



POLITEKNIK
NEGERI JEMBER

PBD

Java Methods (Parameters, Overloading, scope, dan Recursion)

By: Andri Permana Wicaksono

Pengertian Methods

- **Method** adalah blok kode yang dibuat untuk melakukan tugas tertentu dan dapat dipanggil berkali-kali.
- **Tujuan:** mempermudah pengelolaan kode, menghindari duplikasi, dan meningkatkan keterbacaan program.

```
returnType methodName(parameter1, parameter2, ...) {  
    // body method  
    return value; // jika returnType bukan void  
}
```

Parameter pada Method

- **Parameter** adalah variabel yang didefinisikan dalam tanda kurung saat membuat method, digunakan untuk menerima nilai dari pemanggil method.

```
public class ParameterExample {  
    // method dengan parameter  
    static void greet(String name) {  
        System.out.println("Halo, " + name);  
    }  
  
    public static void main(String[] args) {  
        greet("Andi");  
        greet("Budi");  
    }  
}
```

OUTPUT :

Halo, Andi
Halo, Budi

Method Overloading

- **Overloading:** membuat beberapa method dengan nama yang sama tetapi memiliki jumlah atau tipe parameter berbeda.
- Java membedakan method berdasarkan signature (nama + parameter).

```
public class OverloadingExample {  
    // overloading method tambah  
    static int tambah(int a, int b) {  
        return a + b;  
    }  
  
    static double tambah(double a, double b) {  
        return a + b;  
    }  
  
    public static void main(String[] args) {  
        System.out.println(tambah(5, 3));    // memanggil versi int  
        System.out.println(tambah(2.5, 3.5)); // memanggil versi double  
    }  
}
```

OUTPUT:

8
6.0

Scope pada Method

- **Scope:** ruang lingkup dimana sebuah variabel dapat diakses.
- Variabel yang dideklarasikan di dalam method hanya bisa diakses di dalam method tersebut (local variable).
- Variabel yang dideklarasikan di dalam class tetapi di luar method disebut instance variable / global variable, dapat diakses oleh semua method dalam class.

Scope pada Method

- `public class ScopeExample {`
- `int globalVar = 10; // variabel global`
- `void testScope() {`
- `int localVar = 5; // variabel lokal`
- `System.out.println("Global: " + globalVar);`
- `System.out.println("Lokal: " + localVar);`
- `}`
- `public static void main(String[] args) {`
- `ScopeExample obj = new ScopeExample();`
- `obj.testScope();`
- `// System.out.println(localVar); // ERROR: localVar tidak dikenal di sini`
- `}`
- `}`

Recursion pada Method

- **Recursion:** method yang memanggil dirinya sendiri.
- Harus ada base case (kondisi berhenti) agar tidak terjadi infinite recursion.

```
public class RecursionExample {  
    // method rekursif  
    static int factorial(int n) {  
        if (n == 0 || n == 1) {  
            return 1; // base case  
        } else {  
            return n * factorial(n - 1); // recursive call  
        }  
    }  
  
    public static void main(String[] args) {  
        System.out.println("Faktorial 5 = " + factorial(5));  
    }  
}
```

OUTPUT:

Faktorial 5 = 120

Perbandingan Rekursi vs Iterasi

- **Iterasi:** menggunakan perulangan (for/while).
- **Rekursi:** lebih ringkas dan elegan untuk kasus matematis (faktorial, Fibonacci, menelusuri struktur data).
- Namun, rekursi lebih boros memori karena tiap pemanggilan method disimpan di call stack.

Ringkasan

- **Parameter** → memungkinkan method menerima input.
- **Overloading** → method dengan nama sama tetapi parameter berbeda.
- **Scope** → menentukan di mana variabel bisa diakses.
- **Recursion** → method yang memanggil dirinya sendiri.

Tugas

- 1) Buat method perkalian(int a, int b) yang mengembalikan hasil perkalian.
- 2) Overloading method perkalian(double a, double b).
- 3) Gunakan variabel global dan lokal untuk menampilkan perbedaan scope.
- 4) Buat method rekursif untuk menghitung deret Fibonacci.



**POLITEKNIK
NEGERI JEMBER**

Thank You

Insert the Sub Title of Your Presentation

Politeknik Negeri Jember

Jalan Mastrip Kotak Pos 164 Jember 68101

Telp. (0331) 333532-34

Fax. (0331) 333531

