**LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : RPL

Kelas : 4IA14

Praktikum ke- : 2

Tanggal : 14

Materi : oop

NPM : 51421599

Nama : Haziq Duha Zainul Ihsan

Ketua Asisten : aji

Jumlah Lembar :



**LABORATORIUM TEKNIK INFORMATIKA**

**UNIVERSITAS GUNADARMA**

**2023**

**1. Penjelasan Konsep-Kosep Utama dalam OOP**

1. **Encapsulation (Enkapsulasi)**  
   Enkapsulasi adalah konsep dalam OOP di mana data dan fungsi yang bekerja pada data tersebut disatukan dalam satu unit yang disebut **kelas**. Enkapsulasi memungkinkan data hanya dapat diakses melalui metode yang disediakan oleh kelas, sehingga melindungi data dari modifikasi langsung.  
   **Contoh dalam Java:**

public class Mahasiswa {

private String nama; // Properti private

// Constructor

public Mahasiswa(String nama) {

this.nama = nama;

}

// Getter untuk mengakses nama

public String getNama() {

return nama;

}

// Setter untuk mengubah nama

public void setNama(String nama) {

this.nama = nama;

}

}

1. **Inheritance (Pewarisan)**  
   Inheritance adalah konsep OOP yang memungkinkan sebuah kelas (subclass) untuk mewarisi atribut dan metode dari kelas lain (superclass). Dengan pewarisan, kita dapat membuat kelas baru berdasarkan kelas yang sudah ada dan menambah atau mengganti fitur sesuai kebutuhan.  
   **Contoh dalam Java:**

// Kelas Superclass

public class Hewan {

public void makan() {

System.out.println("Hewan sedang makan.");

}

}

// Kelas Subclass

public class Kucing extends Hewan {

public void suara() {

System.out.println("Kucing mengeluarkan suara meong.");

}

}

1. **Polymorphism (Polimorfisme)**  
   Polimorfisme adalah konsep di mana satu metode atau fungsi dapat memiliki banyak bentuk, memungkinkan satu nama fungsi yang sama digunakan untuk kelas-kelas yang berbeda. Polimorfisme dapat dicapai dengan **overriding** (metode di subclass menggantikan metode di superclass) dan **overloading** (dua metode atau lebih memiliki nama yang sama dalam kelas yang sama, tetapi dengan parameter berbeda).  
   **Contoh dalam Java:**

public class Hewan {

public void suara() {

System.out.println("Hewan membuat suara.");

}

}

public class Anjing extends Hewan {

@Override

public void suara() {

System.out.println("Anjing menggonggong.");

}

}

public class Kucing extends Hewan {

@Override

public void suara() {

System.out.println("Kucing mengeong.");

}

}

**2. Program Sederhana yang Menggunakan Konsep OOP**

Berikut adalah contoh program sederhana yang mengimplementasikan Encapsulation, Inheritance, dan Polymorphism.

// Kelas Utama

public class Main {

public static void main(String[] args) {

// Enkapsulasi

Mahasiswa mahasiswa = new Mahasiswa("Haziq");

System.out.println("Nama Mahasiswa: " + mahasiswa.getNama());

// Pewarisan dan Polimorfisme

Hewan hewan1 = new Kucing();

Hewan hewan2 = new Anjing();

hewan1.suara(); // Output: Kucing mengeong.

hewan2.suara(); // Output: Anjing menggonggong.

}

}

// Kelas Mahasiswa untuk Enkapsulasi

class Mahasiswa {

private String nama;

public Mahasiswa(String nama) {

this.nama = nama;

}

public String getNama() {

return nama;

}

public void setNama(String nama) {

this.nama = nama;

}

}

// Kelas Hewan (Superclass)

class Hewan {

public void suara() {

System.out.println("Hewan membuat suara.");

}

}

// Kelas Kucing (Subclass yang menggunakan pewarisan dan polimorfisme)

class Kucing extends Hewan {

@Override

public void suara() {

System.out.println("Kucing mengeong.");

}

}

// Kelas Anjing (Subclass yang menggunakan pewarisan dan polimorfisme)

class Anjing extends Hewan {

@Override

public void suara() {

System.out.println("Anjing menggonggong.");

}

}

**Penjelasan Program**

* **Enkapsulasi**: Pada kelas Mahasiswa, atribut nama bersifat private, dan hanya bisa diakses melalui metode getter dan setter.
* **Pewarisan**: Kelas Kucing dan Anjing merupakan subclass dari kelas Hewan, yang mewarisi metode suara().
* **Polimorfisme**: Kelas Kucing dan Anjing meng-override metode suara() dari kelas Hewan, sehingga saat metode suara() dipanggil dari objek Hewan, outputnya disesuaikan dengan kelas sebenarnya (Kucing atau Anjing).

Output

