****

**Green University of Bangladesh**

**Department of Computer Science and Engineering (CSE)**

**Faculty of Sciences and Engineering**

**Semester: (Fall, Year: 2024), B.Sc. in CSE (Day)**

**Lab Report NO #05**

**Course Title: OOP Lab**

**Course Code: CSE 202**

**Section: D9**

**Lab Experiment Name:**  Polymorphism

**Student Details**

|  |  |  |
| --- | --- | --- |
| **Name** | | **ID** |
| **1.** | Rokonuzzaman Topu | 232002280 |

**Lab Date : 18/09/24**

**Submission Date : 24/10/24**

**Course Teacher’s Name : Wahia Tasnim**

**[For Teachers use only: Don’t write anything inside this box]**

|  |
| --- |
| **Lab Report Status**  **Marks: ………………………………… Signature: .....................**  **Comments: .............................................. Date: ..............................** |

**1. INTRODUCTION**

This task explores the concept of **interfaces** and **polymorphism** in Java. Interfaces in Java define a contract that classes can implement, ensuring specific methods are present.

**2. OBJECTIVES**

The primary objectives of this lab report are as follows:

• To gather knowledge of how interface works.

• Understand multiple inheritance using a class implementing multiple interfaces.

• To solve problems using the concept of interface.

**3. IMPLEMENTATION**

Task 1:

 Define an interface called IsEmergency that contains a single method, soundSiren, which has no arguments and does not return a value.

 Create a class named FireEmergency that implements the IsEmergency interface. The soundSiren method in this class should display "Siren Sounded" when called.

 Develop another class called SmokeAlarm, which does not implement any interface and has an empty body.

 In the main method:

* Create an array named myArray that stores objects of the Object class.
* Instantiate two SmokeAlarm objects and add them to myArray.
* Instantiate two FireEmergency objects and add them to myArray.
* Iterate through the array using a loop to identify objects that are instances of classes implementing the IsEmergency interface. For such objects, invoke their soundSiren method. If an object does not implement the interface, indicate this in the output.

Solution:

interface isEmergency {

    void soundSiren();

}

class FireEmergency implements isEmergency {

    @Override

    public void soundSiren() {

        System.out.println("Siren Sounded");

    }

}

class SmokeAlarm {

}

public class EmergencyAlarm {

    public static void main(String[] args) {

        Object[] myArray = new Object[4];

        myArray[0] = new SmokeAlarm();

        myArray[1] = new SmokeAlarm();

        myArray[2] = new FireEmergency();

        myArray[3] = new FireEmergency();

        for (Object obj : myArray) {

            if (obj instanceof isEmergency) {

                ((isEmergency) obj).soundSiren();

            } else {

                System.out.println("Object is not an instance of IsEmergency.");

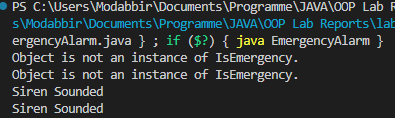
            }

        }

    }

}

Output:



**4. DISCUSSION**

This program illustrates the application of interfaces and type verification in Java. The IsEmergency interface mandates the implementation of the soundSiren method in the FireEmergency class. By leveraging polymorphism, the program stores a mix of objects in an Object[] array and uses the instanceof operator to determine their types at runtime. This approach ensures that only objects implementing the interface are acted upon appropriately. The program demonstrates how interfaces and runtime type checking contribute to building adaptable and modular software solutions..