Subject: SBL-OOPM **Class: SE-DS**

Semester: III **A.Y. 2022-2023**

Experiment No. 8

- **Aim :** To write a program on abstract class and abstract methods.
- **Objectives :** To will learn about Java abstract classes and methods with the help of examples. We will also learn about abstraction in Java.
- **Prerequisites :** Students should know disadvantages of Procedure oriented programming language & the need of OOPs concepts to overcome those disadvantages.
- **Software used :** jdk 1.6.0
- **Theory :**

A class which is declared with the abstract keyword is known as an abstract class in Java . It can have abstract and non-abstract methods (method with the body).
Before learning the Java abstract class, let's understand the abstraction in Java first.

Abstraction in Java

Abstraction is a process of hiding the implementation details and showing only functionality to the user.
Another way, it shows only essential things to the user and hides the internal details, for example, sending SMS where you type the text and send the message. You don't know the internal processing about the message delivery.

Abstraction lets you focus on what the object
does instead of how it does it.

Ways to achieve Abstraction

There are two ways to achieve abstraction in java

1. Abstract class (0 to 100%)
2. Interface (100%)

Abstract class in Java

A class which 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.

Example of abstract class

```
abstract class A{}
```

Abstract Method in Java

A method which is declared as abstract and does not have implementation is known as an abstract method.

Example of abstract method

In this example, if you create the instance of Rectangle class, draw() method of Rectangle class will be invoked.

File: TestAbstraction1.java

```
abstract class Shape{
abstract void draw();
}
//In real scenario, implementation is provided by others i.e. unknown by end user
class Rectangle extends Shape{
void draw(){System.out.println("drawing rectangle");}
}
class Circle1 extends Shape{
void draw(){System.out.println("drawing circle");}
}
//In real scenario, method is called by programmer or user
class TestAbstraction1{
public static void main(String args[]){
Shape s=new Circle1();//In a real scenario, object is provided through method, e.g., getShape() m
ethod
```

```
s.draw();
}
}
```

Output:

```
drawing circle
```

Abstract class having constructor, data member and methods

An abstract class can have a data member, abstract method, method body (non-abstract method), constructor, and even main() method.

File: TestAbstraction2.java

```
//Example of an abstract class that has abstract and non-abstract methods
 abstract class Bike{
   Bike(){System.out.println("bike is created");}
   abstract void run();
   void changeGear(){System.out.println("gear changed");}
 }
//Creating a Child class which inherits Abstract class
 class Honda extends Bike{
 void run(){System.out.println("running safely..");}
 }
//Creating a Test class which calls abstract and non-abstract methods
 class TestAbstraction2{
 public static void main(String args[]){
  Bike obj = new Honda();
  obj.run();
  obj.changeGear();
 }
}
```

Output:

```
bike is created
running safely..
gear changed
```

```
class Bike12{
abstract void run();
}
```

Output:

```
compile time error
```

Rule: If you are extending an abstract class that has an abstract method, you must either provide the implementation of the method or make this class abstract.

CONCLUSION : Summaries what you understood from this lab.