**Java lab sheet-06**

1).1.

// MyFirstInterface interface

interface MyFirstInterface {

// Integer type variable

int x = 10;

// Abstract method

void display();

}

Declare the variable without public static final keywords:

Int x=10;

Declare the variable with public static final keywords:

public static final int x = 10;

In both cases, the variable "x" will behave as a constant, meaning its value cannot be changed once it's set. There is no difference between these two approaches in terms of functionality. Using public static final simply redundant when declaring variables in an interface because they are implicitly assumed to have those modifiers. So, the above two variable declarations are equivalent in an interface.

2.

interface MyFirstInterface {

void display(); // Equivalent to: abstract void display();

}

When declaring a method in an interface, it is always implicitly assumed to be abstract, whether or not you explicitly use the abstract keyword. Therefore, there is no difference between declaring the abstract method with or without the abstract keyword in an interface.

3.

class InterfaceImplemented implements MyFirstInterface {

@Override

public void display() {

System.out.println("Value of x: " + x);

}

}

**2).**

// Speaker interface

interface Speaker {

void speak();

}

class Politician implements Speaker {

@Override

public void speak() {

System.out.println("Politician speaks about policies and governance.");

}

}

// Priest class implementing the Speaker interface

class Priest implements Speaker {

@Override

public void speak() {

System.out.println("Priest speaks about religious teachings and spirituality.");

}

}

// Lecture class implementing the Speaker interface

class Lecture implements Speaker {

@Override

public void speak() {

System.out.println("Lecture covers a specific educational topic.");

}

}

public class TestSpeaker {

public static void main(String[] args) {

Speaker politician = new Politician();

Speaker priest = new Priest();

Speaker lecture = new Lecture();

System.out.println("Politician:");

politician.speak();

System.out.println("\nPriest:");

priest.speak();

System.out.println("\nLecture:");

lecture.speak();

}

}