TUGAS PERTEMUAN 4

Disusun untuk memenuhi tugas mata kuliah Pemrograman Berorientasi Objek (Teori)

Disusun oleh:

Bandyaga Adiansyah Sugandi

NIM 231511037

2B – D3

Jurusan Teknik Komputer dan Informatika

Politeknik Negeri Bandung

2024

A. Penugasan

1) Carilah sebuah contoh kode program Java yang memuat relasi antar kelas melalui:
    a. Dependency
    b. Aggregation
    c. Inheritance
2) Jelaskan masing-masing relasi tersebut pada kasus yang dibuat!

B. Source Code

1) Dependency

```
import java.util.Scanner; // dependency dari scanner

class Printer {
    public void print(String namaUser) {
        System.out.println("Username: " + namaUser);
    }
}

class User {
    public void createUsername() {
        Scanner scanner = new Scanner(System.in); // memanggil
dependency scanner
        System.out.print("Masukkan nama Anda: ");
        String namaUser = scanner.nextLine();

        Printer printer = new Printer(); // dependency dari class
Printer
        printer.print(namaUser); // memanggil dependency Printer
    }
}

public class Dependency {
    public static void main(String[] args) {
        User user = new User();
        user.createUsername();
    }
}
```

2) Aggregation

```
class Engine {
    public void start() {
        System.out.println("Mesin mobil telah hidup");
    }
}

class Car {
    private Engine engine;

    public Car(Engine engine) {
        this.engine = engine; // Aggregation
    }

    public void startCar() {
        engine.start();
```

```
        System.out.println("Mobil berjalan.");
    }
}

public class Aggregation {
    public static void main(String[] args) {
        Engine engine = new Engine();
        Car car = new Car(engine);
        car.startCar();
    }
}
```

3) Inheritance

```
class Animal {
    public void makeSound() {
        System.out.println("Suara: ");
    }
}

class Dog extends Animal { // Inheritance
    @Override
    public void makeSound() {
        System.out.println("Anjing: Guk guk!");
    }
}

class Cat extends Animal {
    @Override
    public void makeSound() {
        System.out.println("Kucing: Meow meow!");
    }
}

public class Inheritance {
    public static void main(String[] args) {
        Animal sound = new Animal();
        Dog dog = new Dog();
        Cat cat = new Cat();

        sound.makeSound();
        dog.makeSound();
        cat.makeSound();
    }
}
```

C. Penjelasan

1) Dependency
   - ⇨ Dependency adalah sebuah cara yang menunjukkan bahwa suatu class dapat menggunakan class lain. Relasi ini menggambarkan **"uses-a".** Class yang memiliki dependency tidak memiliki kepemilikan terhadap objek pada class lain, dan hanya menggunakan objek tersebut secara sementara.
   - ⇨ Pada contoh di atas, class User memiliki dependency terhadap Printer untuk mencetak username pengguna. Pengguna menginput username pada class User

menggunakan Scanner yang juga merupakan dependency bawaan **java.util.Scanner** yang diimport.

2) Aggregation
   - ⇨ Aggregation adalah relasi yang menunjukkan bahwa suatu kelas terdiri dari kelas lain sebagai bagian dari keseluruhan. Relasi ini menggambarkan hubungan **"has-a"** tetapi dengan kepemilikan yang lemah. Objek dari kelas lain dapat berdiri sendiri tanpa terikat dengan objek dari kelas yang mengagregasi.
   - ⇨ Pada contoh di atas, class Car memiliki agregasi dengan class Engine. Namun class Engine tetap dapat berdiri sendiri tanpa kebergantungan pada class Car.
3) Inheritance
   - ⇨ Inheritance adalah relasi dimana suatu class merupakan turunan (subclass) dari kelas lain (superclass). Relasi ini menggambarkan **"is-a"** dan memungkinkan subclass untuk mewarisi perilaku (behavior) dari superclass.
   - ⇨ Pada contoh di atas, class Dog dan Cat adalah turunan dari class Animal yang dilakukan extends. Kemudian menggunakan metode makeSound() yang sudah di-override untuk memberikan perilaku (behavor) khusus untuk membuat suara anjing dan kucing.

D. Referensi dan Github

https://www.javatpoint.com/aggregation-in-java
https://www.w3schools.com/java/java_inheritance.asp
https://coursepress.lnu.se/courses/object-oriented-analysis-and-design/02-theory/design_class_relations
https://github.com/basganajaah/Pemrograman-Berorientasi-Objek---Praktek