Vidyavardhini's College of Engineering and Technology
Department of Artificial Intelligence & Data Science

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

Experiment No. 3

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 objects.

Code:

```
class Bike1 {

   Bike1()

   {

      System.out.println("Bike is created");

   }

   public static void main(String args[]) {

      Bike1 b = new Bike1();

   }

}
```

OUTPUT:

```
PS C:\Users\mynam> cd "c:\Users\mynam\Downloads\java-new-main\java-new-main\JavaFiles-main\JavaFiles-main\s-59(se)\" ; if ($?) { javac Bike1.java } ; if ($?) { java Bike1 }
Bike is created
PS C:\Users\mynam\Downloads\java-new-main\java-new-main\JavaFiles-main\JavaFiles-main\s-59(se)>
```

Conclusion:

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

In Java, you can create a class template, also known as a class blueprint, using the class keyword. A class template serves as a model for objects that you create based on its structure. Below is an example of how you can create a simple class template in Java:

```
public class MyClassTemplate {
   // Fields or data members
   private int myField;

   // Constructor
   public MyClassTemplate(int myField) {
      this.myField = myField;
   }

   // Methods
   public void setMyField(int myField) {
      this.myField = myField;
```

```
    }

    public int getMyField() {
        return myField;
    }
}
```

Here's a brief explanation of the components used in the above example:

public class MyClassTemplate: This line declares a class named MyClassTemplate.

private int myField;: This line declares a private integer variable myField, which is a field of the class.

public MyClassTemplate(int myField): This is the constructor, which initializes the myField variable.

public void setMyField(int myField): This is a setter method that allows you to modify the value of the myField variable.

public int getMyField(): This is a getter method that returns the value of the myField variable