SNS COLLEGE OF ENGINEERING

Kurumbapalayam(Po), Coimbatore – 641 107
Accredited by NAAC-UGC with ‘A’ Grade
Approved by AICTE, Recognized by UGC & Affiliated to Anna University, Chennai

Department of Information Technology

19CS204 OBJECT ORIENTED PROGRAMMING

I YEAR /II SEMESTER

Topic – Abstract Class

Abstract Classes

- 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.

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

- Abstraction can be achieved with either **abstract classes** or **interfaces.**

Abstract Classes

- The abstract keyword is a non-access modifier, used for classes and methods:

Abstract class:

- It is a restricted class that cannot be used to create objects (to access it, it must be inherited from another class).

Abstract method:

- can only be used in an abstract class, and it does not have a body. The body is provided by the subclass (inherited from).

Abstract Classes

Abstract Class

- A class which contains the **abstract** keyword in its declaration is known as abstract class.

- The abstract class in Java cannot be instantiated (we cannot create objects of abstract classes)

- To use an abstract class, you have to inherit it from another class, provide implementations to the abstract methods in it.

Abstract Classes

Rules for Abstract Class

- 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.

Abstract Classes

Abstract Method in Java

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

Rules for Abstract Methods

- abstract keyword is used to declare the method as abstract.
- You have to place the abstract keyword before the method name in the method declaration.
- An abstract method contains a method signature, but no method body. (Implementation)
- If you inherit an abstract class, you have to provide implementations to all the abstract methods in it.

Abstract Classes

```
abstract class A {
abstract void callme();
// concrete methods are still allowed in abstract classes
void callmetoo() {
System.out.println("This is a concrete method.");
}
}
class B extends A {
void callme() {
System.out.println("B's implementation of callme.");
}
}
class AbstractDemo {
public static void main(String args[]) {
B b = new B();
b.callme();
b.callmetoo();
}
}
```

Abstract Classes

```
// Using abstract methods and classes.
abstract class Figure {
double dim1;
double dim2;
Figure(double a, double b) {
dim1 = a;
dim2 = b;
}
// area is now an abstract method
abstract double area();
}

class Rectangle extends Figure {
Rectangle(double a, double b) {
super(a, b);
}
// override area for rectangle
double area() {
System.out.println("Inside Area for Rectangle.");
return dim1 * dim2;
}
}
```

```
class Triangle extends Figure {
Triangle(double a, double b) {
super(a, b);
}
// override area for right triangle
double area() {
System.out.println("Inside Area for Triangle.");
return dim1 * dim2 / 2;
}
}
```

Abstract Classes

```
class AbstractAreas {
public static void main(String args[]) {
// Figure f = new Figure(10, 10); // illegal now
Rectangle r = new Rectangle(9, 5);
Triangle t = new Triangle(10, 8);
r.area();
t.area();
System.out.println("Area is " + r.area());
System.out.println("Area is " + t.area());
}
}
```

THANK YOU