Kalyani Government Engineering College

Department of Computer Application



# **CA2 ASSIGNMENT**

|  |  |
| --- | --- |
| Name | Rajendra Kumar Shaw |
| Roll | 10271022021 |
| Stream | MCA 2nd Sem |
| Subject Name | OOPS with JAVA |
| Subject Code | MCAN-203 |

Table of Content Page

1. Finalizer Methods………………………………………………………….3

* Introduction
* Syntax
* Example

1. Abstract Class……………………………………………………………..4

* Introduction
* Rules of abstract class

1. Abstract Method………………………………………………………...4-5
2. Example of Abstract Class with Abstract Method………………………..5

**Finalizer Methods**

**Introduction:**

* In java, we have a default garbage collection system, which automatically frees up the memory resources used by Objects. But in some cases the objects may hold other non-object resources such as file descriptors or windows system fonts etc. The java garbage collection system cannot free these resources. So, in order to free these resources we have to use finalize method.
* This method can be added in any class , we don’t have to define anything in this method. Java calls this method whenever it is about to reclaim the space for that object.
* It is also possible that an object never gets garbage collected and thus the finalize method is never called.

**Syntax:**

* finalize(){}

**E.g.**

class MyClass{

display(){}

}

public class sample{

public static void main(String[] args){  
 MyClass a = new MyClass();

a.display();

}

protected void finalize(){

System.out.println(“Finalize method is called.”); //we can remove this line

} //as it is not necessary

}

* Finalize methods should be declared at the end of the program.
* It can be simply finalize(){} without having any body.

**Abstract Class**

**Introduction:**

* In some situations we need to define a superclass that declares the structure of a given abstractions without providing a complete implementation of every method. That means only the declaration will be done in the parent class that will be shared by all of its subclasses, leaving it to each subclass to define and fill the details in it. The object of abstract class cannot be declared, meanwhile their members are accessed by the objects of their subclass.
* Syntax: abstract class classname{}

**Rules of using an Abstract class:**

* An abstract class must be declared with an abstract keyword.
* From an abstract class, object cannot be created.
* The abstract method of an abstract class must be defined in the subclass.
* We cannot declare abstract-constuctor or abstract-static methods.
* 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.
* It can have abstract and non-abstract methods.

**Abstract Method**

* Sometimes, we may require certain methods to be overridden by subclasses. The subclassers must override them, because they have no implementation specified in the superclass. That means the subclass cannot simply use the version defined in the superclass, these kind of methods is said to be Abstract Methods.
* The subclass must override them.
* It can only be declared inside an abstract class.
* Syntax: abstract type name(parameter);

**E.g of Abstract class with abstract method**

abstract class Abs{

int i;

Abs(int t){

i = t;

}

Abstract void display();

}

class Sub extends Abs{

int j;

Sub(int p1, int p2){

Super(p1);

j = p2;

}

void display(){

System.out.println(“Abstract value: “+i);

System.out.println(“Another value: “+j);

}

}

public class Sample{

public static void main(String[] args){

Sub s = new Sub(10,20);

s.display();

}

}