

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

<b>Lab Number:</b>	7
<b>Student Name:</b>	Siddhant Kedar
<b>Roll No :</b>	E-05

**Title: Write a java program to implement Multiple Inheritance using Interfaces. Create an interface called Management with selectCandidate() method. Another interface called Department with allotSubject() method. Class called HOD will implements these two interfaces and define the methods and access them with valid objects.**

**Learning Objective:**

- Students will be able to implement multiple inheritance using Interface concept.

**Learning Outcome:**

- Understanding the abstraction concept and hiding of unnecessary code using interface.

**Theory:**

Java does not support multiple inheritance. This means that a class cannot extend more than one class.

**Multiple Inheritance :** Multiple Inheritance is a feature of an object-oriented concept, where a class can inherit properties of more than one parent class. The problem occurs when there exist methods with the same signature in both the superclasses and subclass. On calling the method, the compiler cannot determine which class method to be called and even on calling which class method gets the priority.

Multiple inheritance in Java programming is achieved or implemented using interfaces. Java does not support multiple inheritance using classes.

In simple term, a class can inherit only one class and multiple interfaces in a java programs.

**Reasons to use Interface**

There are mainly three reasons to use interface. They are given below.

- It is used to achieve abstraction.
- By interface, we can support the functionality of multiple inheritance.
- It can be used to achieve loose coupling.

**Faculty: Ms. Deepali Kayande**

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

- **How to declare an interface?**
- An interface is declared by using the interface keyword. It provides total abstraction; means all the methods in an interface are declared with the empty body, and all the fields are public, static and final by default. A class that implements an interface must implement all the methods declared in the interface.

**Algorithm:**

- 1) Start
- 2) Create interface - Management and declare select candidate() in it.
- 3) Create interface - Department and declare allotsubject() in it
- 4) Create a class HOD to inherit interface management and department and to take input of details.
- 5) Create the object of the HOD class in main function and call the methods.
- 6) Print results

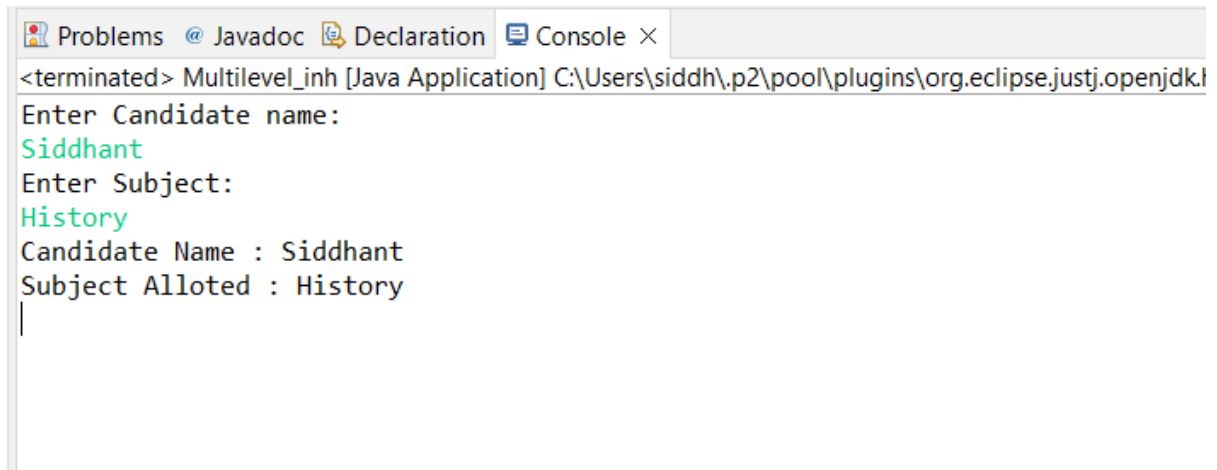
```
import java.util.*;
interface Management
{
    void selectCandidate();
}
interface Department
{
    void allotSubject();
}
class HOD implements Department, Management
{
    String Candidate;
    String Subject;
    void getdata()
    {
        Scanner in=new Scanner(System.in);

        System.out.println("Enter Candidate name:");
        Candidate=in.nextLine();
        System.out.println("Enter Subject:");
        Subject=in.nextLine();
    }
    public void selectCandidate()
    {
        System.out.println("Candidate Name : "+Candidate );
    }
    public void allotSubject()
```

**Don Bosco Institute of Technology, Kurla(W)**  
**Department of Electronics and Tele-Communication Engineering**  
**ECL304 - Skill Lab: C++ and Java Programming**  
**Sem III**  
**2021-22**

```
{  
System.out.println("Subject Alloted : "+Subject);  
}  
}  
public class Multilevel_inh  
{ public static void main (String[] args)  
{  
  
HOD ob = new HOD(); ob.getdata();  
ob.selectCandidate(); ob.allotSubject();  
}  
}
```

**Output :**



The screenshot shows the Eclipse IDE's Console window. The title bar includes 'Problems', '@ Javadoc', 'Declaration', and 'Console'. The console text is as follows:

```
<terminated> Multilevel_inh [Java Application] C:\Users\siddh\p2\pool\plugins\org.eclipse.justj.openjdk.l  
Enter Candidate name:  
Siddhant  
Enter Subject:  
History  
Candidate Name : Siddhant  
Subject Alloted : History  
|
```