

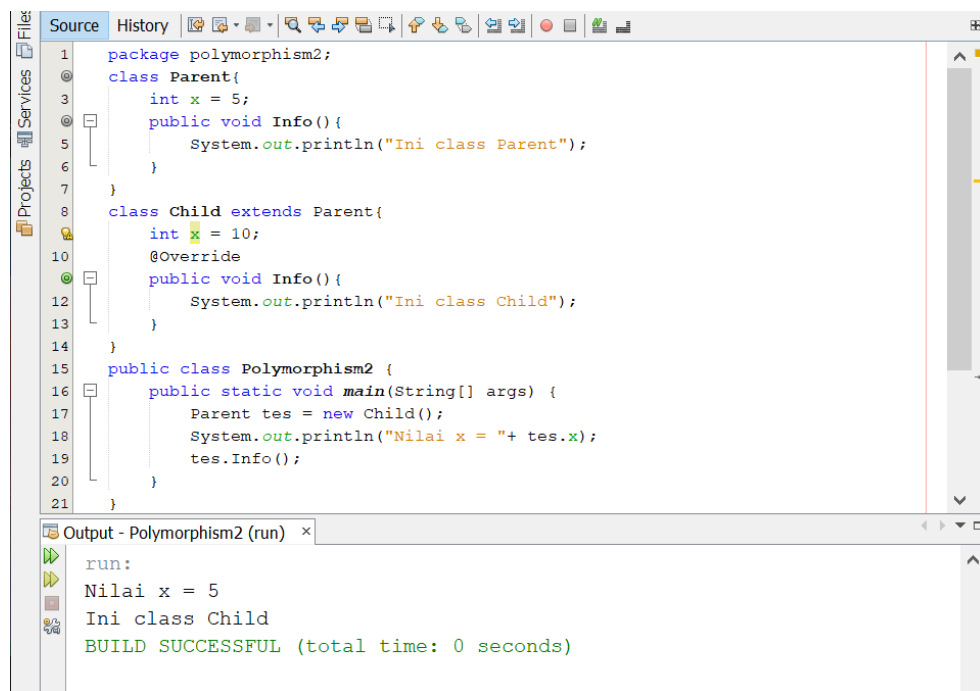
POLYMORPHISM 2

Langkah Percobaan

1. VMI

Virtual Method Invocation (VMI) bisa terjadi jika terjadi polimorfisme dan overriding. Pada saat obyek yang sudah dibuat tersebut memanggil overridden method pada parent class, kompiler Java akan melakukan invocation (pemanggilan) terhadap overriding method pada subclass, dimana yang seharusnya dipanggil adalah overridden method. Berikut contoh terjadinya VMI:

a. Pemanggilan method



```
1 package polymorphism2;
2 class Parent{
3     int x = 5;
4     public void Info(){
5         System.out.println("Ini class Parent");
6     }
7 }
8 class Child extends Parent{
9     int x = 10;
10    @Override
11    public void Info(){
12        System.out.println("Ini class Child");
13    }
14 }
15 public class Polymorphism2 {
16     public static void main(String[] args) {
17         Parent tes = new Child();
18         System.out.println("Nilai x = " + tes.x);
19         tes.Info();
20     }
21 }
```

Output - Polymorphism2 (run)

```
run:
Nilai x = 5
Ini class Child
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Pada saat obyek yang sudah dibuat tersebut memanggil overridden method pada parent class, kompiler Java akan melakukan invocation (pemanggilan) terhadap overriding method pada subclass, dimana yang seharusnya dipanggil adalah overridden method.

b. Pemanggilan constructor

```
package polymorphism2;
class Parent{
    int x = 5;
    public Parent(){
        System.out.println("Konstruktor Parent");
    }
    public void fungsiParent(){
        System.out.println("Fungsi Class Parent");
    }
}
class Child extends Parent{
    int x = 10;
    public Child(){
        System.out.println("Konstruktor Child");
    }
    public void fungsiParent(){
        System.out.println("Fungsi Class Child");
    }
}
public class Polymorphism2 {
    public static void main(String[] args) {
        Parent obj = new Child();
        System.out.println(obj.x);
        obj.fungsiParent();
    }
}
```

Output - Polymorphism2 (run) x

```
run:
Konstruktor Parent
Konstruktor Child
5
Fungsi Class Child
```

2. Polymorphic arguments

Polymorphic arguments adalah tipe suatu parameter yang menerima suatu nilai yang bertipe subclass-nya. Berikut contoh dari polymorphics arguments:

```
Source History
1 package polymorphism2;
2 class food{
3     void eat(){
4         System.out.println("This food is great");
5     }
6 }
7 class pie extends food{
8     @Override
9     void eat(){
10        System.out.println("This pie is great");
11    }
12 }
13 class tuna extends food{
14     @Override
15     void eat(){
16        System.out.println("This tuna is great");
17    }
18 }
19 class fatty{
20     public void taste(food x){
21         x.eat();
22     }
23 }
24 public class PolyArgument {
25     public static void main(String[] args) {
26         fatty bucky = new fatty();
27         food fo = new food();
28         food po = new pie();
29
30         bucky.taste(fo);
31         bucky.taste(po);
32     }
33 }
```

run:

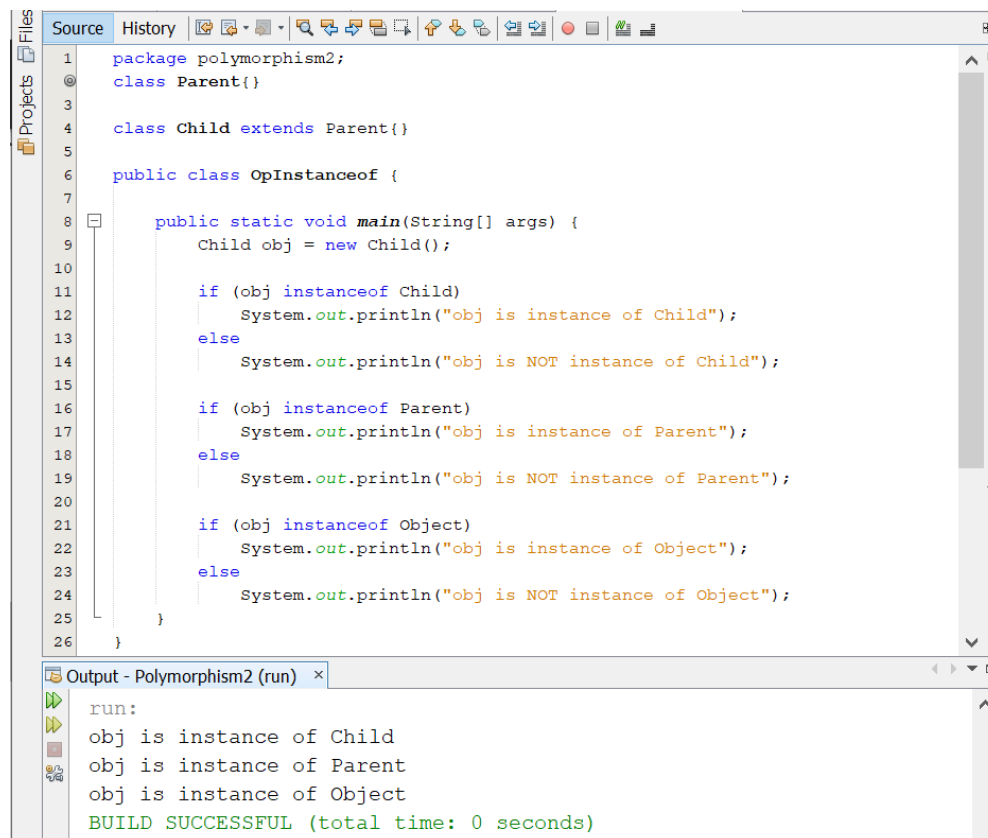
```
This food is great
This pie is great
BUILD SUCCESSFUL (total
```

- Dengan menggunakan polymorphic Arguments kita dapat membuat method yang dapat menerima parent objek dan tetap dapat bekerja dengan baik menggunakan objek-objek dari kelas turunannya. Dengan adanya polymorphic argument kita cukup mendefinisikan satu method saja yang bisa digunakan untuk menangani behavior semua subclass.

3. Operator instanceof

Pernyataan instanceof sangat berguna untuk mengetahui tipe asal dari suatu polymorphic arguments.

a. Parent-child relationship



```

1 package polymorphism2;
2 class Parent{}
3
4 class Child extends Parent{}
5
6 public class OpInstanceof {
7
8     public static void main(String[] args) {
9         Child obj = new Child();
10
11         if (obj instanceof Child)
12             System.out.println("obj is instance of Child");
13         else
14             System.out.println("obj is NOT instance of Child");
15
16         if (obj instanceof Parent)
17             System.out.println("obj is instance of Parent");
18         else
19             System.out.println("obj is NOT instance of Parent");
20
21         if (obj instanceof Object)
22             System.out.println("obj is instance of Object");
23         else
24             System.out.println("obj is NOT instance of Object");
25     }
26 }
  
```

Output - Polymorphism2 (run) x

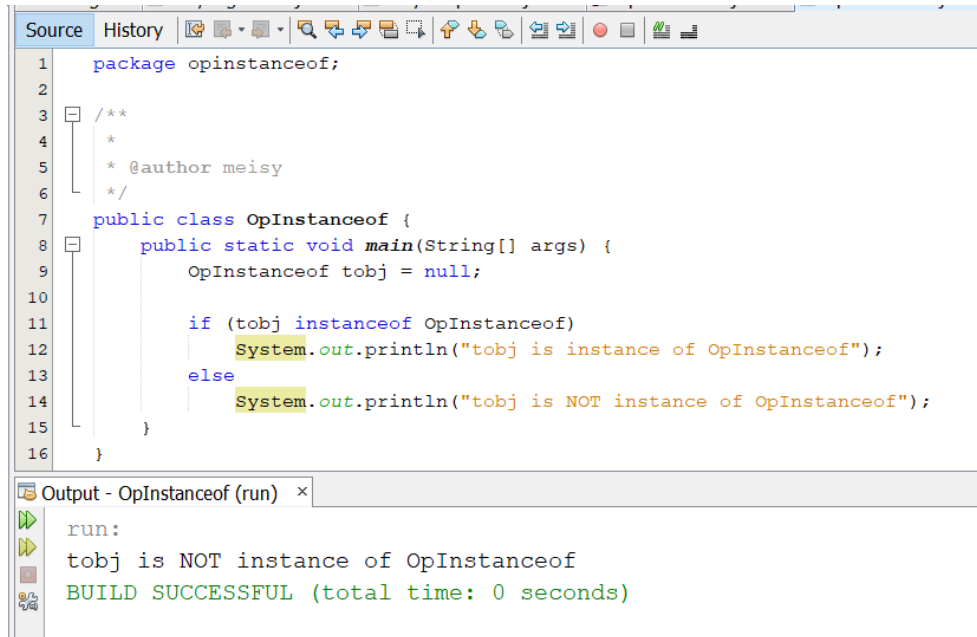
```

run:
obj is instance of Child
obj is instance of Parent
obj is instance of Object
BUILD SUCCESSFUL (total time: 0 seconds)
  
```

- Pada contoh code diatas ,kita membuat parent class dan child class, yang akan kita gunakan sebagai sample percobaan. Didalam method *main*, kita menginstansiasi objek obj pada class *Child()*. Dengan menggunakan statement *System.out.println ()*, kita akan mengecek kondisi apakah obj merupakan instansiasi dari class *Child()*, dengan menggunakan

keyword *instanceof* , kita dapat melihat hasilnya, yang akan menampilkan output berupa nilai *true* atau *false*.

b. Null object



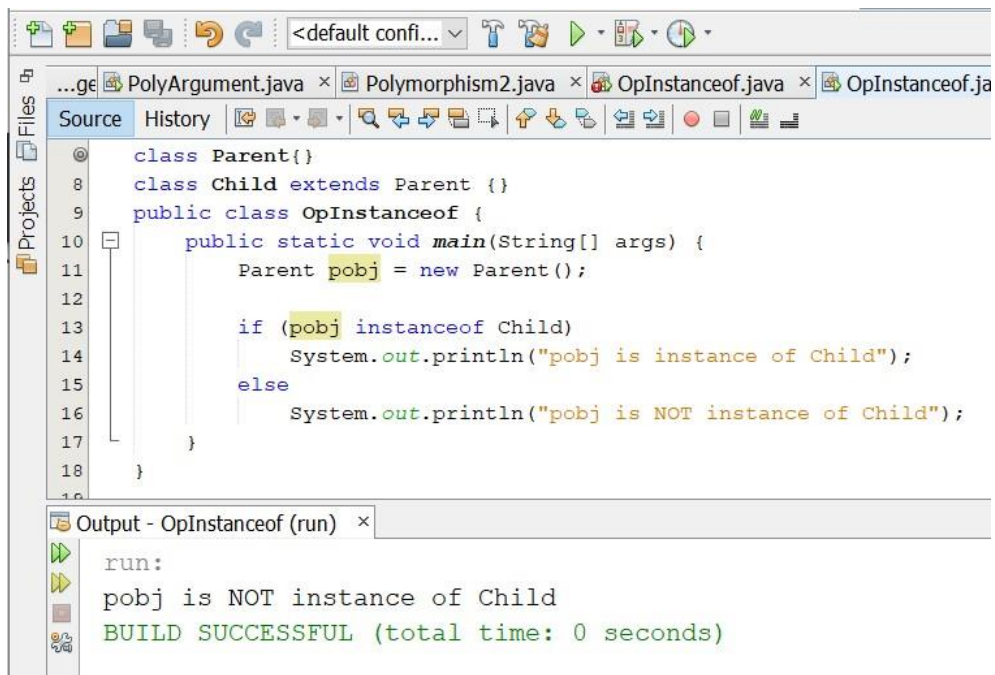
```
1 package opinstanceof;
2
3 /**
4  *
5  * @author meisy
6  */
7 public class OpInstanceof {
8     public static void main(String[] args) {
9         OpInstanceof tobj = null;
10
11         if (tobj instanceof OpInstanceof)
12             System.out.println("tobj is instance of OpInstanceof");
13         else
14             System.out.println("tobj is NOT instance of OpInstanceof");
15     }
16 }
```

Output - OpInstanceof (run) x

```
run:
tobj is NOT instance of OpInstanceof
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Karena obj nya null sehingga tidak instance.

c. Bukan instance dari child class



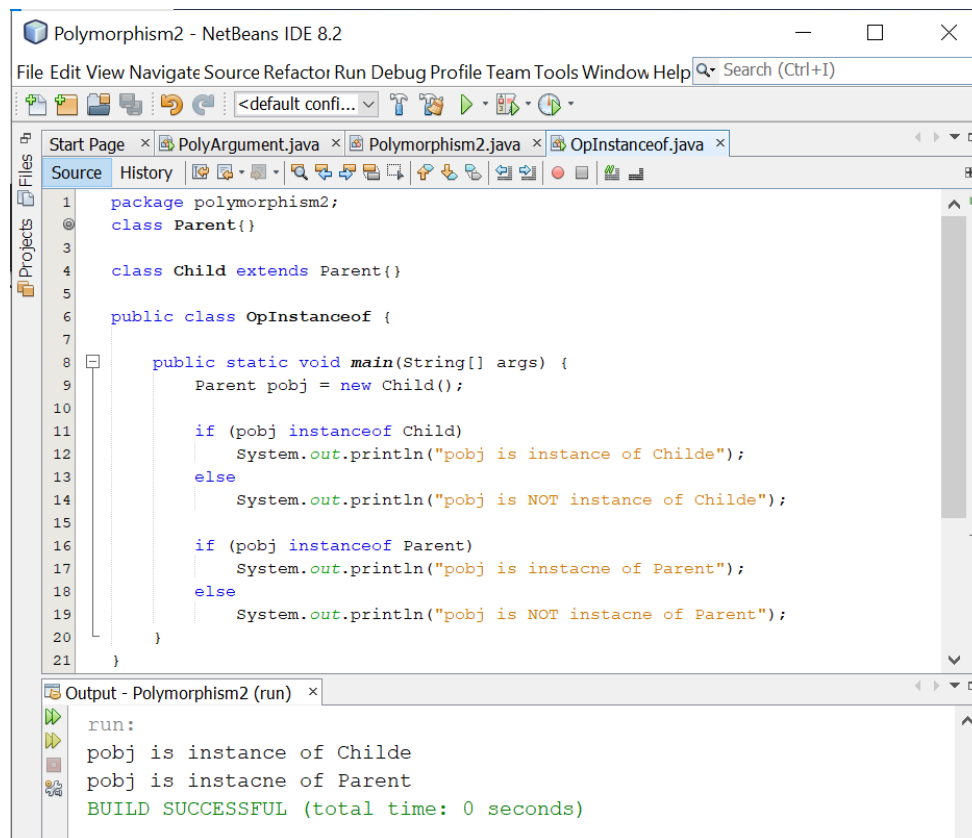
```
1 class Parent{}
2 class Child extends Parent {}
3 public class OpInstanceof {
4     public static void main(String[] args) {
5         Parent pobj = new Parent();
6
7         if (pobj instanceof Child)
8             System.out.println("pobj is instance of Child");
9         else
10            System.out.println("pobj is NOT instance of Child");
11     }
12 }
```

Output - OpInstanceof (run) x

```
run:
pobj is NOT instance of Child
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Karena konstruktor pada main adalah Parent.

d. Instance dari child class (polymorphism)



```
Polymorphism2 - NetBeans IDE 8.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help Search (Ctrl+I)
Start Page x PolyArgument.java x Polymorphism2.java x OpInstanceof.java x
Source History
1 package polymorphism2;
2 class Parent{}
3
4 class Child extends Parent{}
5
6 public class OpInstanceof {
7
8     public static void main(String[] args) {
9         Parent pobj = new Child();
10
11         if (pobj instanceof Child)
12             System.out.println("pobj is instance of Childe");
13         else
14             System.out.println("pobj is NOT instance of Childe");
15
16         if (pobj instanceof Parent)
17             System.out.println("pobj is instacne of Parent");
18         else
19             System.out.println("pobj is NOT instacne of Parent");
20     }
21 }
Output - Polymorphism2 (run) x
run:
pobj is instance of Childe
pobj is instacne of Parent
BUILD SUCCESSFUL (total time: 0 seconds)
```

- Instance dari child class karena konstruktor pada main adalah Child.

BAB IV KESIMPULAN

4.1. Kesimpulan

Kesimpulannya, Polymorphism adalah penggunaan salah satu item seperti fungsi dan atribut pada berbagai jenis objek yang berbeda dalam bahasa pemrograman.

4.2. Saran

Untuk di praktikum tidak ada kesulitan tetapi saya mempunyai kesulitan di bagian membuat laporan lengkap, saran saya yaitu ibu/bapak dosen menjelaskan contoh membuat laporan lengkap yang benar, dari saya pribadi, saya khawatir laporan saya tidak sesuai.

DAFTAR PUSTAKA