

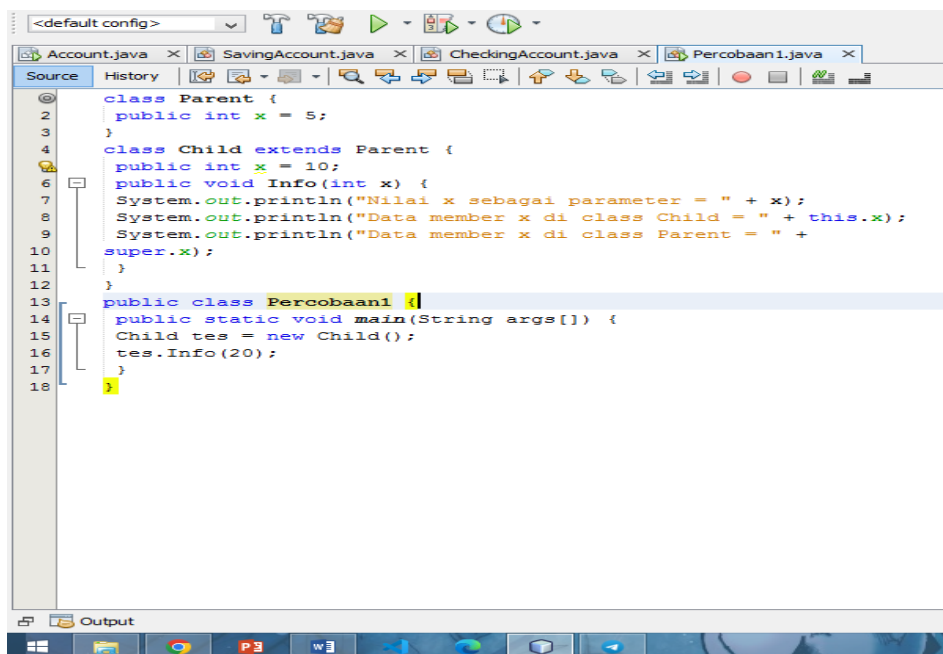
Nama : Andhika Oktasandira  
NIM : 20210040054  
Kelas : TI21A

1. Kode Program upload ke github dengan nama repository praktikum-inheritance
2. Berikan analisa setiap percobaan dalam bentuk File teks pdf dan upload juga ke github praktikum-inheritance

Jawab

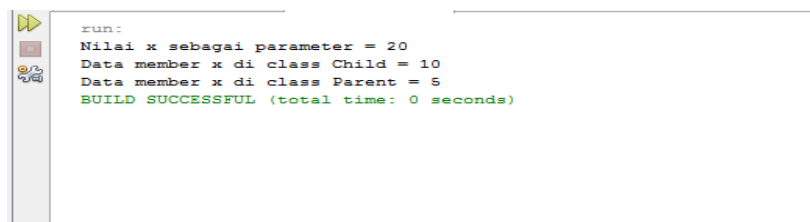
Percobaan berikut ini menunjukkan penggunaan kata kunci “super”.

Input percobaan 1



```
<default config>
Account.java x SavingAccount.java x CheckingAccount.java x Percobaan1.java x
Source History
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 Percobaan1 {
    public static void main(String args[]) {
        Child tes = new Child();
        tes.Info(20);
    }
}
```

Output percobaan 1

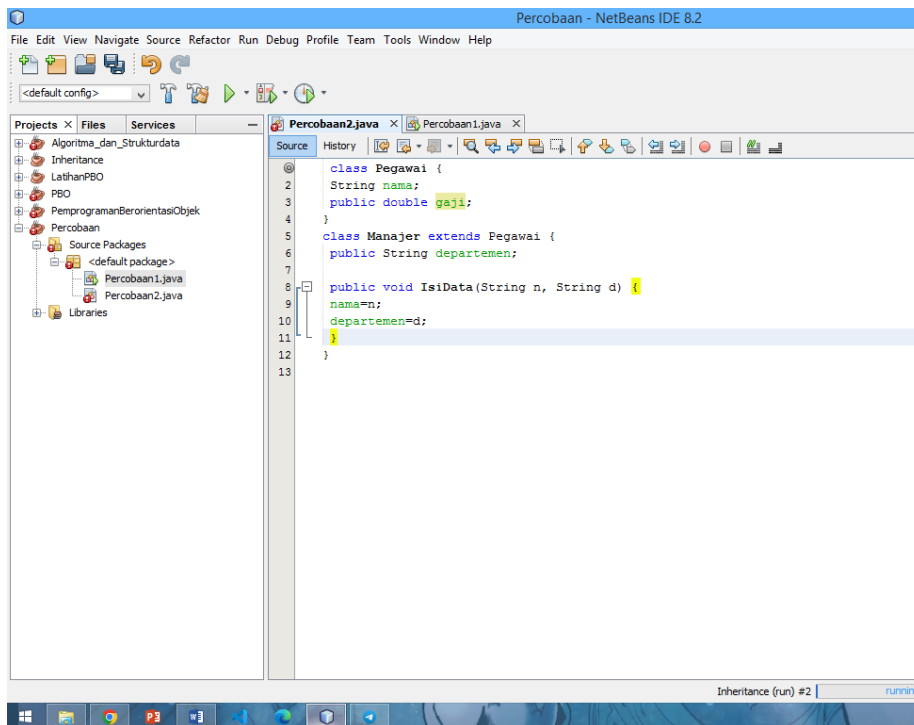


```
run:
Nilai x sebagai parameter = 20
Data member x di class Child = 10
Data member x di class Parent = 5
BUILD SUCCESSFUL (total time: 0 seconds)
```

Pada percobaan pertama ini class parent sebagai induk class yang memiliki atribut integer = 5 , Child sebagai sub class dan didalam class child terdapat sebuah nilai parameter 20 karena ditentukan dari tes.info. dan ada data member dari class parent bernilai 5 kenapa nilainya 5 karena “super” mengambil nilai integer dari class parent

## Percobaan 2

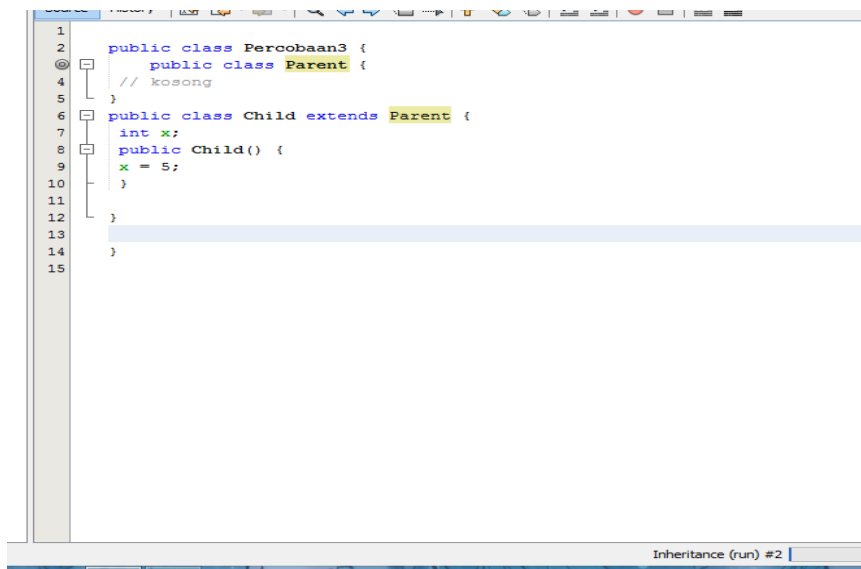
Percobaan berikut ini menunjukkan penggunaan kontrol akses terhadap atribut parent class. Mengapa terjadi error, dan bagaimana solusinya?



Solusi yang saya lakukan yaitu menghapus “public” sebelum kalimat class yang awalnya “public class Pegawai” menjadi class Pegawai begitupun dengan class Manajer dan saya juga menambahkan String nama; agar nama=n; di public void isiData tidak terjadi error.

## Percobaan 3

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

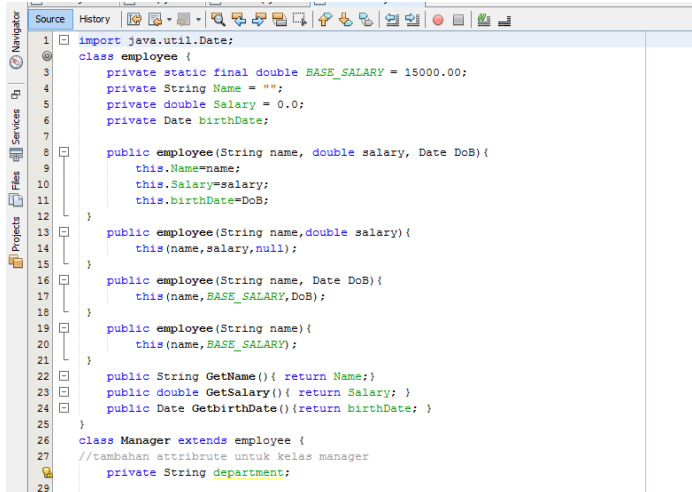


Solusinya saya menambahkan class utama yaitu percobaan3, kenapa costruktur terjadi error karena konstruktur tersebut berada di subclass

## Percobaan 4

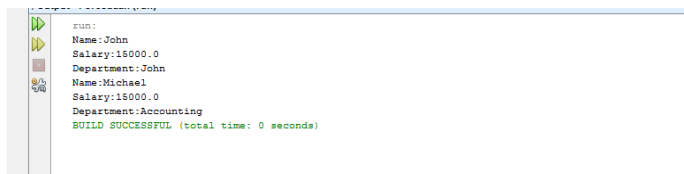
Percobaan berikut ini menunjukkan penggunaan kelas Employee dan subkelas Manager yang merupakan turunannya. Kelas TestManager digunakan untuk menguji

### Input



```
1 import java.util.Date;
2
3 class employee {
4     private static final double BASE_SALARY = 15000.00;
5     private String Name = "";
6     private double Salary = 0.0;
7     private Date birthDate;
8
9     public employee(String name, double salary, Date DoB){
10         this.Name=name;
11         this.Salary=salary;
12         this.birthDate=DoB;
13     }
14     public employee(String name,double salary){
15         this(name,salary,null);
16     }
17     public employee(String name, Date DoB){
18         this(name,BASE_SALARY,DoB);
19     }
20     public employee(String name){
21         this(name,BASE_SALARY);
22     }
23     public String GetName(){ return Name;}
24     public double GetSalary(){ return Salary; }
25     public Date GetbirthDate(){return birthDate; }
26 }
27 class Manager extends employee {
28     //tambahan attribut untuk kelas manager
29     private String department;
```

### Output



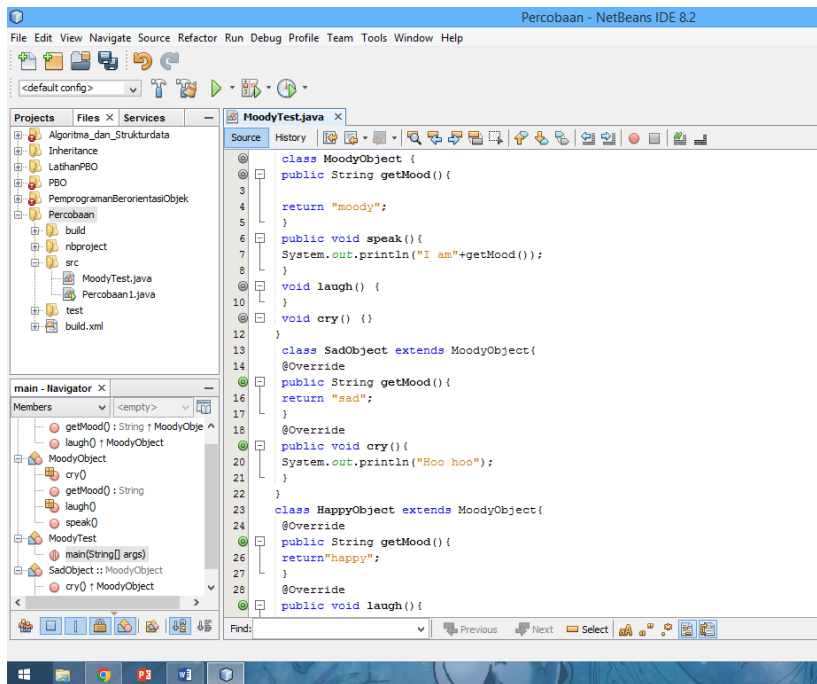
```
run:
Name:John
Salary:15000.0
Department:John
Name:Michael
Salary:15000.0
Department:Accounting
BUILD SUCCESSFUL (total time: 0 seconds)
```

## Percobaan 5

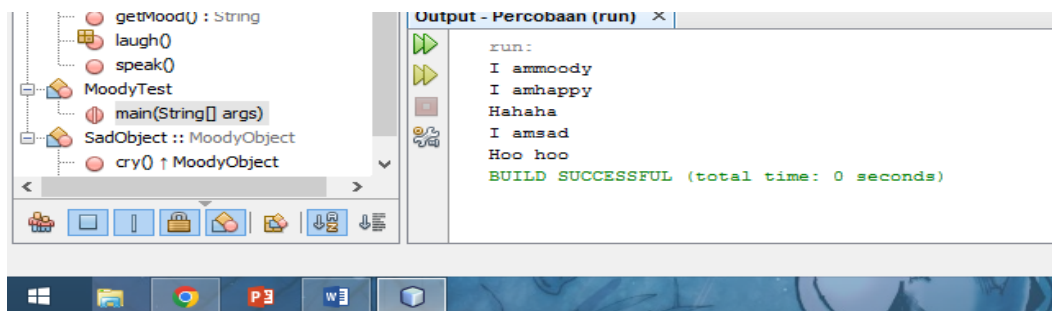
Percobaan berikut ini menunjukkan penggunaan kelas MoodyObject dengan subkelas HappyObject dan SadObject. Kelas MoodyTest digunakan untuk menguji kelas dan subkelas.

- SadObject berisi : sad, method untuk menampilkan pesan, tipe public
- HappyObject berisi : laugh, method untuk menampilkan pesan, tipe public
- MoodyObject berisi :
  1. getMood, memberi nilai mood sekarang, tipe public, return type string
  2. Speak, menampilkan mood, tipe public

input



Output

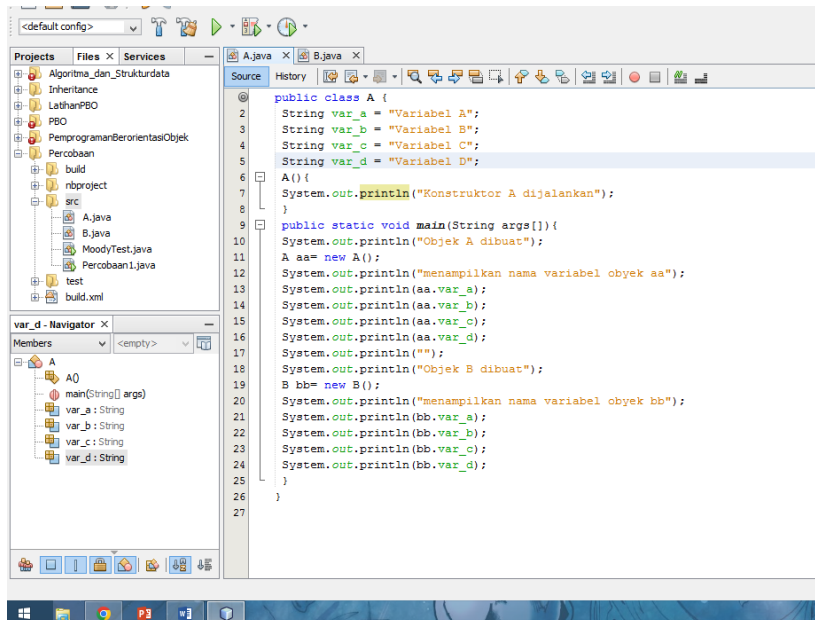


saya memperbaikinya hanya dengan menghapus public di tiap subclass.

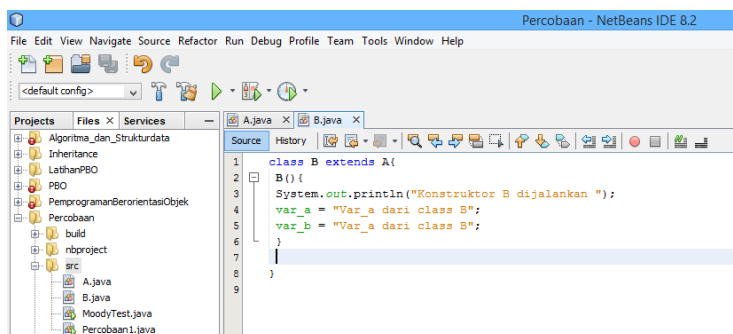
## Percobaan 6

Percobaan berikut ini menunjukkan penggunaan kelas A dan dengan subkelas B. Simpan kedua kelas ini dalam 2 file yang berbeda (A.java dan B.java) dan dalam satu package. Perhatikan proses pemanggilan konstruktor dan pemanggilan variabel

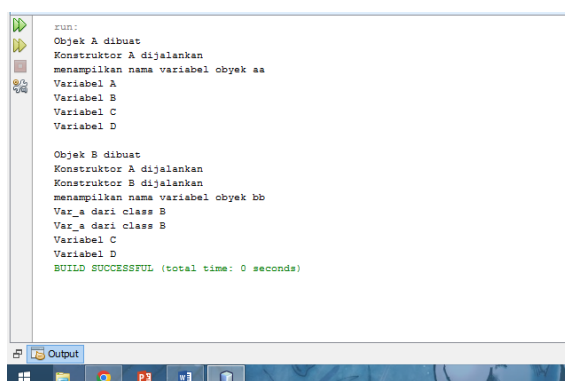
### Input class A.java



### Input class B.java



### Output



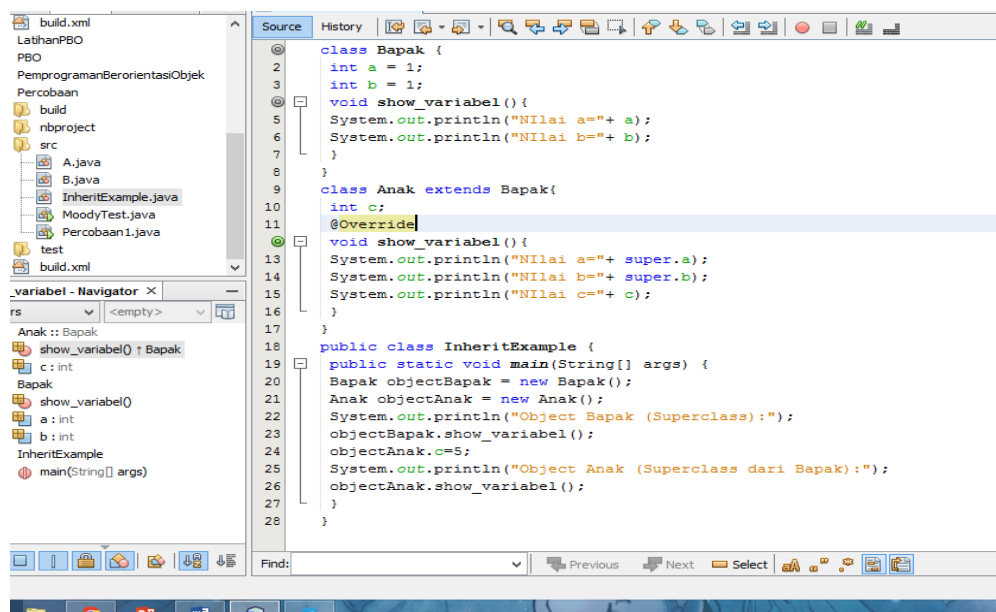
Saya memisahkan subclass B dari kelas A tanpa mengubah apapun dan Cuma sekedar drag and drop dan saya juga sudah memperhatikan konstruktornya.

## Percobaan 7

Percobaan berikut ini menunjukkan penggunaan Inheritance dan Overriding method pada kelas Bapak dan subkelas Anak. Terjadi override pada method `show_variabel`. Perhatikan perubahan nilai pada variabel `a`, `b`, dan `c`.

Kemudian lakukan modifikasi pada method `show_variabel()` pada class Anak. Gunakan `super` untuk menampilkan nilai `a` dan `b` (memanfaatkan method yang sudah ada pada superclass).

input

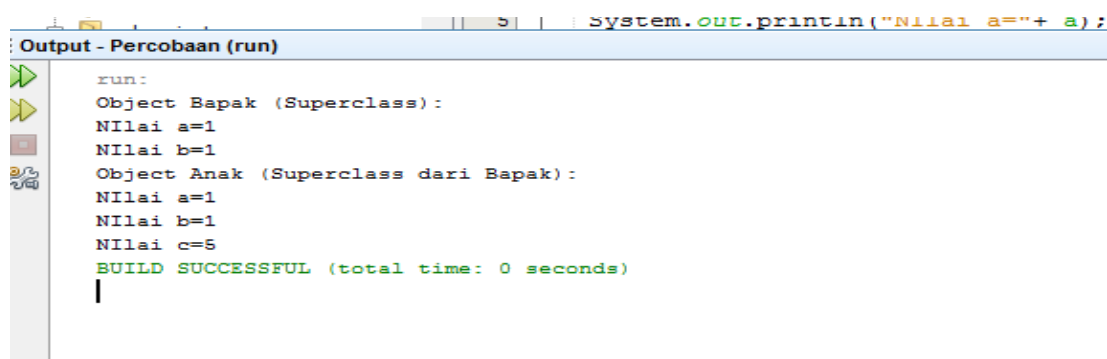


```
class Bapak {
    int a = 1;
    int b = 1;
    void show_variabel() {
        System.out.println("Nilai a=" + a);
        System.out.println("Nilai b=" + b);
    }
}

class Anak extends Bapak {
    int c;
    @Override
    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();
        System.out.println("Object Bapak (Superclass):");
        objectBapak.show_variabel();
        objectAnak.c = 5;
        System.out.println("Object Anak (Superclass dari Bapak):");
        objectAnak.show_variabel();
    }
}
```

Output



```
run:
Object Bapak (Superclass):
Nilai a=1
Nilai b=1
Object Anak (Superclass dari Bapak):
Nilai a=1
Nilai b=1
Nilai c=5
BUILD SUCCESSFUL (total time: 0 seconds)
```

Perbedaan dengan yang sebelum diubah yaitu pada subclass anak nilai `a`, `b` yang mewarisi nilai bapak dan `c` yaitu nilai dari objek si anak atau bukan nilai warisan.

## Percobaan 8

Percobaan berikut ini menunjukkan penggunaan overriding method pada kelas Parent dan subkelas Baby, saat dilakukan pemanggilan konstruktor superclass dengan menggunakan super.

### Input parent

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

### Input baby

```
Parent.java
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");
    }
}
```

### input testbaby

```
TestBaby.java
public class TestBaby {
    public static void main(String args[]) {
        Baby x = new Baby("Hafidza");
        x.Cry();
    }
}
```

## Output

```
run:
Konstruktor Baby
Hafidza
Owek owek
BUILD SUCCESSFUL (total time: 0 seconds)
```

Dengan begitu programnya berjalan dengan lancar