TUGAS PRAKTIKUM

Nama : Dave Maulana Ferros

Kelas : TI21E

NIM : 20210040114

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);
    }
}
```

Hasil Analisa : Ketika object 'tes' dibuat, dan object tersebut memanggil fungsi info, maka output yang dihasilkan yaitu '20, 15, 5'. Meskipun variabel yang dipanggil sama yaitu 'x', yang membedakan adalah pemanggilan dari variabelnya.

Jika hanya variabel 'x' yang dipanggil, maka nilai yang muncul adalah 20 (karena x merupakan nilai dari parameter). Jika yang dipanggil 'this.x' maka nilai 10 lah yang akan diambil. Sedangkan 'super.x' akan memanggil nilai dari parent class yaitu 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;
   }
}
```

Hasil Analisa : Terjadi error karena pada class IsiData di dalam class Manajer memanggil variabel nama, sedangkan didalam class Manajer tidak terdapat variabel nama (Ada di parent class tetapi dalam kondisi private)

Solusi : Mengubah atribut nama pada class Pegawai menjadi public agar bisa diakses oleh class Manajer, serta pemanggilan nama pada fungsi IsiData diubah 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;
    }
}
```

Hasil Analisa : Tidak terjadi error walaupun parent class tidak mempunyai Constructor

Percobaan 4:

Percobaan berikut ini menunjukkan penggunaan kelas Employee dan subkelas Manager yang merupakan turunannya. Kelas TestManager digunakan untuk menguji kelas Manager.

```
class Employee {
    private static final double BASE_SALARY = 15000.00;
    private String Name = "";
    private double Salary = 0.0;
    private Date birthDate;

    public Employee() {}
    public Employee(String name, double salary, Date DoB){
        this.Name=name;
        this.Salary=salary;
        this.birthDate=DoB;
    }
    public Employee(String name,double salary){
        this(name,salary,null);
    }
    public Employee(String name, Date DoB){
        this(name,BASE_SALARY,DoB);
    }
    public Employee(String name){
        this(name,BASE_SALARY);
    }

    public String GetName(){ return Name;}
    public double GetSalary(){ return Salary; }
}

class Manager extends Employee {

    //tambahan attribrute untuk kelas manager
    private String department;

    public Manager(String name,double salary,String dept){
        super(name,salary);
        department=dept;
    }
    public Manager(String n,String dept){
        super(n);
        department=dept;
    }
    public Manager(String dept){
        super();
        department=dept;
    }
    public String GetDept(){
        return department;
    }
}

public class TestManager {

    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());

        Utama = new Manager("Michael","Accounting");
        System.out.println("Name:"+ Utama.GetName());
        System.out.println("Salary:"+ Utama.GetSalary());
        System.out.println("Department:"+ Utama.GetDept());
    }
}
```

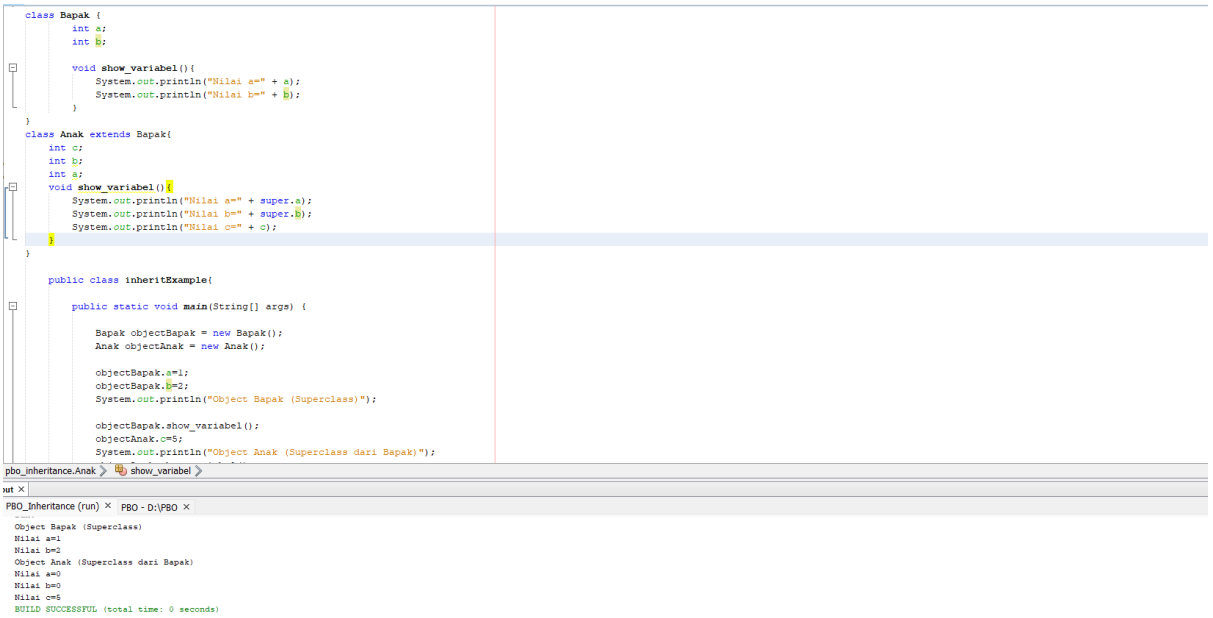
Hasil Analisa : Tidak terdapat error karena semua penggunaan fungsi sudah benar

Pemanggilan objek pertama menggunakan constructor dengan 3 parameter yaitu nama,salary, dan Dept sedangkan object kedua menggunakan constructor dengan 2 parameter yaitu nama dan Dept

Percobaan 5 :
Hasil Analisa : Tidak terjadi masalah saat menjalankan kode, program ini menjalankan kelas yang dibuat untuk menjalankan fungsinya.

Percobaan 6 :
Hasil Analisa : Terdapat 2 kelas yaitu kelas A sebagai parent, dan kelas B sebagai subclass dari parent A, subclass dari A akan mengganti nilai ari var_a dan var_b dari paket class nya. Saat objek B dibuat, constructor A akan tetap dijalankan.

Percobaan 7 :



```
class Bapak {
        int a;
        int b;

        void show_variabel(){
            System.out.println("Nilai a=" + a);
            System.out.println("Nilai b=" + b);
        }
}
class Anak extends Bapak{
    int c;
    int b;
    int a;
    void show_variabel(){
        System.out.println("Nilai a=" + super.a);
        System.out.println("Nilai b=" + super.b);
        System.out.println("Nilai c=" + c);
    }
}

    public class inheritExample{

        public static void main(String[] args) {

            Bapak objectBapak = new Bapak();
            Anak objectAnak = new Anak();

            objectBapak.a=1;
            objectBapak.b=2;
            System.out.println("Object Bapak (Superclass)");

            objectBapak.show_variabel();
            objectAnak.c=5;
            System.out.println("Object Anak (Superclass dari Bapak)");
```

```
pbo_inheritance.Anak > show_variabel >
out ×
PBO_Inheritance (run) × PBO - D:\PBO ×
Object Bapak (Superclass)
Nilai a=1
Nilai b=2
Object Anak (Superclass dari Bapak)
Nilai a=0
Nilai b=0
Nilai c=5
BUILD SUCCESSFUL (total time: 0 seconds)
```

Hasil Analisa : Walaupun sudah menggunakan super pada subclass untuk mengakses nilai pada parent class, nilai a dan b dari subclass akan tetap 0, karena nilainya dasarnya 0. Jadi objek subclass 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");
    }
```

Hasil Analisa : Pada kelas Parent menurunkan Baby. Terdapat super() pada fungsi constructor yang akan meng-override class parent-nya. this.babyName = babyName untuk melempar nilai babyName pada objek dengan parameter constructor babyName