan parameter secukupnya, hindari terlalu banyak
- **Return early** - Keluar dari method secepat mungkin jika kondisi terpenuhi
- **Fokus pada tugas** - Method sebaiknya hanya mengolah input menjadi output
- **Dokumentasi** - Berikan komentar untuk method yang kompleks

Latihan

Latihan 1: Method Dasar

Soal 1: Method Sederhana
Buat method `volumeTabung(double r, double t)` yang mengembalikan volume tabung. **Rumus:** $V = \pi \times r^2 \times t$

Contoh:

- Input: r=7, t=10 $\rightarrow$ Output: 1539.38 (sekitar)
- Gunakan `Math.PI` untuk nilai

Latihan 2: Method Boolean

Soal 2: Validasi Bilangan
Buat method `isBilanganPrima(int n)` yang mengembalikan `true` jika bilangan prima, `false` jika bukan.

Ciri bilangan prima:

- Hanya bisa dibagi 1 dan dirinya sendiri
- Contoh: 2, 3, 5, 7, 11, 13, ...

Contoh:

- Input: 7 $\rightarrow$ Output: true
- Input: 9 $\rightarrow$ Output: false

Latihan 3: Method Rekursif

Soal 3: Deret Fibonacci
Buat method rekursif `fibonacci(int n)` yang mengembalikan suku ke-n dari deret Fibonacci.

Rumus Fibonacci:

- F(1) = 1, F(2) = 1
- F(n) = F(n-1) + F(n-2) untuk n >2

Contoh:

- Input: 5 → Output: 5
- Input: 7 → Output: 13

Kesimpulan

Kesimpulan

Inti Bab 9: Fungsi/Method

- **Fungsi/Method** adalah blok kode yang melakukan tugas spesifik dan dapat dipanggil berulang
- **Struktur method** terdiri dari 6 komponen: modifier, return type, nama method, parameter, exception, statement
- **Jenis method**: tanpa parameter, dengan parameter, dengan return value, dan rekursif
- **Method overloading** memungkinkan beberapa method dengan nama sama tetapi parameter berbeda
- Penggunaan method meningkatkan **Reusability, Readability, Modularity, dan Maintainability** kode

Referensi

Referensi

Referensi:

- **Modul Praktikum Algoritma dan Pemrograman - Modul 9**
  Departemen Matematika FSAD ITS
- **Oracle Java Tutorials - Defining Methods**
  `https://docs.oracle.com/javase/tutorial/java/javaOO/methods.html`
- **GeeksforGeeks - Methods in Java**
  `https://www.geeksforgeeks.org/methods-in-java/`

Referensi

Referensi:

- **W3Schools - Java Methods**
  `https://www.w3schools.com/java/java_methods.asp`
- **Programiz - Java Methods**
  `https://www.programiz.com/java-programming/methods`

Terima Kasih