Experiment No. 3
Implement a program that demonstrates the concepts of class and objects
Date of Performance:
Date of Submission:

Aim: Implement a program that demonstrates the concepts of class and objects

Objective: To develop the ability of converting real time entity into objects and create their classes.

Theory:

A class is a user defined blueprint or prototype from which objects are created. It represents the set of properties i.e., members and methods that are common to all objects of one type. In general, class declarations can include these components, in order:

1. Modifiers: A class can be public or has default access.
2. class keyword: class keyword is used to create a class.
3. Class name: The name should begin with a initial letter (capitalized by convention).
4. Superclass (if any): The name of the class's parent (superclass), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.
5. Interfaces (if any): A comma-separated list of interfaces implemented by the class, if any, preceded by the keyword implements. A class can implement more than one interface.
6. Body: The class body surrounded by braces, {}.

An OBJECT is a basic unit of Object-Oriented Programming and represents the real-life entities. A typical Java program creates many objects, which interact by invoking methods. An object consists of:

1. State: It is represented by attributes of an object. It also reflects the properties of an object.
2. Behavior: It is represented by methods of an object. It also reflects the response of an object with other objects.

3. Identity: It gives a unique name to an object and enables one object to interact with other object.

Code:

```
class Bike {
    String brand;
    String color;

    void displayInfo() {
        System.out.println("Brand: " + brand);
        System.out.println("Color: " + color);
    }
}

public class Main {
    public static void main(String[] args) {
        Bike bike1 = new Bike();
        Bike bike2 = new Bike();
        Bike bike3 = new Bike();

        bike1.brand = "Harley-Davidson";
        bike1.color = "Black";
        bike2.brand = "Ducati";
        bike2.color = "Red";
        bike3.brand = "Kawasaki";
        bike3.color = "Green";

        System.out.println("Bike 1 Information:");
        bike1.displayInfo();
        System.out.println("\nBike 2 Information:");
        bike2.displayInfo();
        System.out.println("\nBike 3 Information:");
        bike3.displayInfo();
    }
}
```

Output:

```
C:\Users\admin\Desktop\JL>java Main.java
Bike 1 Information:
Brand: Harley-Davidson
Color: Black

Bike 2 Information:
Brand: Ducati
Color: Red

Bike 3 Information:
Brand: Kawasaki
Color: Green
```

Conclusion:

Comment on how you create a class template and their objects.

→ To create a class template in Java, you define a class using the `class` keyword, specifying its attributes and methods. For example, a `Bike` class might include fields like `brand` and `color` and a method like `displayInfo()`. To create objects, instantiate the class with the `new` keyword, such as `Bike bike1 = new Bike();`, and then set attributes and invoke methods to interact with these objects. This approach models real-world entities, allowing you to manage and manipulate their state and behaviors within your code