LAPORAN RESMI

Praktikum 14 Polimorfisme II

Mata Kuliah: Praktek Pemrograman Berbasis Objek

Disusun oleh:

Bayu Hadi Leksana (3122500046)

2 D3 Teknik Informatika B

Dosen Pengampu: Andhik Ampuh Yunanto S.Kom., M.Kom.

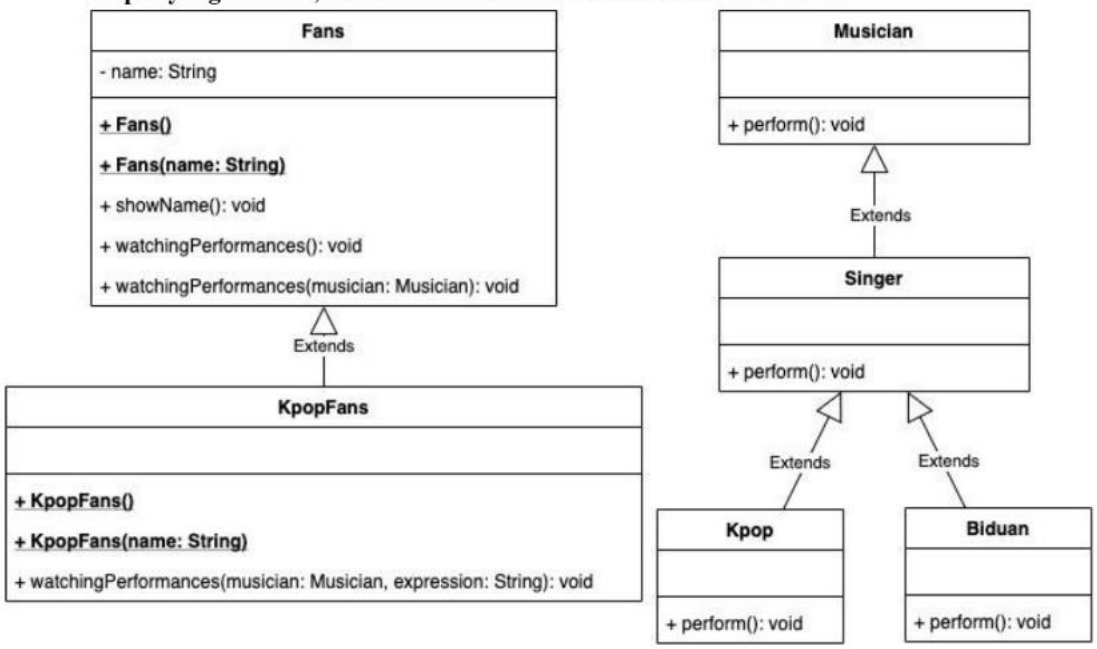
PROGRAM STUDI D3 TEKNIK INFORMATIKA

POLITEKNIK ELEKTRONIKA NEGERI SURABAYA

2023/2024

A. PERCOBAAN

Impelemntasikan class diagram berikut dalam kode program. Hasil harus sesuai output yang diminta, dan class Test tidak boleh diubah/dimodifikasi!



saat method perform dijalankan

Tercetak kalimat berikut:
Performer: Beraksi di atas panggung
Singer: Beraksi diatas panggung, bernyanyi dengan merdu
Kpop: Beraksi diatas panggung, bernyanyi dengan merdu, dan ngedance
Biduan: Beraksi diatas panggung, bernyanyi dengan merdu, dengan cengkok melayu

Source Code:

Musician.java

```
public class Musician {
    public void perform() {
        System.out.print("Beraksi di atas panggung");
    }
}
```

Singer.java

```
public class Singer extends Musician {
    public void perform() {
        super.perform();
        System.out.print(", bernyanyi dengan merdu");
    }
}
```

Kpop.java

```
public class Kpop extends Singer{
    public void perform() {
        super.perform();
        System.out.print(", dan ngedance");
```

```
    }
}
```

Biduan.java

```
public class Biduan extends Singer{
    public void perform() {
        super.perform();
        System.out.print(", dengan cengkok melayu");
    }
}
```

Fans.java

```
public class Fans {
    private String name;
    public Fans() {
        this.name = "noname";
    }
    public Fans(String name) {
        this.name = name;
    }
    public void showName() {
        System.out.print(name);
    }
    public void watchingPerformance() {
        showName();
        System.out.println(": melihat idolanya dari youtube");
    }
    public void watchingPerformance(Musician musician) {
        showName();
        System.out.print(": melihat idolanya ");
        musician.perform();
        System.out.println(" ");
    }
}
```

KpopFans.java

```
public class KpopFans extends Fans {
    public KpopFans() {
        super();
    }
    public KpopFans(String name) {
        super(name);
    }
    public void watchingPerformance(Musician musician, String
expression) {
        showName();
        System.out.print(": ");
        System.out.print(expression);
        System.out.print("melihat idolanya ");
        musician.perform();
        System.out.println(" ");
    }
}
```

Output:

```
C:\Users\bayuh\IdeaProjects\pbo\praktikum14\music\src>java Test
noname: melihat idolanya dari youtube
Mona: melihat idolanya Beraksi di atas panggung
Mona: melihat idolanya Beraksi di atas panggung, bernyanyi dengan merdu
Tomi: melihat idolanya Beraksi di atas panggung, bernyanyi dengan merdu, dengan cengkok melayu
Febi: teriak histerismelihat idolanya Beraksi di atas panggung, bernyanyi dengan merdu, dan ngedance
```

B. LATIHAN

1. Tunjukkan contoh Overloading dalam percobaan diatas!
   Jawab:

```
public Fans() {
    this.name = "noname";
}
public Fans(String name) {
    this.name = name;
}
```

Kode di atas adalah contoh overloading. Pada kode di atas tepatnya adalah overloading pada konstruktor. Jadi, konstruktor pertama memiliki modivier public dan tanpa menerima parameter, lalu akan mengisi variabel name dengan string “noname”. Sedangkan konstruktor kedua sama-sama memiliki modivier public, namun dengan menerima parameter String name, yang kemudian akan mengisi variabel name dengan sesuai parameter yang didapat.

Hal ini juga berlaku pada kode ini:

```
public KpopFans() {
    super();
}
public KpopFans(String name) {
    super(name);
}
```

Bedanya, kode tersebut berada di class KpopFans yang kode dalam konstruktornya diturunkan dari parent classs nya, yakni class Fans.

2. Tunjukkan contoh Overriding method dan overriden method pada percobaan diatas!
   Jawab:

```
public class Musician {
    public void perform() {
        System.out.print("Beraksi di atas panggung");
    }
}
```

```
public class Singer extends Musician {
    public void perform() {
        super.perform();
        System.out.print(", bernyanyi dengan merdu");
    }
}
```

```
public class Kpop extends Singer{
    public void perform() {
        super.perform();
        System.out.print(", dan ngedance");
    }
}
```

```
public class Biduan extends Singer{
    public void perform() {
        super.perform();
        System.out.print(", dengan cengkok melayu");
    }
}
```

Kode di atas adalah contoh dari overriding method dalam beberapa class. Jadi antara class Musician (parent) dengan class Singer (child), overridden method-nya adalah method perform() yang ada di class Musician, sedangkan method perform() pada class Singer adalah yang melakukan override.
Kemudian antara class Singer (parent) dengan class Kpop (child), overridden method-nya adalah method perform() yang ada di class Singer, sedangkan method perform() pada class Kpop adalah yang melakukan override. Begitu juga pada class Biduan (child) yang mana method perform() nya melakukan override dari class Singer (parent).

3. Tunjukkan contoh Overloding yang terjadi dalam satu class, pada percobaan diatas!
   Jawab:

```
public void watchingPerformance() {
    showName();
    System.out.println(": melihat idolanya dari youtube");
}
public void watchingPerformance(Musician musician) {
    showName();
    System.out.print(": melihat idolanya ");
    musician.perform();
    System.out.println(" ");
}
```

Kode di atas adalah contoh overloading dalam satu class, yakni class Fans. Pada class Fans, terdapat dua method watchingPerformance() yang sama-sama memiliki modifier public dan return type void. Namun pada method watchingPerformance() yang kedua menerima satu parameter, yakni musician yang bertipe Musician. Kedua method itu juga menjalankan kode yang berbeda.

4. Tunjukkan contoh Overloading yang terjadi antara superclass dan subclass pada percobaan diatas!
   Jawab:

```
public class KpopFans extends Fans {
    public KpopFans() {
        super();
    }
    public KpopFans(String name) {
        super(name);
    }
```

```
    public void watchingPerformance(Musician musician, String
expression) {
        showName();
        System.out.print(": ");
        System.out.print(expression);
        System.out.print("melihat idolanya ");
        musician.perform();
        System.out.println(" ");
    }
}
```

Kode di atas adalah kode pada class KpopFans yang merupakan child class dari class Fans. Pada class ini terjadi overloading antara parent/superclass dengan subclass. Sudah diketahui sebelumnya bahwa di class Parent terdapat dua method watchingPerformance(), yang satu tanpa parameter dan yang kedua dengan parameter. Kemudian pada class KpopFans ini melakukan overloading lagi untuk method watchingPerformance() menjadi terdapat dua parameter, yakni Musician musician dan String expression. Lalu method watchingPerformance() yang ketiga ini menjalankan kode yang berbeda juga dengan memanfaatkan parameter expression nya.

5. Tunjukkan contoh Virtual Method Invocation yang terjadi pada percobaan diatas!
   Jawab:

```
Fans fans3 = new KpopFans("Tomi");
fans3.watchingPerformance(new Biduan());
```

Kode di atas, adalah contoh Virtual Method Invocation. Jadi saat method watchingPerformance() pada objek fans3 dipanggil, maka akan menjalankan method watchingPerformance() pada class Fans. Hal ini karena meskipun instansiasinya nya menggunakan class KpopFans, namun pada class KpopFans tidak melakukan override method watchingPerformance() dari class parent nya (Fans). Sehingga yang dijalankan adalah watchingPerformance() pada class Fans. Kecuali jika class KpopFans melakukan override maka yang akan dijalankan adalah method watchingPerformance() pada class KpopFans.

6. Tunjukkan contoh Polimorfism pada percobaan diatas!
   Jawab:

```
Fans fans3 = new KpopFans("Tomi");
```

Kode di atas adalah contoh polimorfism yang ada. Objek fans3 memiliki tipe referensi Fans, tetapi sebenarnya dibuat sebagai objek dari kelas turunan, yaitu KpopFans. Ini disebut polimorfisme, di mana objek dari kelas turunan dapat diakses melalui referensi kelas dasar.

7. Tunjukkan contoh inheritance pada percobaan diatas!
   Jawab:

```
public class KpopFans extends Fans {
    public KpopFans() {
        super();
    }
```

```
    public KpopFans(String name) {
        super(name);
    }
    public void watchingPerformance(Musician musician, String
expression) {
        showName();
        System.out.print(": ");
        System.out.print(expression);
        System.out.print("melihat idolanya ");
        musician.perform();
        System.out.println(" ");
    }
}
```

```
public class Singer extends Musician {
    public void perform() {
        super.perform();
        System.out.print(", bernyanyi dengan merdu");
    }
}
```

```
public class Kpop extends Singer{
    public void perform() {
        super.perform();
        System.out.print(", dan ngedance");
    }
}
```

```
public class Biduan extends Singer{
    public void perform() {
        super.perform();
        System.out.print(", dengan cengkok melayu");
    }
}
```

Kode-kode di atas adalah contoh inheritance. Class KpopFans merupakan turunan dari class Fans. Class Singer merupakan turunan dari class Musician. Lalu class Kpop dan class Biduan merupakan turunan dari class Singer.

C. KESIMPULAN

Polimorfisme dalam pemrograman berorientasi objek (OOP) adalah konsep yang memungkinkan objek dari kelas yang berbeda untuk diakses dan dimanipulasi menggunakan antarmuka umum, seperti melalui tipe kelas dasar atau antarmuka. Ini menciptakan fleksibilitas dalam desain, memungkinkan penggunaan objek kelas turunan tanpa harus peduli dengan tipe objek yang sebenarnya. Dua bentuk umum polimorfisme adalah polimorfisme compile-time (melalui overloading) dan polimorfisme runtime (melalui overriding), yang memungkinkan pemanggilan metode yang tepat sesuai dengan tipe objek aktual pada waktu eksekusi program. Polimorfisme meningkatkan modularitas, reusabilitas, dan perluasan kode dalam paradigma OOP.