LAB_ANP_C6339_CLASSES

Student ID: AF0339439

YELLA UDAY KUMAR

Assignment 1:

Create a simple Factory Pattern for creating shapes (e.g., Circle, Square, Triangle). Define an interface called Shape with a draw method, and create concrete classes Circle, Square, and Triangle that implement the Shape interface. Implement a ShapeFactory that has a method createShape which takes a string (e.g., "circle", "square", "triangle") as input and returns the corresponding shape object. Write a program to demonstrate the usage of the factory to create different shapes and call their draw methods.

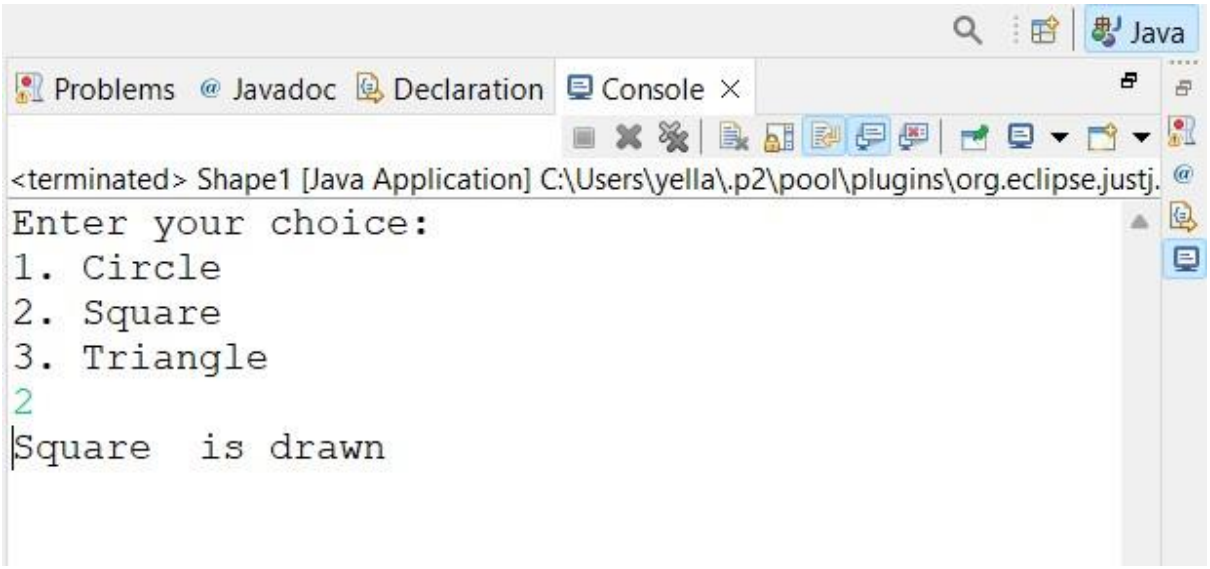
Program:1

```java
package Designpatterns;
import java.util.Scanner;

    // Shape interface
    interface Shape {
        void draw();
    }

    // Concrete class: Circle
    class Circle implements Shape {
        @Override
        public void draw() {
            System.out.println("Circle  is drawn");
        }
    }

    // Concrete class: Square
    class Square implements Shape {
        @Override
        public void draw() {
            System.out.println("Square  is drawn");
        }
    }
```

```java
    // Concrete class: Triangle
    class Triangle implements Shape {
        @Override
        public void draw() {
            System.out.println("Triangle  is drawn");
        }
    }

    // ShapeFactory
    class ShapeFactory {
        // Method to create a shape based on user input
        public Shape createShape(int choice) {
            switch (choice) {
                case 1:
                    return new Circle();
                case 2:
                    return new Square();
                case 3:
                    return new Triangle();
                default:
                    return null; // Return null for unknown
choices
            }
        }
    }

    // Program to demonstrate the usage of the factory
pattern with user input
    public class Shape1 {
        public static void main(String[] args) {
            Scanner scanner = new Scanner(System.in);
            ShapeFactory shapeFactory = new ShapeFactory();

            // Prompt user to enter a choice
            System.out.println("Enter your choice:");
            System.out.println("1. Circle");
            System.out.println("2. Square");
            System.out.println("3. Triangle");

            // Read user choice
            int userChoice = scanner.nextInt();

            // Create shape based on user's choice
            Shape chosenShape =
shapeFactory.createShape(userChoice);

            if (chosenShape != null) {
                // Call draw method of the chosen shape
                chosenShape.draw();
            } else {
                System.out.println("Invalid choice");
```

```
            }

            scanner.close();
        }
    }
```

Output:



Assignment 2 :

Create a simple Singleton Pattern for a logging class. Implement a Logger class that logs messages. Ensure that only one instance of the Logger class can be created, and all log messages are written to a single log file. Write a program to demonstrate the usage of the Logger class to log messages from multiple parts of the application.

Program:

```
package Designpatterns;

public class Logger {

public  static Logger log = new Logger();

     private Logger()
     {
          System.out.println("Logger instance is created.");
     }

     public static Logger createobject()
     {
          return log;
     }

     public void loggerInmsg()
     {
          System.out.println("call login registered");
     }

     public void loggerOutmsg()
     {
          System.out.println("call logout registered");
     }
```

****

```java
package Designpatterns;

public class Singleton_Logger {

	public static void main(String[] args) {

		//user1
		Logger log = Logger.createobject();

		log.loggerInmsg();
		log.loggerOutmsg();

		 System.out.println("");
		//user2
		Logger log1 = Logger.createobject();
		log1.loggerInmsg();
		log1.loggerOutmsg();


	}

}
```

Output: