

## Installation of Java and Simple Java Programs

### Aim:

To install Java Development Kit (JDK), configure the environment, and write simple Java programs including Hello World.

### PRE LAB EXERCISE

#### QUESTIONS

1. What is JDK and why is it required?

JDK (Java Development Kit) is a software package used to develop Java applications. It includes tools required to write, compile, debug, and run Java programs.

Why it is required:

- To compile Java source code into bytecode
- To run Java programs
- To develop Java applications using tools like:
  - javac (compiler)
  - java (interpreter)
  - debugger and other utilities

2. Difference between JDK, JRE, and JVM.

Feature	JDK	JRE	JVM
Full form	Java Development Kit	Java Runtime Environment	Java Virtual Machine
Purpose	Develop and run Java programs	Run Java programs	Executes Java bytecode
Contains	JRE + development tools	JVM + libraries	Only execution
Used by	Programmers	Users	engine System

### 3. What is the purpose of the main() method in Java?

The main() method is the starting point of execution of any Java program.

Purpose:

- It tells the JVM where the program starts
- Without main() method, the program will not execute Syntax:

```
public static void main(String[] args)
```

## IN LAB EXERCISE

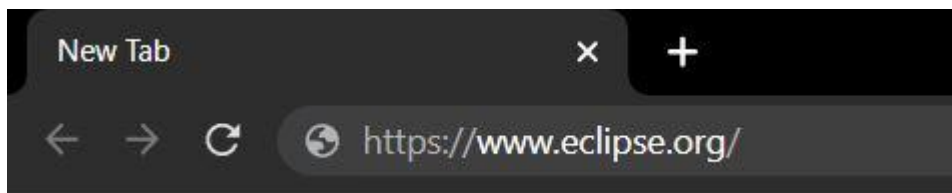
### Objective:

To verify Java installation and execute a basic Java program.

### INSTALLATION STEPS:

#### STEP 1: Open Browser

- Open your browser and go to the official [URL](https://www.eclipse.org/) Eclipse Downloads page.



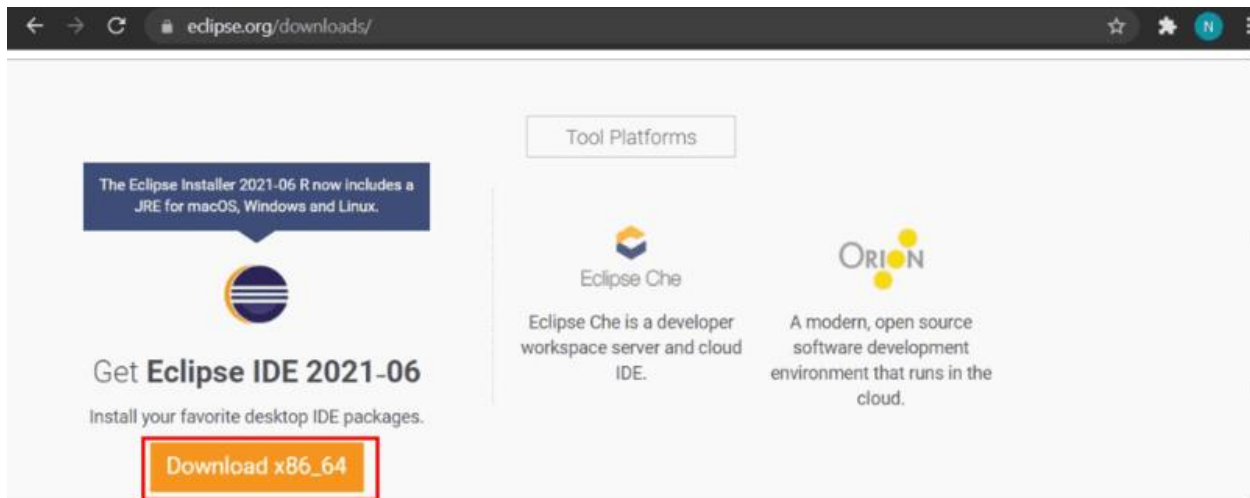
#### STEP 2: Download Eclipse Installer

- Then, click on the "Download" button to download Eclipse IDE.

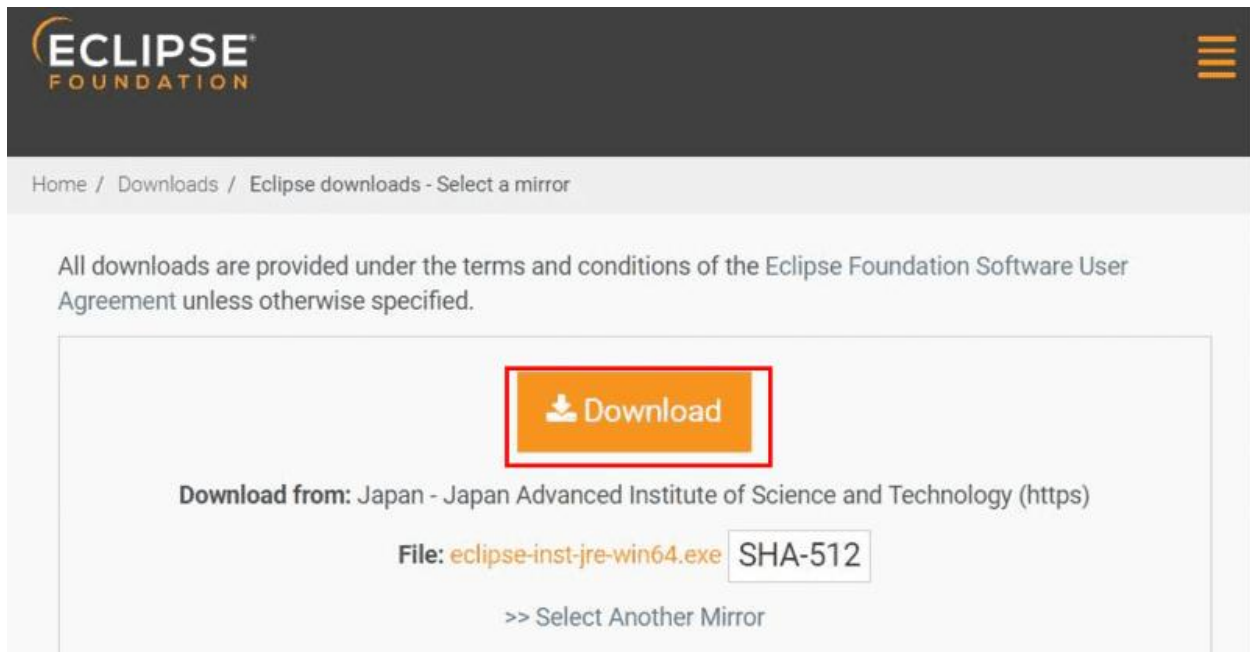


### STEP 3: Download EXE

- Now, click on the "Download x86\_64" button.

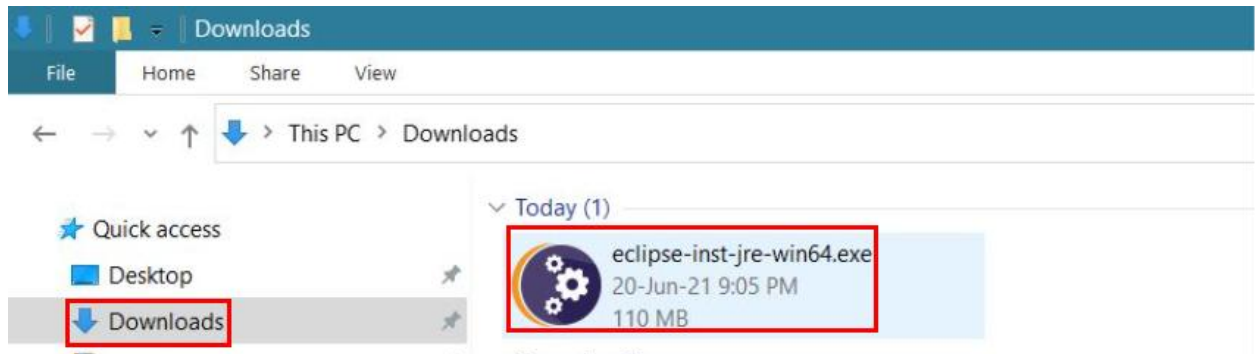


**STEP 4:** Then click on the "Download" button. After clicking on the download button the .exe file for the eclipse will be downloaded.



## STEP 5: Open Download EXE

- Now go to File Explorer and click on "Downloads" after that click on the "*eclipse-inst-jre-win64.exe*" file for installing Eclipse IDE.

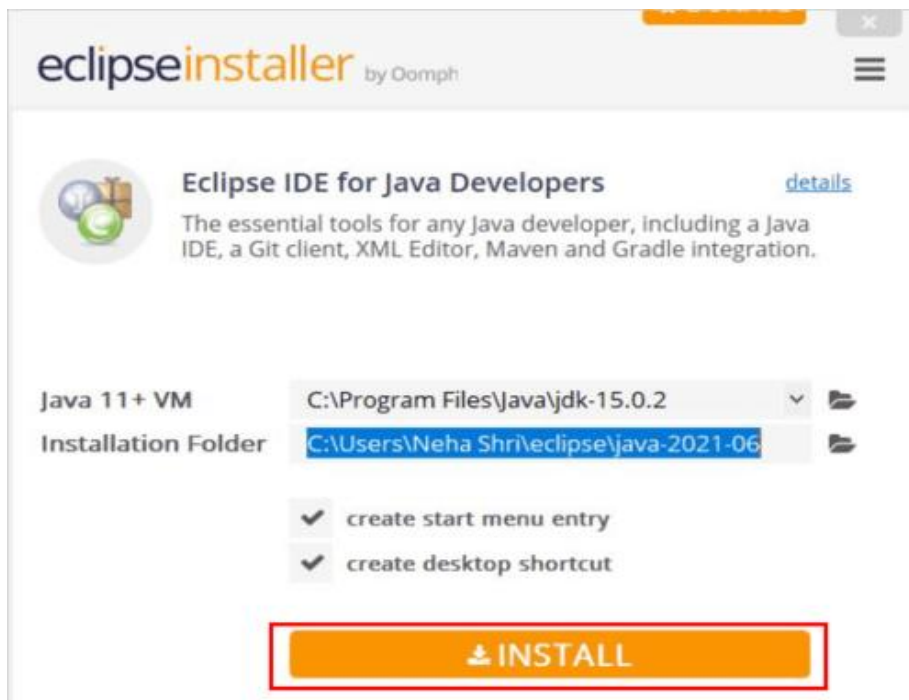


## STEP 6: Install Eclipse

- Then, click on "Eclipse IDE for Java Developers".

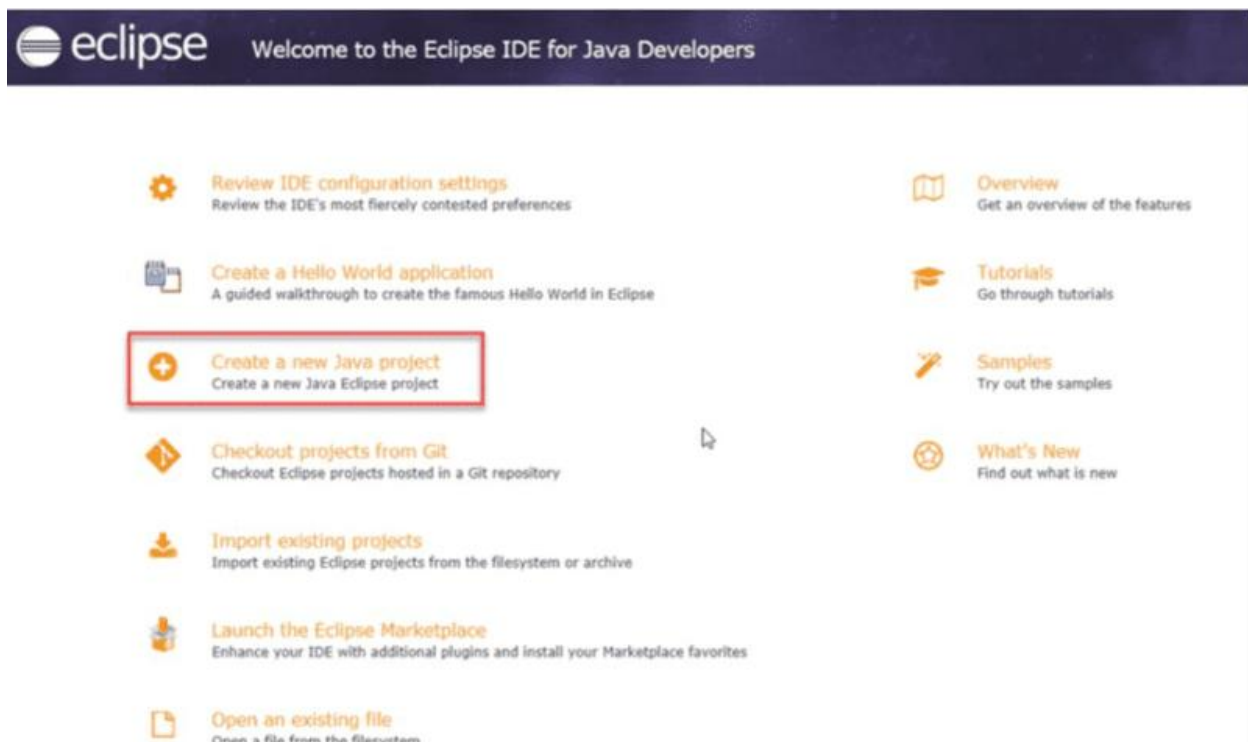


**STEP 7: Then, click on the "Install" button.**




## Step 8: Create New Project

Now click on "Create a new Java project".

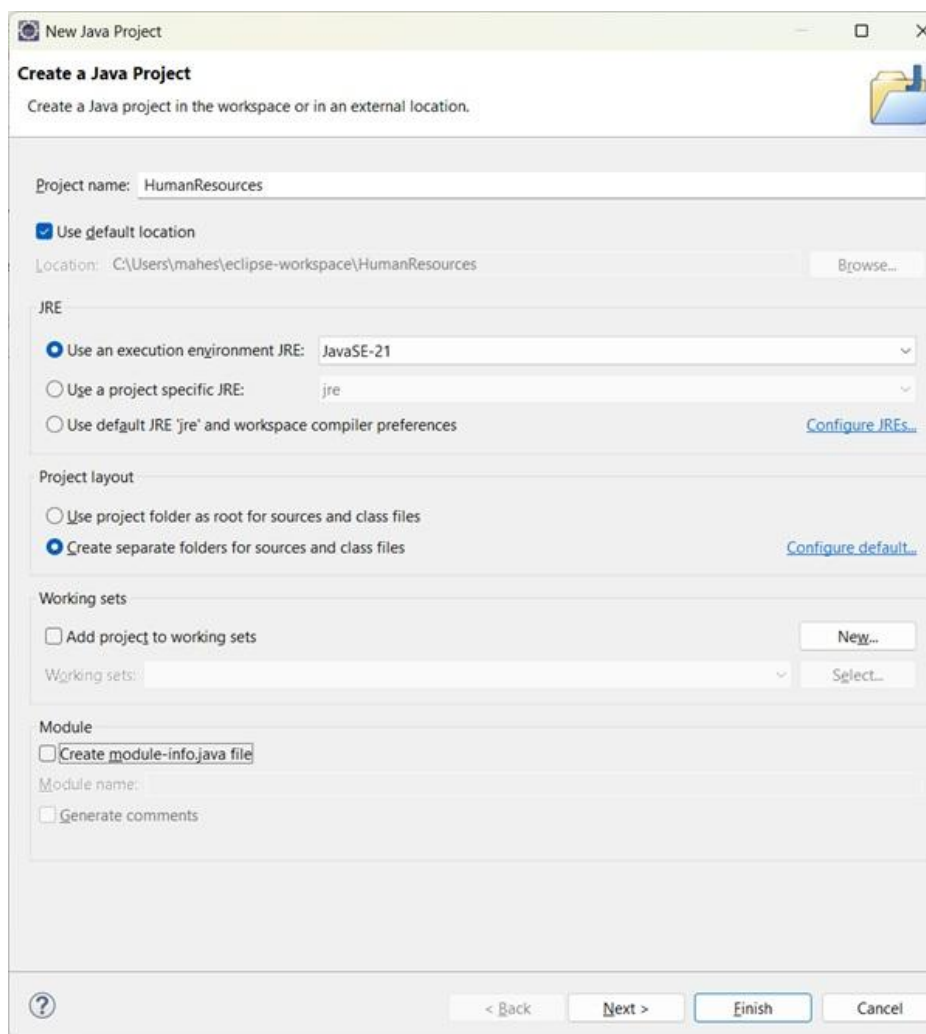


## STEP 9: Create a new java project

- By clicking on the File menu and choosing New → Java Project.
- By right clicking anywhere in the Project Explorer and selecting New → Java Project.
- By clicking on the New button (  ) in the Tool bar and selecting Java Project.


## STEP 10: Enter the Project Name

- Select the Java Runtime Environment (JRE) or leave it at the default
- Select the Project Layout which determines whether there would be a separate folder for the source codes and class files. The recommended option is to create separate folders for sources and class files.



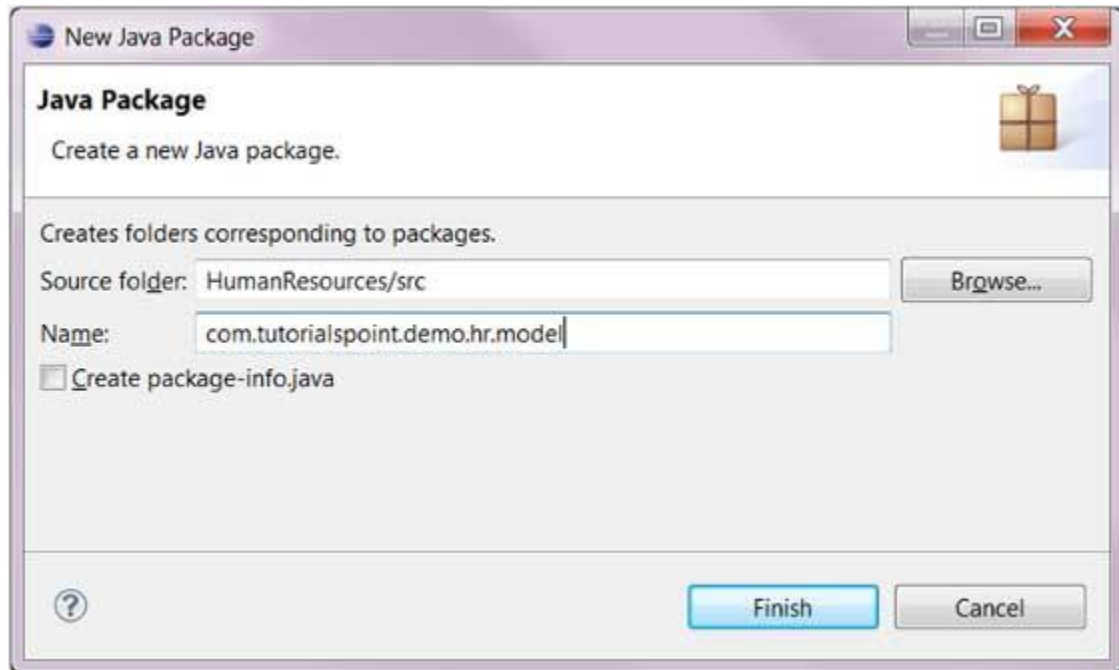
## STEP 11: Create a new java package

- By clicking on the File menu and selecting New → Package.
- By right click in the package explorer and selecting New → Package.



- By clicking on the package icon which is in the tool bar(  ).

#### STEP 11:

- Enter/confirm the source folder name.
- Enter the package name.
- Click on the Finish button.

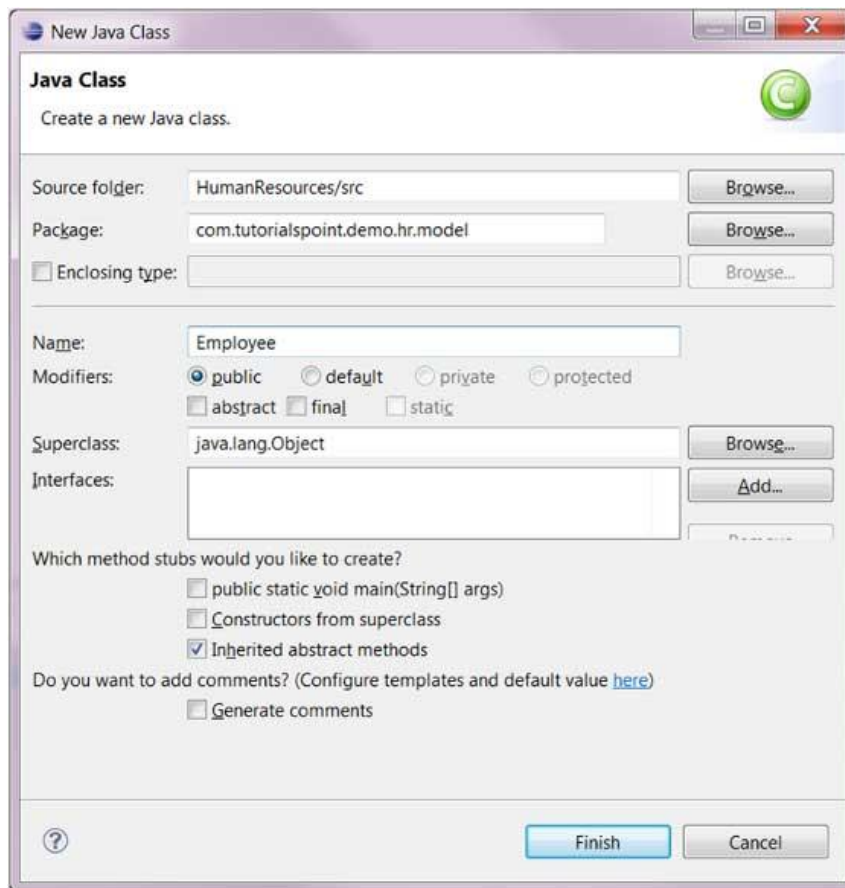


#### STEP 12: Create a New Java class.

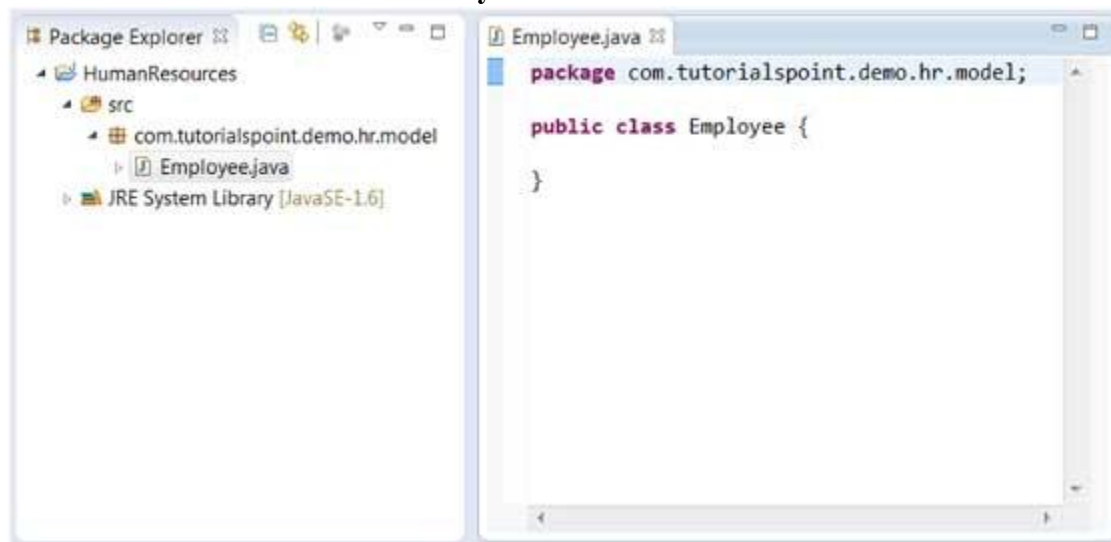
- By clicking on the File menu and selecting New → Class.
- By right clicking in the package explorer and selecting New → Class.
- By clicking on the class drop down button (  ) and selecting class (  ).

#### STEP 13:

- Ensure the source folder and package are correct.
- Enter the class name.
- Select the appropriate class modifier.
- Enter the super class name or click on the Browse button to search for an existing class.
- Click on the Add button to select the interfaces implemented by this class.
- Examine and modify the check boxes related to method stubs and comments.



**STEP 14: Class created successfully.**





## **BASIC PROGRAMS:**

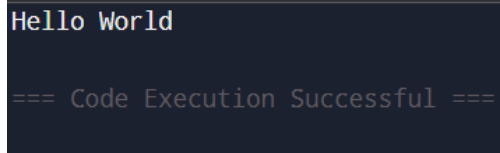
### **Program 1: Hello World Program**

#### **Source Code:**

```
class HelloWorld {  
    public static void main(String[] args) {  
        System.out.println("Hello World");  
    }  
}
```

#### **Output:**

Hello World



```
Hello World  
  
=== Code Execution Successful ===
```

### **Program 2: Display Personal Details**

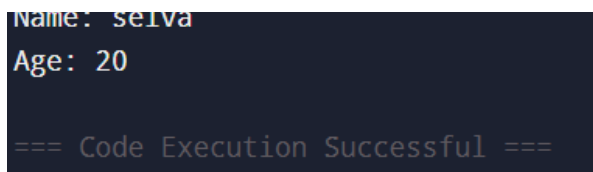
#### **Source Code:**

```
class DisplayInfo {  
    public static void main(String[] args) {  
        System.out.println("Name: selva");  
        System.out.println("Age: 20");  
    }  
}
```

#### **Output:**

Name:selva

Age: 20

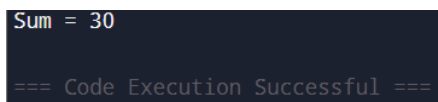


```
Name: selva  
Age: 20  
  
=== Code Execution Successful ===
```

### **Program 3: Addition of Two Numbers**

**Source Code:**

```
class AddTwoNumbers {  
    public static void main(String[] args) {  
        int a = 10, b = 20;  
        System.out.println("Sum = " + (a + b));  
    }  
}
```

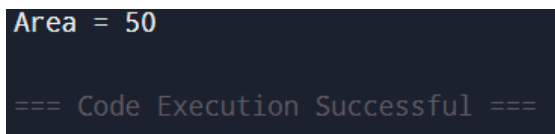
**Output:**

```
Sum = 30  
=== Code Execution Successful ===
```

Sum = 30

**Program 4: Area of a Rectangle****Source Code:**

```
class AreaRectangle {  
    public static void main(String[] args) {  
        int length = 10, breadth = 5;  
        System.out.println("Area = " + (length * breadth));  
    }  
}
```



```
Area = 50  
=== Code Execution Successful ===
```

**Output:**

Area = 50

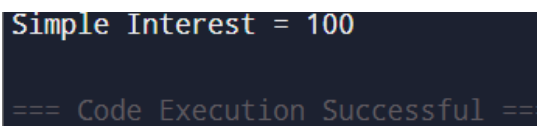
**Program 5: Simple Interest Calculation****Source Code:**

```
class SimpleInterest {  
    public static void main(String[] args) {  
        int p = 1000;
```

```
int r = 5;
int t = 2;
int si = (p * r * t) / 100;
System.out.println("Simple Interest = " + si);
}
}
```

### Output:

Simple Interest = 100



```
Simple Interest = 100
=== Code Execution Successful ===
```

## POST LAB EXERCISE

1. Write a Java program to display your name and department.

Program:

```
class DisplayDetails {
    public static void main(String[] args) {
        System.out.println("Name: selva");
        System.out.println("Department: Computer Science");
    }
}
```

Output:

Name: selva

Department: Computer Science

2. Modify the program to print the output in the same line.

Program:

```
class DisplayDetailsSameLine {    public
static void main(String[] args) {
    System.out.print("Name: selva ");
```

```
        System.out.print("Department: Computer Science");  
    }  
}
```

Output:

Name: selva Department: Computer Science 3.

What happens if main() is written without static?

Answer:

If the main() method is written without static, the program will not run.

Reason:

- The JVM calls main() without creating an object.
- A non-static method requires an object.
- Hence, JVM throws an error and execution fails.

4. Why is Java called platform independent?

Answer:

Java is called platform independent because Java programs can run on any operating system without modification.

Reason:

- Java source code is compiled into bytecode.
- Bytecode runs on JVM, which is available for different platforms (Windows, Linux, macOS).
- Hence, Java follows the principle: "Write Once, Run Anywhere."

5. Write a program to find the cube of a number.

Program:

```
class CubeNumber {  
    public static void main(String[] args) {  
int num = 5;  
        int cube = num * num * num;  
System.out.println("Cube = " + cube);  
    }  
}
```

Output:

Cube = 125

**Result:**

**Thus the Java IDE was successfully installed and a simple Java program was executed.**

#### **ASSESSMENT**

<b>Description</b>	<b>Max Marks</b>	<b>Marks Awarded</b>
Pre Lab Exercise	<b>5</b>	
In Lab Exercise	<b>10</b>	
Post Lab Exercise	<b>5</b>	
Viva	<b>10</b>	
<b>Total</b>	<b>30</b>	
<b>Faculty Signature</b>		