Nama : Fahad Rizki

Kelas : TI21F

NIM : 20210040158

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);
    }
}
```

Jawab : 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 pemanggila 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 yangmenempel pada objek itu karena itu bernilai 10. Sedang "super.x" dia aka 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;
   }
}
```

Jawab : 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:

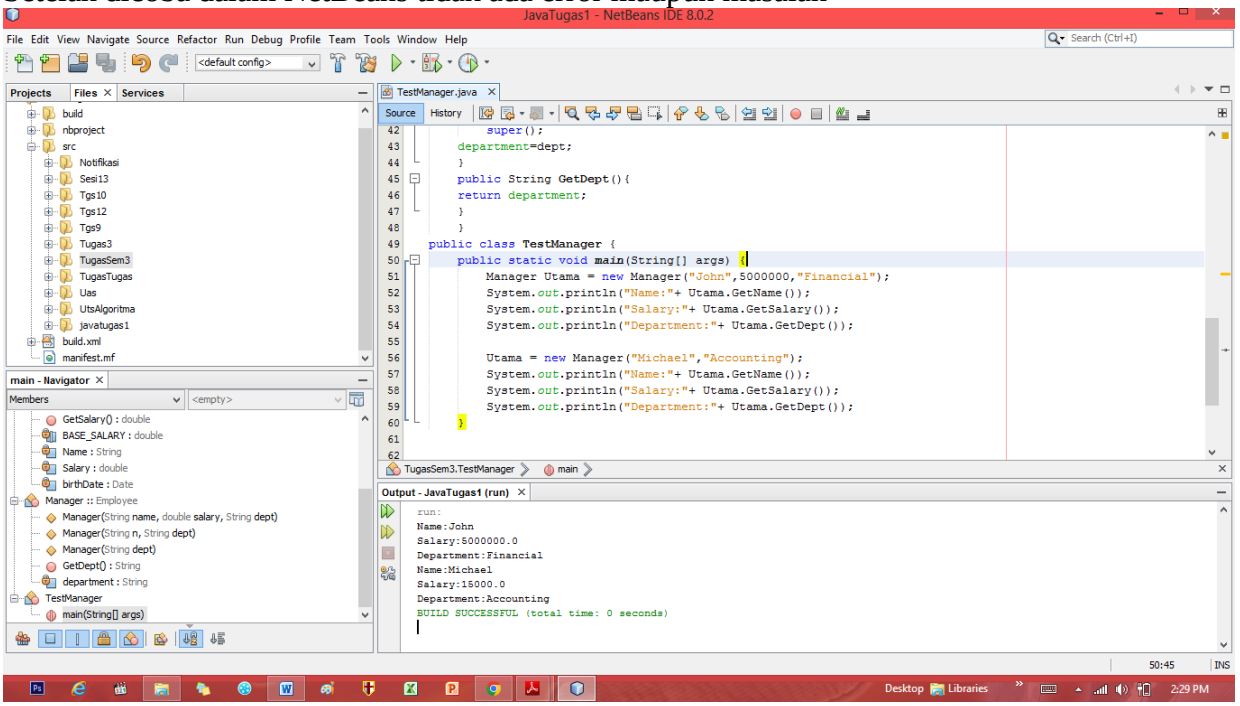
Percobaan berikut ini menunjukkan penggunaan konstruktor yang tidak diwariskan. Mengapa terjadi error, dan bagaimana solusinya?

```
public class Parent {
     // kosong
}

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

Jawab : tidak terjadi error walaupun parent class tidak mempunyai konstruktor

Percobaan 4:

Setelah dicoba dalam NetBeans tidak ada error maupun masalah



karena semua penggunaan sudah benar, Pemanggilan objek pertama menggunakan construktor dengan 3 parameter yaitu nama,salary dan Dept sedangkan objek kedua menggunakan construktor dengan 2 parameter yaitu nama dan Dept.

Percobaan 5:

```
public class MoodyObject {

    protected String getMood(){
        return "moody";
    }
    public void speak(){
        System.out.println("I am"+getMood());
    }
    void laugh() {}
    void cry() {}
}

public class SadObject extends MoodyObject{
    protected String getMood(){
        return "sad";
    }
    public void cry(){
        System.out.println("Hoo hoo");
    }
}

public class HappyObject extends MoodyObject{
    protected String getMood(){
        return"happy";
    }
    public void laugh(){
    System.out.println("Hahaha");
    }
}

public class MoodyTest {
    public static void main(String[] args) {

        MoodyObject m = new MoodyObject();

        //test perent class
        m.speak();
```

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

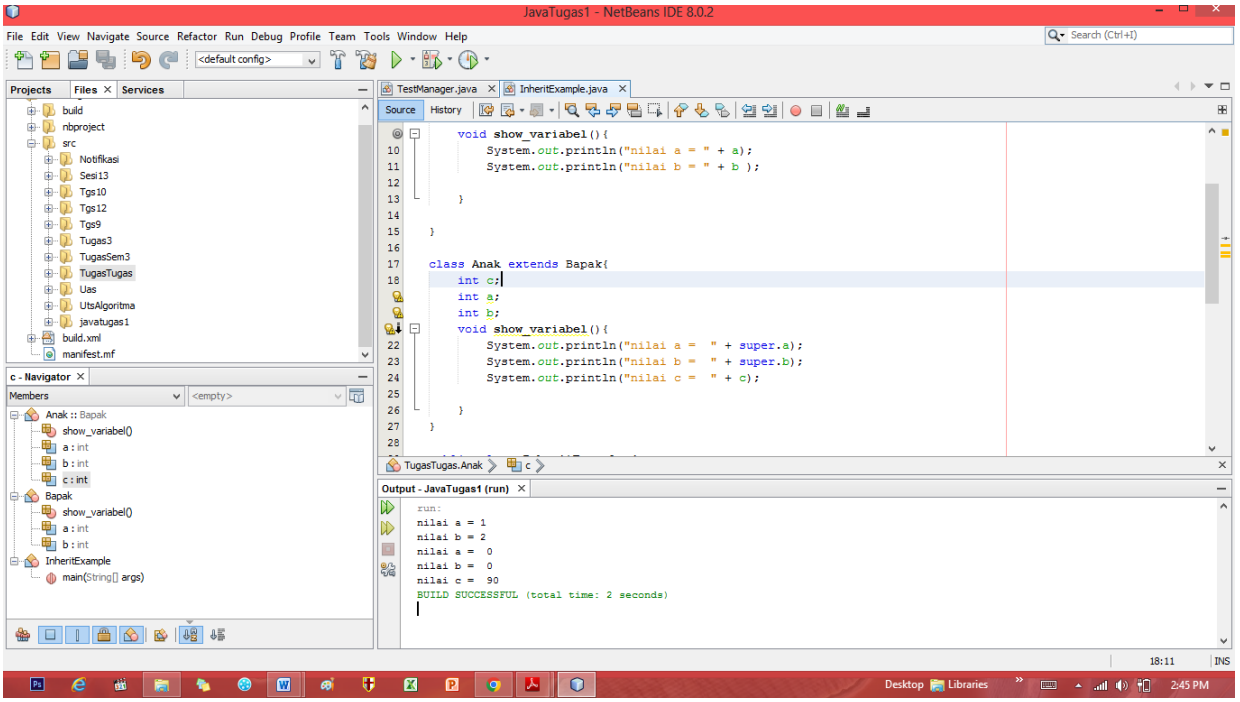
Percobaan 6:

```
class A {
    String var_a = "Variabel A";
    String var_b = "Variabel B";
    String var_c = "Variabel C";
    String var_d = "Variabel D";

    A(){
        System.out.println("Konstruktor A dijalankan");
    }
}

class B extends A{
    B(){
        System.out.println("Konstruktor B dijalankan ");
        var_a = "Var_a dari class B";
        var_b = "Var_a dari class B";
    }

    public static void main(String args[]){

        System.out.println("Objek A dibuat");
        A aa= new A();
        System.out.println("menampilkan nama variabel obyek aa");
        System.out.println(aa.var_a);
        System.out.println(aa.var_b);
        System.out.println(aa.var_c);
        System.out.println(aa.var_d);
        System.out.println("");

        System.out.println("Objek B dibuat");
        B bb= new B();
        System.out.println("menampilkan nama variabel obyek bb");
        System.out.println(bb.var_a);
        System.out.println(bb.var_b);
        System.out.println(bb.var_c);
        System.out.println(bb.var_d);
    }
}
```

Jawab : terdapat 2 kelas yaitu kelas A sebagai parent class dan kelas B sebagai subclassdari 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:



Jawab : 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 “overide” 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 construktor yang akan mengoveride kelas parentnya. this.babyName = babyName untuk passing nilai babyName pada objek dengan parameter contruktor babyName .