Experiment No: 5

AIM: To demonstrate the use of abstract classes and interfaces.

Date:

CO mapped: CO-2

Objectives:

a) To understand the purpose and usage of abstract classes and interfaces in object-oriented programming. Develop the ability to design and implement abstract classes and interfaces effectively to promote code reusability, ensure consistent behavior in class hierarchies, and facilitate the development of flexible and extensible software systems.
b) Abstract classes and interfaces are important OOP concepts that allow you to define common contracts and behaviors for classes. Achieving this objective will enable you to use these tools to create more modular and maintainable software, especially when dealing with class hierarchies and multiple implementations.

Background:

A class that is declared as abstract is known as an abstract class. It can have abstract and non-abstract methods. It needs to be extended and its method implemented. It cannot be instantiated.
Points to Remember

- An abstract class must be declared with an abstract keyword.
- It can have abstract and non-abstract methods.
- It cannot be instantiated.
- It can have constructors and static methods also.
- It can have final methods which will force the subclass not to change the body of the method.

An interface in Java is a blueprint of a class. It has static constants and abstract methods. The interface in Java is a mechanism to achieve abstraction. There can be only abstract methods in the Java interface, not the method body. It is used to achieve abstraction and multiple inheritance in Java. In other words, you can say that interfaces can have abstract methods and variables. It cannot have a method body.

Practical questions:

1. Describe abstract class called Shape which has three subclasses say Triangle, Rectangle, Circle. Define one method area() in the abstract class and override this area() in these three subclasses to calculate for specific object, i.e., area() of Triangle subclass should calculate area of triangle etc. Same for Rectangle and Circle.
2. Write a program that demonstrates the instance of operator. Declare interfaces I1 and I2. Interface I3 extends both of these interfaces. Also declare interface I4. Class X implements I3. Class W extends X and implements I4. Create an object of class W. Use the instance of operator to test if that object implements each of the interfaces and is of type X.

3. Write a java program to implement an interface called Exam with a method Pass (int mark) that returns a boolean. Write another interface called Classify with a method Division (int average) which returns a String. Write a class called Result which implements both Exam and Classify. The Pass method should return true if the mark is greater than or equal to 50 else false. The Division method must return "First" when the parameter average is 60 or more, "Second" when average is 50 or more but below 60, "No division" when average is less than 50.

Observations: Put Output of the program

Conclusion: (Sufficient space to be provided)

Quiz: (Sufficient space to be provided for the answers)

1. Explain how interfaces promote the concept of multiple inheritances in OOP.
2. What is an interface, and how does it differ from an abstract class?
3. When would you choose to use an abstract class over an interface, and vice versa, in your software design?
4. Can a class implement multiple interfaces? If so, what benefits does this provide?
5. Can you declare an interface method static? Justify your answer.

Suggested Reference:

1. https://www.tutorialspoint.com/java/
2. https://www.geeksforgeeks.org/
3. https://www.w3schools.com/java/
4. https://www.javatpoint.com/