

Nama : Khaliffa Mutiara Dalenti

Kelas : TI21E

NIM : 20210040099

## Tugas Praktikum Inheritance

### Percobaan 1:

Percobaan berikut ini menunjukkan penggunaan kata kunci "super".

```
class Parent {
    public int x = 5;
}

class Child extends Parent {
    public int x = 10;
    public void Info(int x) {
        System.out.println("Nilai x sebagai parameter = " + x);
        System.out.println("Data member x di class Child = " + this.x);
        System.out.println("Data member x di class Parent = " +
            super.x);
    }
}

public class NilaiX {
    public static void main(String args[]) {
        Child tes = new Child();
        tes.Info(20);
    }
}
```

Analisa: ketika objek "tes" dibuat dan objek itu memanggil fungsi info maka output yang akan dihasilkan yaitu 20, 10, 5. Walaupun variabel yang sama yaitu "x" tetapi yang membedakan adalah pemanggil variabelnya.

Jika hanya "x" saja yang dipanggil maka x itu hanya nilai dari parameter (jika itu dalam sebuah fungsi) karena itu bernilai 20. Jika "this.x" maka nilai yang diambil adalah nilai x yang menempel pada objek itu karena itu bernilai 10. Sedang "super.x" dia akan mengambil nilai pada parent class karena itu bernilai 5.

### Percobaan 2:

Percobaan berikut ini menunjukkan penggunaan kontrol akses terhadap atribut parent class. Mengapa terjadi error, dan bagaimana solusinya?

```
public class Pegawai {
    private String nama;
    public double gaji;
}

public class Manajer extends Pegawai {
    public String departemen;

    public void IsiData(String n, String d) {
        nama=n;
        departemen=d;
    }
}
```

Analisa: terjadi error karena pada kelas Manajer dan dalam fungsi Isi Data memanggil variabel nama sedangkan dalam class Manajer tidak ada variabel nama.

Solusi: atribut nama pada kelas pegawai access modifier diganti dari private menjadi public.

Dan pemanggilan nama pada fungsi IsiData diganti menjadi super.nama = n

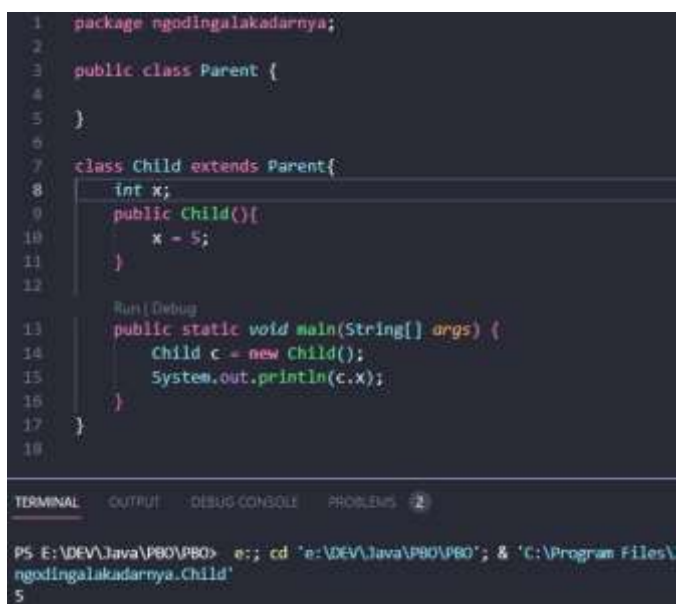
### Percobaan 3:

Percobaan berikut ini menunjukkan penggunaan konstruktor yang tidak diwariskan.

Mengapa terjadi error, dan bagaimana solusinya?

```
public class Parent {  
    // kosong  
}  
  
public class Child extends Parent {  
    int x;  
    public Child() {  
        x = 5;  
    }  
}
```

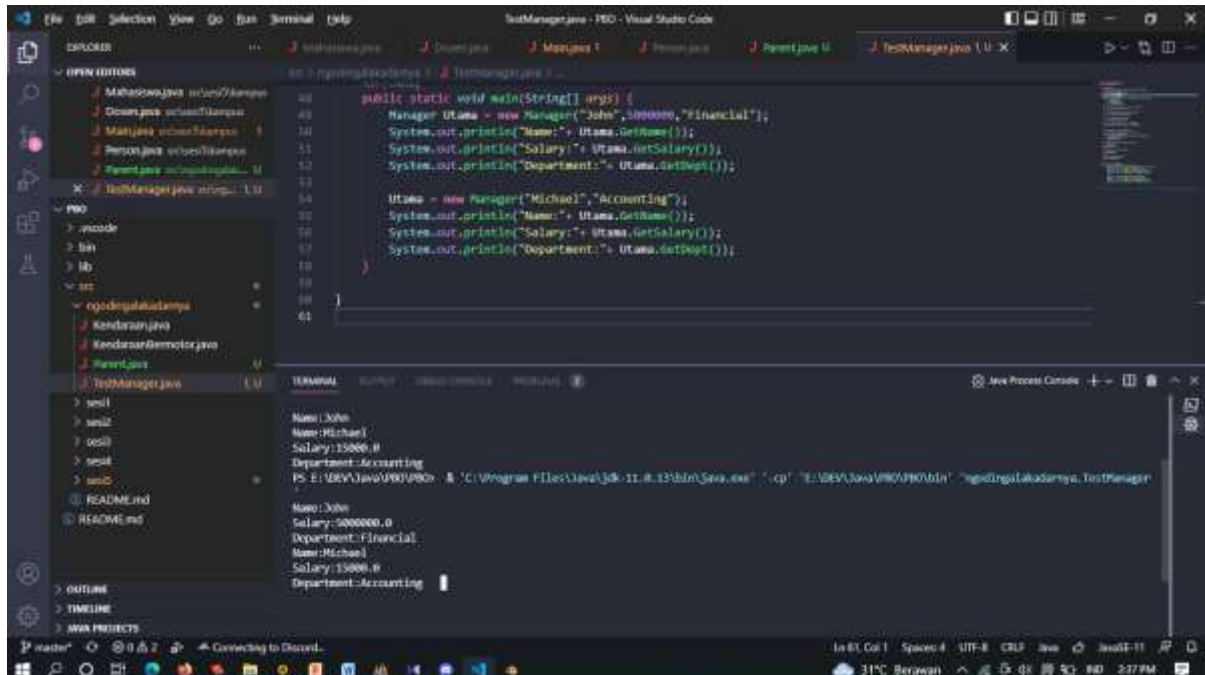
Analisa: tidak terjadi error walaupun parent class tidak mempunyai konstruktor



```
1 package ngodingalakadarnya;  
2  
3 public class Parent {  
4  
5 }  
6  
7 class Child extends Parent{  
8     int x;  
9     public Child(){  
10         x = 5;  
11     }  
12  
13     Run | Debug  
14     public static void main(String[] args) {  
15         Child c = new Child();  
16         System.out.println(c.x);  
17     }  
18 }  
  
TERMINAL  OUTPUT  DEBUG CONSOLE  PROBLEMS  2  
PS E:\DEV\Java\PBO\PBO> e;; cd "e:\DEV\Java\PBO\PBO"; & 'C:\Program Files\  
ngodingalakadarnya.Child'  
5
```

#### Percobaan 4:

Setelah dicoba dalam vscode tidak ada error maupun masalah



The screenshot shows the Visual Studio Code interface with a Java project. The main editor displays the `TestManager.java` file, which contains a `main` method. The code creates two `Manager` objects: `Utama` with parameters `"John", 5000000, "Financial"` and `Utama2` with parameters `"Michael", "Accounting"`. The `main` method calls `getManager()` on both objects and prints their details. The left sidebar shows the project structure with files like `Manager.java`, `Parent.java`, and `TestManager.java`. The bottom panel shows the output of the program, which matches the expected results from the code.

```
public static void main(String[] args) {  
    Manager Utama = new Manager("John", 5000000, "Financial");  
    System.out.println("Name: " + Utama.getName());  
    System.out.println("Salary: " + Utama.getSalary());  
    System.out.println("Department: " + Utama.getDept());  
  
    Manager Utama2 = new Manager("Michael", "Accounting");  
    System.out.println("Name: " + Utama2.getName());  
    System.out.println("Salary: " + Utama2.getSalary());  
    System.out.println("Department: " + Utama2.getDept());  
}
```

Output:

```
Name: John  
Salary: 5000000.0  
Department: Financial  
Name: Michael  
Salary: 13000.0  
Department: Accounting
```

karena semua penggunaan sudah benar,

Pemanggilan objek pertama menggunakan contruktur dengan 3 parameter yaitu nama, salary dan Dept sedangkan objek kedua menggunakan konstruktor dengan 2 parameter yaitu nama dan Dept.

#### Percobaan 5 :

Analisis: Tidak ada masalah dalam program ini, program ini akan menjalankan kelas yang dibuat menjalankan fungsi fungsinya.

#### Percobaan 6 :

Analisis: terdapat 2 kelas yaitu kelas A sebagai parent class dan kelas B sebagai subclass dari A, sub class A akan mengganti nilai `var_a` dan `var_b` dari parent kelas nya. Ketika objek B dibuat, constuktor A akan tetap dijalankan.

### Percobaan 7 :

```
15 class Anak extends Bapak{
16     int c;
17     int a;
18     int b;
19     void show_variabel(){
20         System.out.println("nilai a = " + super.a);
21         System.out.println("nilai b = " + super.b);
22         System.out.println("nilai c = " + c);
23     }
24 }
25
26
27 public class InheritExample {
28
29     Run | Debug
30
31 TERMINAL OUTPUT DEBUG CONSOLE PROBLEMS 2 Java Process Console
32
33 Try the new cross-platform PowerShell https://aka.ms/pscore6
34
35 PS E:\DEV\Java\PBO\PBO> & 'C:\Program Files\Java\jdk-11.0.13\bin\java.'
36 'E:\DEV\Java\PBO\PBO\bin' 'ngodingalakadarnya.InheritExample'
37
38 nilai a = 1
39 nilai b = 2
40 nilai a = 0
41 nilai b = 0
42 nilai c = 90
```

Analisis: Walaupun sudah menggunakan super pada kelas anak untuk mengakses nilai dari parent kelas, nilai a dan b dari kelas anak akan tetap 0 karena pada dasarnya blueprint nya bernilai 0. Jadi objek Anak tidak akan melakukan “override” pada objek Bapak, selama dalam bentuk Objek.

### Percobaan 8 :

```
public class Parent {
    String parentName;
    Parent() {}

    Parent(String parentName) {
        this.parentName = parentName;
        System.out.println("Konstruktor parent");
    }
}

class Baby extends Parent {
    String babyName;

    Baby(String babyName) {
        super();
        this.babyName = babyName;
        System.out.println("Konstruktor Baby");
        System.out.println(babyName);
    }

    public void Cry() {
        System.out.println("Owek owek");
    }
}
```

Analisis: pada kelas Baby menurunkan Parent. terdapat super() pada fungsi konstruktor yang akan mengoveride kelas parentnya. this.babyName = babyName untuk passing nilai babyName pada objek dengan parameter konstruktor babyName