Experiment Number : 01 **Date: 22.01.2025**

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, compile, debug, and run Java applications. It is required to compile and execute Java code.

2. Difference between JDK, JRE, and JVM.
   - **JDK**: Develops and runs Java programs
   - **JRE**: Runs Java programs
   - **JVM**: Executes Java bytecode

3. What is the purpose of the main() method in Java?
   The main() method is the **starting point** of a Java program.

IN LAB EXERCISE

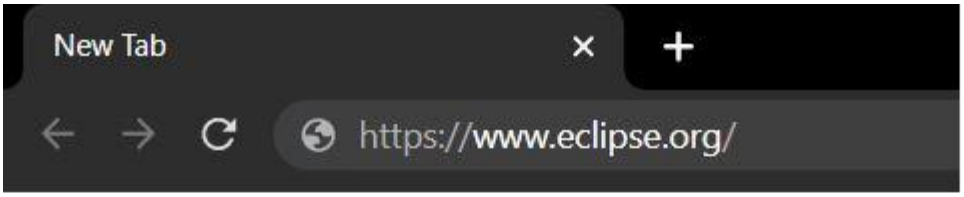
Objective:

To verify Java installation and execute a basic Java program.

INSTALLATION STEPS:

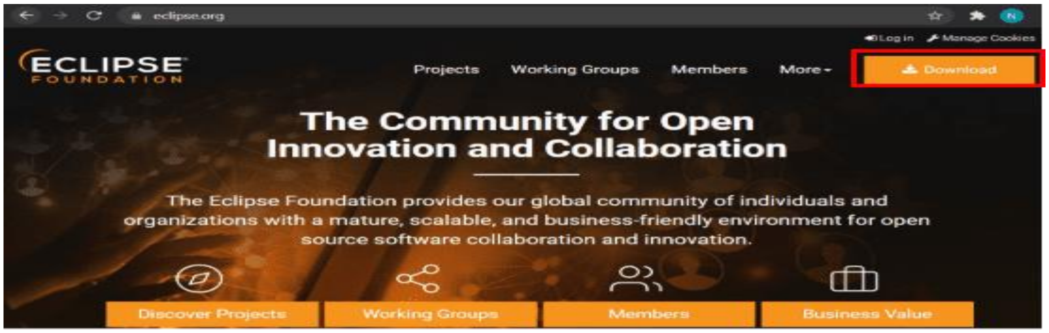
STEP 1: Open Browser

- Open your browser and go to the official URL Eclipse Downloads page.



STEP 2: Download Eclipse Installer

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

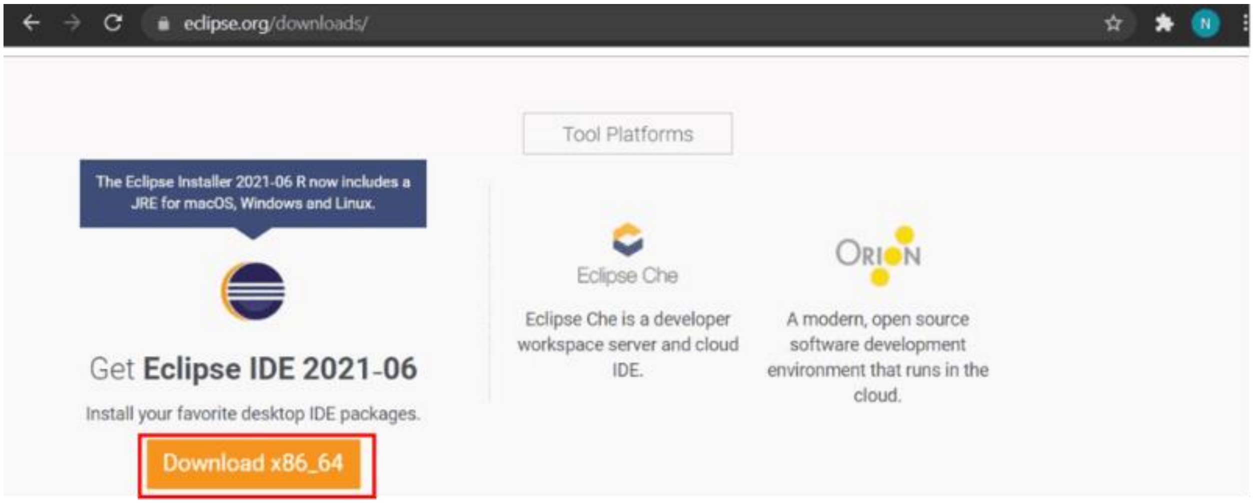


The Community for Open Innovation and Collaboration

The Eclipse Foundation provides our global community of individuals and organizations with a mature, scalable, and business-friendly environment for open source software collaboration and innovation.

STEP 3: Download EXE

- Now, click on the "Download x86_64" button.



The Eclipse Installer 2021-06 R now includes a JRE for macOS, Windows and Linux.

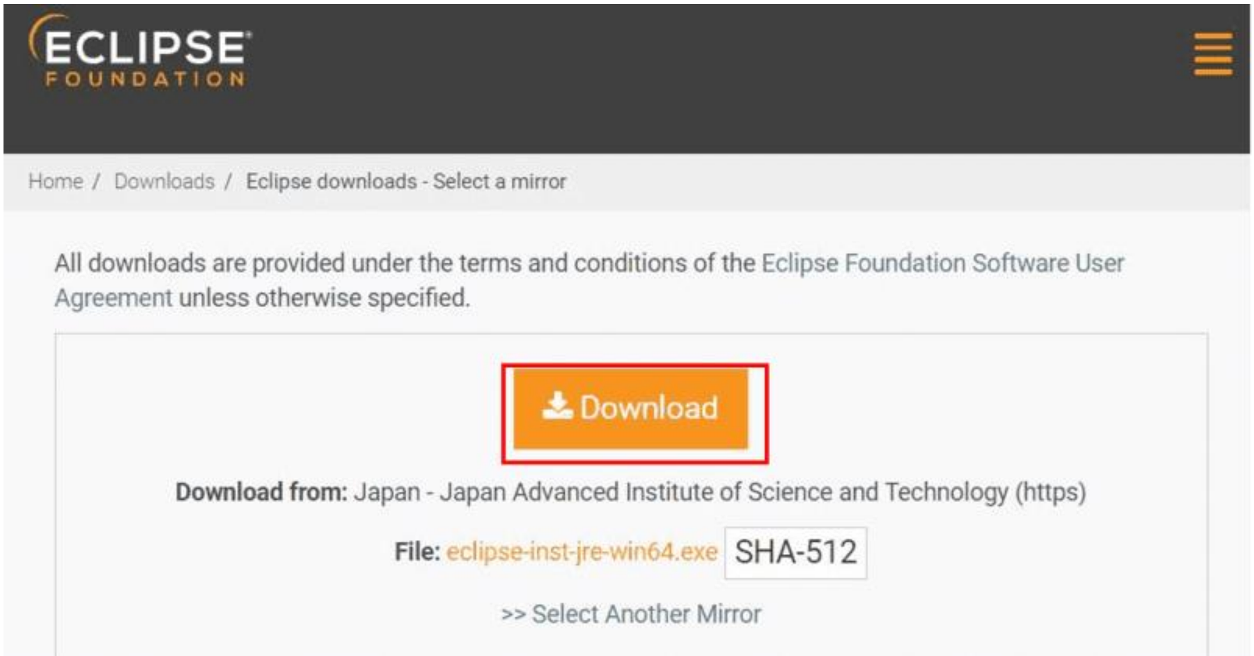
Get **Eclipse IDE 2021-06**

Install your favorite desktop IDE packages.

Eclipse Che is a developer workspace server and cloud IDE.

A modern, open source software development environment that runs in the cloud.

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



All downloads are provided under the terms and conditions of the Eclipse Foundation Software User Agreement unless otherwise specified.

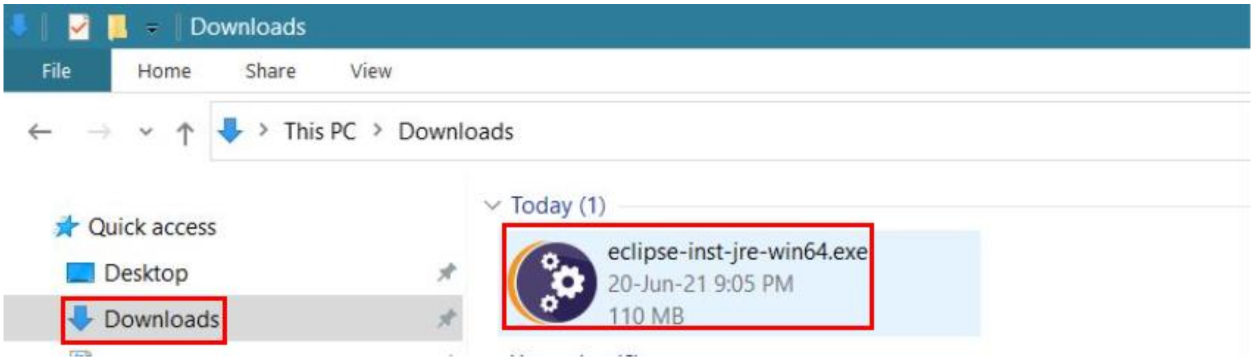
Download from: Japan - Japan Advanced Institute of Science and Technology (https)

File: eclipse-inst-jre-win64.exe SHA-512

>> Select Another Mirror

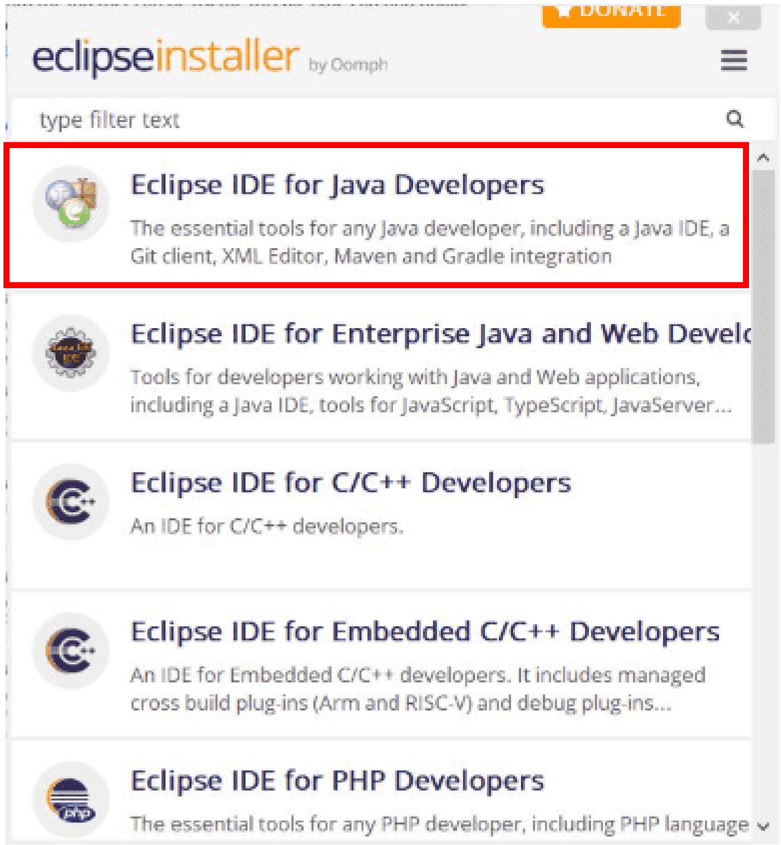
STEP 5: Open Download EXE

- 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".



Eclipse IDE for Java Developers

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration

Eclipse IDE for Enterprise Java and Web Develc

Tools for developers working with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer...

Eclipse IDE for C/C++ Developers

An IDE for C/C++ developers.

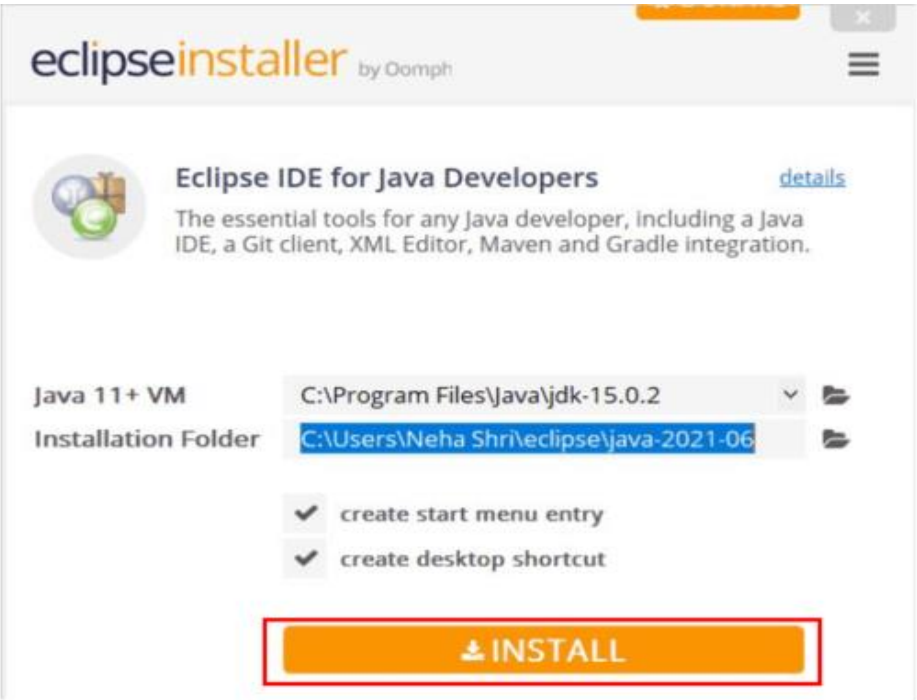
Eclipse IDE for Embedded C/C++ Developers

An IDE for Embedded C/C++ developers. It includes managed cross build plug-ins (Arm and RISC-V) and debug plug-ins...

Eclipse IDE for PHP Developers

The essential tools for any PHP developer, including PHP language

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



Eclipse IDE for Java Developers

The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration.

Java 11+ VM: C:\Program Files\Java\jdk-15.0.2

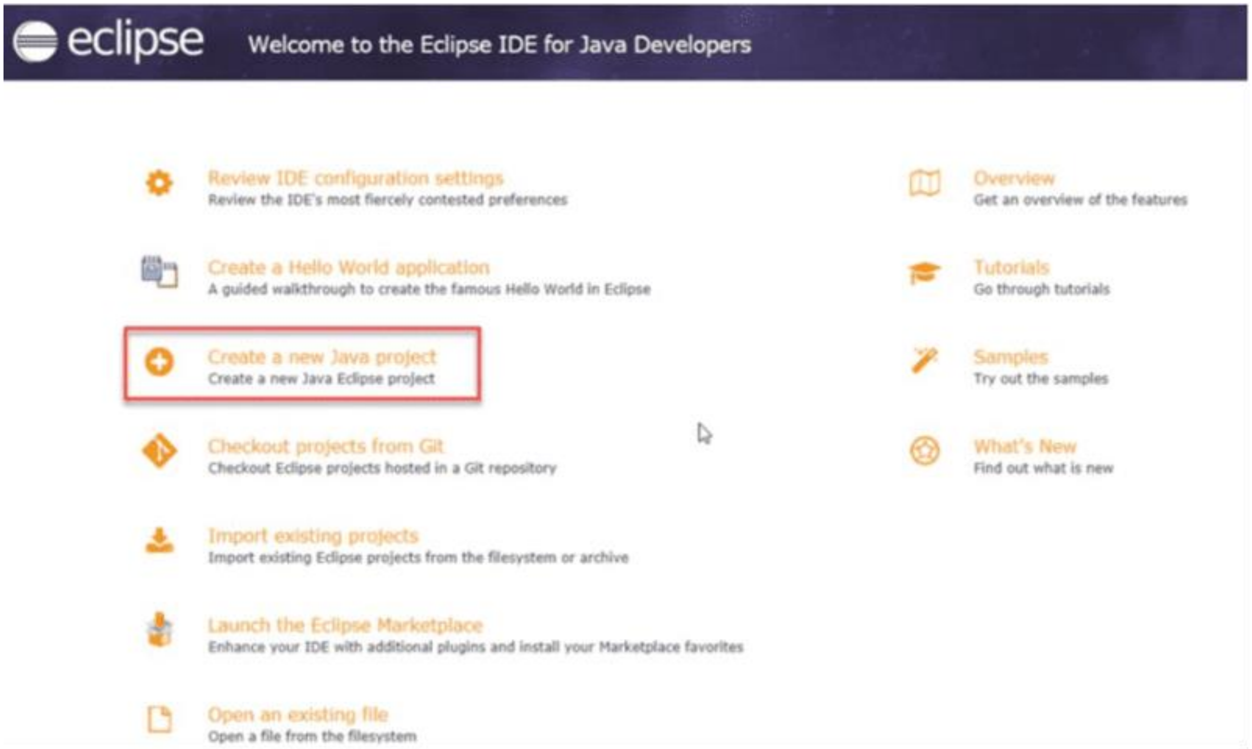
Installation Folder: C:\Users\Neha Shri\eclipse\java-2021-06

✓ create start menu entry

✓ create desktop shortcut

Step 8: Create New Project

Now click on "Create a new Java project".



Review IDE configuration settings
Review the IDE's most fiercely contested preferences

Create a Hello World application
A guided walkthrough to create the famous Hello World in Eclipse

Create a new Java project
Create a new Java Eclipse project

Checkout projects from Git
Checkout Eclipse projects hosted in a Git repository

Import existing projects
Import existing Eclipse projects from the filesystem or archive

Launch the Eclipse Marketplace
Enhance your IDE with additional plugins and install your Marketplace favorites

Open an existing file
Open a file from the filesystem

Overview
Get an overview of the features

Tutorials
Go through tutorials

Samples
Try out the samples

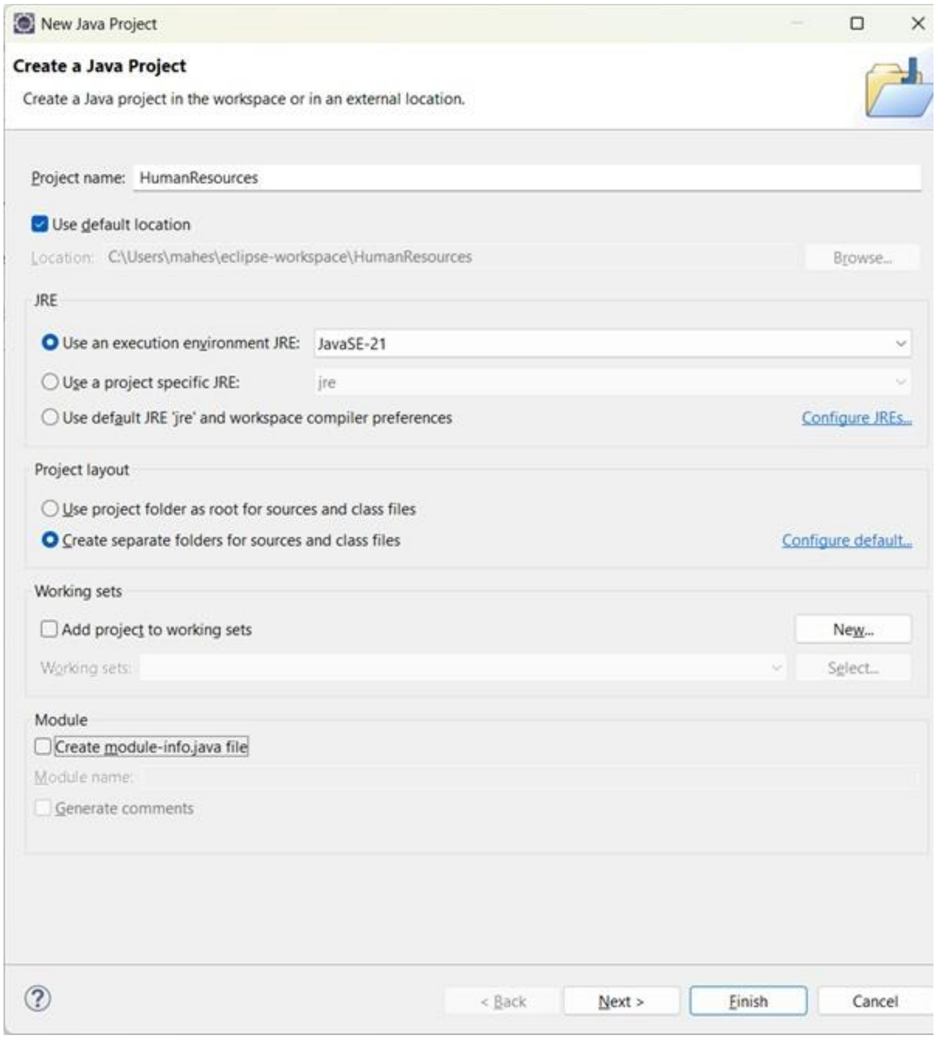
What's New
Find out what is new

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.
- 

New Java Project

Create a Java Project

Create a Java project in the workspace or in an external location.

Project name: HumanResources

☑ Use default location

Location: C:\Users\mahes\eclipse-workspace\HumanResources

JRE

● Use an execution environment JRE: JavaSE-21

○ Use a project specific JRE: jre

○ Use default JRE 'jre' and workspace compiler preferences

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files

Working sets

☐ Add project to working sets

Working sets:

Module

☐ Create module-info.java file

Module name:

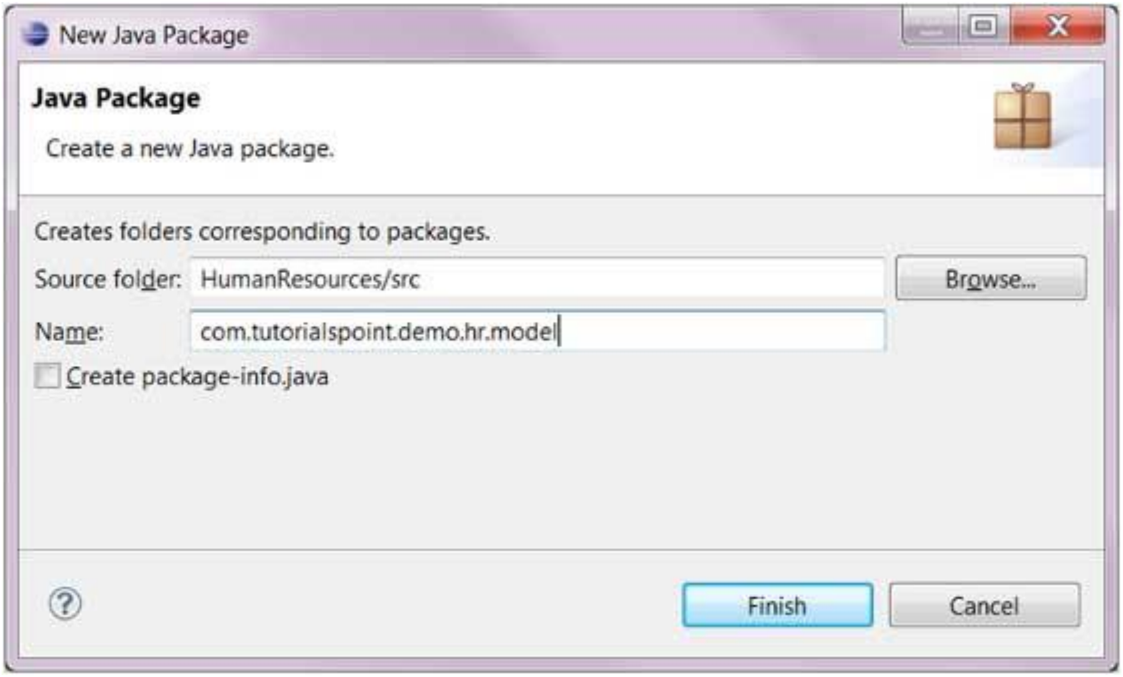
☐ Generate comments

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.



New Java Package

Java Package

Create a new Java package.

Creates folders corresponding to packages.

Source folder: HumanResources/src

Name: com.tutorialspoint.demo.hr.model

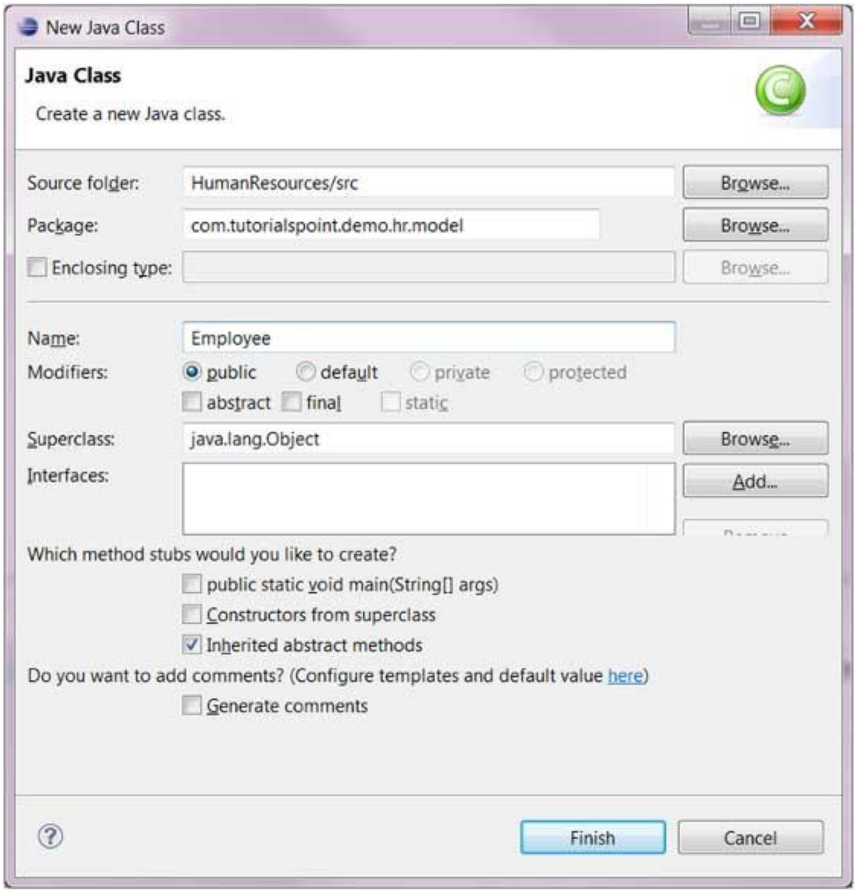
□ Create package-info.java

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 (C ▾) and selecting class (C).

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.


New Java Class

Java Class

Create a new Java class.

Source folder: HumanResources/src

Package: com.tutorialspoint.demo.hr.model

□ Enclosing type:

Name: Employee

Modifiers: ● public ○ default ○ private ○ protected

□ abstract □ final □ static

Superclass: java.lang.Object

Interfaces:

Which method stubs would you like to create?

□ public static void main(String[] args)

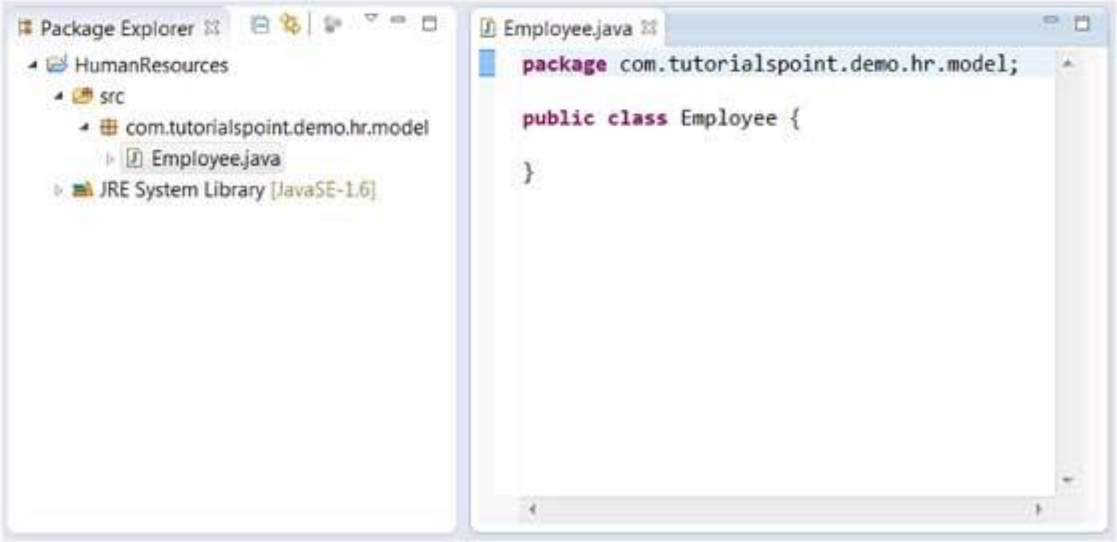
□ Constructors from superclass

☑ Inherited abstract methods

Do you want to add comments? (Configure templates and default value here)

□ Generate comments

STEP 14: Class created successfully.


Package Explorer
HumanResources
src
com.tutorialspoint.demo.hr.model
Employee.java
JRE System Library [JavaSE-1.6]
Employee.java

```
package com.tutorialspoint.demo.hr.model;

public class Employee {

}
```

BASIC PROGRAMS:

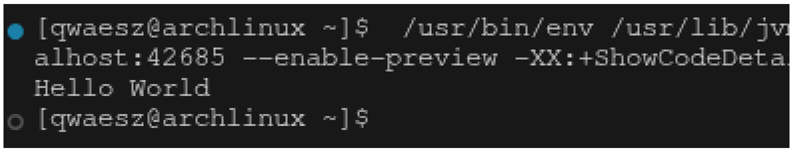
Program 1: Hello World Program

Source Code:

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

Output:

```
Hello World
```



```
[qwaesz@archlinux ~]$  /usr/bin/env /usr/lib/jv
alhost:42685 --enable-preview -XX:+ShowCodeDeta
Hello World
[qwaesz@archlinux ~]$
```

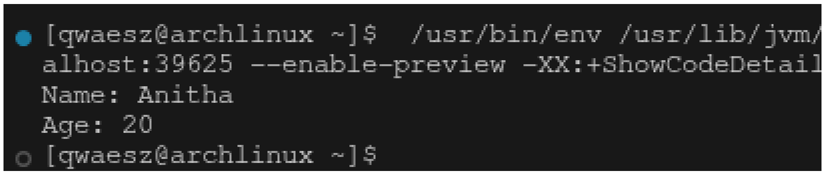
Program 2: Display Personal Details

Source Code:

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

Output:

```
Name: Anitha
Age: 20
```



```
[qwaesz@archlinux ~]$  /usr/bin/env /usr/lib/jvm/
alhost:39625 --enable-preview -XX:+ShowCodeDetail
Name: Anitha
Age: 20
[qwaesz@archlinux ~]$
```

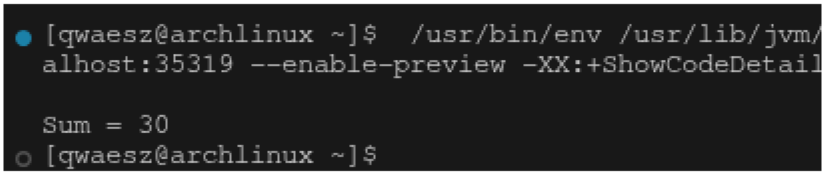
Program 3: Addition of Two Numbers

Source Code:

```
class AddTwoNumbers {

public static void main(String[] args) {

int a = 10, b = 20;

System.out.println("Sum = " + (a + b));

}

}
```

Output:

Sum = 30



```
● [qwaesz@archlinux ~]$  /usr/bin/env /usr/lib/jvm/
  alhost:35319 --enable-preview -XX:+ShowCodeDetail

  Sum = 30
○ [qwaesz@archlinux ~]$
```

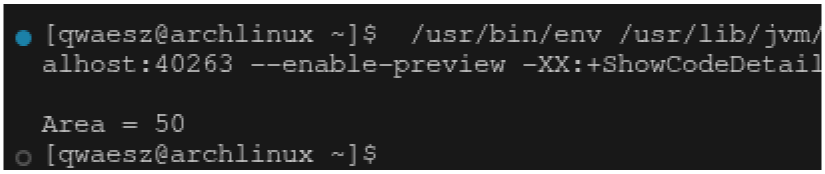
Program 4: Area of a Rectangle

Source Code:

```
class AreaRectangle {

public static void main(String[] args) {

int length = 10, breadth = 5;

System.out.println("Area = " + (length * breadth));

}

}
```

Output:

Area = 50



```
● [qwaesz@archlinux ~]$  /usr/bin/env /usr/lib/jvm/
  alhost:40263 --enable-preview -XX:+ShowCodeDetail

  Area = 50
○ [qwaesz@archlinux ~]$
```

Program 5: Simple Interest Calculation

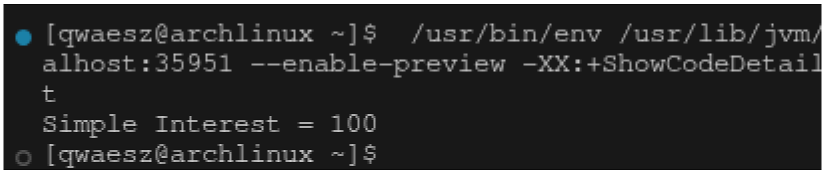
Source Code:

```
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



```
● [qwaesz@archlinux ~]$  /usr/bin/env /usr/lib/jvm/
  alhost:35951 --enable-preview -XX:+ShowCodeDetail
  t
  Simple Interest = 100
○ [qwaesz@archlinux ~]$
```

POST LAB EXERCISE

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

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

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

```
class DisplayInfo {

   public static void main(String[] args) {
      System.out.print("Name: Kavya ");
      System.out.print("Department: Computer Science");
   }
}
```

3. What happens if `main()` is written without `static`?

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

Reason:
The JVM calls the main() method without creating an object.
Without static, JVM cannot access main() and throws an error: Main method not found in class

4. Why is Java called platform independent?

   Java is called platform independent because Java programs are compiled into bytecode, which can run on any operating system that has a JVM.

   Key idea: Write Once, Run Anywhere (WORA). The JVM converts bytecode into machine code depending on the platform.

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

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

}
```

Result:

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

ASSESSMENT

Description	Max Marks	Marks Awarded
Pre Lab Exercise	**5**	
In Lab Exercise	**10**	
Post Lab Exercise	**5**	
Viva	**10**	

Total	**30**	
Faculty Signature		