Nama : Rakhadinar Jaladara
NPM : 51421246
Kelas : 4IA17

M2 Laporan Akhir Rekayasa Perangkat Lunak 2

1. Jelaskan konsep - konsep utama dalam Object-Oriented Programming (OOP) seperti Encapsulation, Inheritance, dan Polymorphism. Berikan contoh penerapan setiap konsep ini dalam program Java.

   a. Encapsulation (Enkapsulasi)
   Encapsulation adalah konsep yang menggabungkan data (atribut) dan metode yang bekerja pada data tersebut menjadi satu unit yang disebut class. Enkapsulasi menyembunyikan detail implementasi dari pengguna dan hanya menyediakan antarmuka publik untuk berinteraksi dengan data.
   Contoh Encapsultion:

```
public class Mahasiswa {
    private String nama;
    private String npm;
    private int umur;

    public Mahasiswa(String nama, String npm, int umur) {
        this.nama = nama;
        this.npm = npm;
        this.umur = umur;
    }

    // Getter dan Setter untuk mengakses dan mengubah nilai atribut
    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public String getNpm() {
        return npm;
    }

    public void setNpm(String npm) {
        this.npm = npm;
    }

    public int getUmur() {
        return umur;
    }

    public void setUmur(int umur) {
        this.umur = umur;
```

```
    }
}
```

b. Inheritance (Pewarisan)
Inheritance adalah konsep di mana sebuah class (subclass atau anak class) mewarisi atribut dan metode dari class lain (superclass atau induk class). Ini memungkinkan reuse kode dan menciptakan hierarki class.

Contoh Inheritance:

```
public class MahasiswaSarjana extends Mahasiswa {
    private String jurusan;

    public MahasiswaSarjana(String nama, String npm, int umur, String jurusan) {
        super(nama, npm, umur); // Memanggil konstruktor superclass
        this.jurusan = jurusan;
    }

    public String getJurusan() {
        return jurusan;
    }

    public void setJurusan(String jurusan) {
        this.jurusan = jurusan;
    }

    @Override
    public void tampilkanData() {
        super.tampilkanData(); // Memanggil metode superclass
        System.out.println("Jurusan: " + jurusan);
    }
}
```

c. Polymorphism (Polimorfisme)
Polymorphism adalah kemampuan sebuah metode untuk berperilaku berbeda berdasarkan objek yang memanggilnya. Ini dapat dicapai melalui metode overriding atau metode overloading.

Contoh Polymorphism:

```
public class Main {
    public static void main(String[] args) {
        Mahasiswa mahasiswa = new Mahasiswa("Andi", "51421234", 20);
        MahasiswaSarjana mahasiswaSarjana = new MahasiswaSarjana("Budi",
"51421235", 21, "Informatika");

        tampilkanData(mahasiswa);
        tampilkanData(mahasiswaSarjana);
    }

    public static void tampilkanData(Mahasiswa mahasiswa) {
```

```
        mahasiswa.tampilkanData(); // Memanggil metode yang sesuai berdasarkan tipe objek
    }
}
```

2. Buatlah sebuah program yang memanfaatkan konsep-konsep OOP dan berikan penjelasannya
   Program manajemen kendaraan
   Main.java:

Main.java

```java
package la2_51421246;

public class Main {
    public static void main(String[] args) {
        Vehicle car = new Car("Toyota", "Camry", 2020, 4);
        Vehicle motorcycle = new Motorcycle("Harley-Davidson", "Sportster", 2019, false);

        displayVehicleInfo(car);
        System.out.println();
        displayVehicleInfo(motorcycle);
    }

    // Metode untuk menampilkan informasi kendaraan
    public static void displayVehicleInfo(Vehicle vehicle) {
        vehicle.displayInfo(); // Polymorphism: memanggil metode displayInfo() sesuai tipe objek
    }
}
```

   Vehicle.java:

Vehicle.java

```java
package la2_51421246;

public class Vehicle {
    private String brand;
    private String model;
    private int year;

    public Vehicle(String brand, String model, int year) {
        this.brand = brand;
        this.model = model;
        this.year = year;
    }

    public void displayInfo() {
        System.out.println("Brand: " + brand);
        System.out.println("Model: " + model);
        System.out.println("Year: " + year);
    }
}
```

Car.java:

Car.java

```java
package la2_51421246;

public class Car extends Vehicle {
    private int doors;

    public Car(String brand, String model, int year, int doors) {
        super(brand, model, year);
        this.doors = doors;
    }

    @Override
    public void displayInfo() {
        super.displayInfo(); // Memanggil metode displayInfo() dari kelas Vehicle
        System.out.println("Doors: " + doors);
    }
}
```

Motorcycle.java:

Motorcycle.java

```java
package la2_51421246;

public class Motorcycle extends Vehicle {
    private boolean hasSidecar;

    public Motorcycle(String brand, String model, int year, boolean hasSidecar) {
        super(brand, model, year);
        this.hasSidecar = hasSidecar;
    }

    @Override
    public void displayInfo() {
        super.displayInfo(); // Memanggil metode displayInfo() dari kelas Vehicle
        System.out.println("Has Sidecar: " + hasSidecar);
    }
}
```

Output:

```
PS C:\Users\rkhdi\OneDrive\Desktop\RPL\M2>  & 'C
rs\rkhdi\AppData\Roaming\Cursor\User\workspaceSt
Brand: Toyota
Model: Camry
Year: 2020
Doors: 4

Brand: Harley-Davidson
Model: Sportster
Year: 2019
Has Sidecar: false
PS C:\Users\rkhdi\OneDrive\Desktop\RPL\M2>
```

Penjelasan:

1. Encapsulation (Enkapsulasi):

    - Kelas Vehicle, Car, dan Motorcycle masing-masing memiliki atribut privat dan metode publik untuk mengakses dan memodifikasi data tersebut.

    - Atribut brand, model, dan year di kelas Vehicle diakses melalui metode displayInfo.

2. Inheritance (Pewarisan):

    - Kelas Car dan Motorcycle adalah subclass dari Vehicle, mewarisi atribut dan metode dari kelas Vehicle.

    - Subclass Car dan Motorcycle menambahkan atribut khusus (doors dan hasSidecar) dan mengoverride metode displayInfo untuk menyertakan informasi tambahan.

3. Polymorphism (Polimorfisme):

    - Dalam metode displayVehicleInfo, objek vehicle dari tipe Vehicle dapat merujuk ke objek Car atau Motorcycle.

    - Metode displayInfo yang dipanggil sesuai dengan tipe objek yang sebenarnya (objek Car atau Motorcycle), menunjukkan perilaku polimorfik.