Program No.1: Install and Setup java environment .Install java editor (Eclipse for Enterprise Java) and configure workspace. Execution of first java program. Java code execution process.

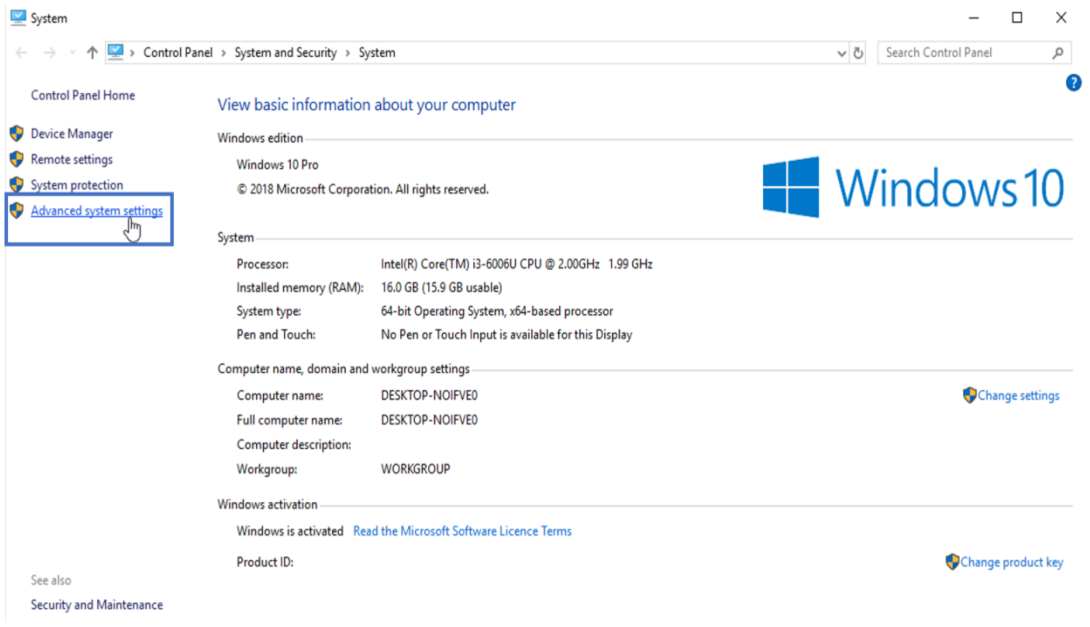
Java environment setup:

In today's world, Java is one of the most popular programming language which is being used by most skilled professionals. However, using it on the command line is not feasible sometimes. Therefore, to overcome this, we can use it on **Eclipse IDE.** Let's see how to setup Java environment on Eclipse IDE.

Step 1: Install Java
Step 2: Setup Eclipse IDE on Windows
Step 3: Hello World Program

Install Java:

- Follow the below steps to complete your Java installation.
- Go to the Java Downloads Page and click on the option of **Download.**
- Now, once the file is downloaded, run the installer and keep clicking on **Next**, till you finally get a dialog box, which say, you have finished installing.
- Once the installation is over follow the below instructions to set the path of the file.
- Now right click on ThisPC/ My Computer Icon-> Go to its properties and its **Advanced System Settings**. Refer below.


System
Control Panel > System and Security > System
Search Control Panel

Control Panel Home
Device Manager
Remote settings
System protection
Advanced system settings

View basic information about your computer

Windows edition
Windows 10 Pro
© 2018 Microsoft Corporation. All rights reserved.

Windows 10

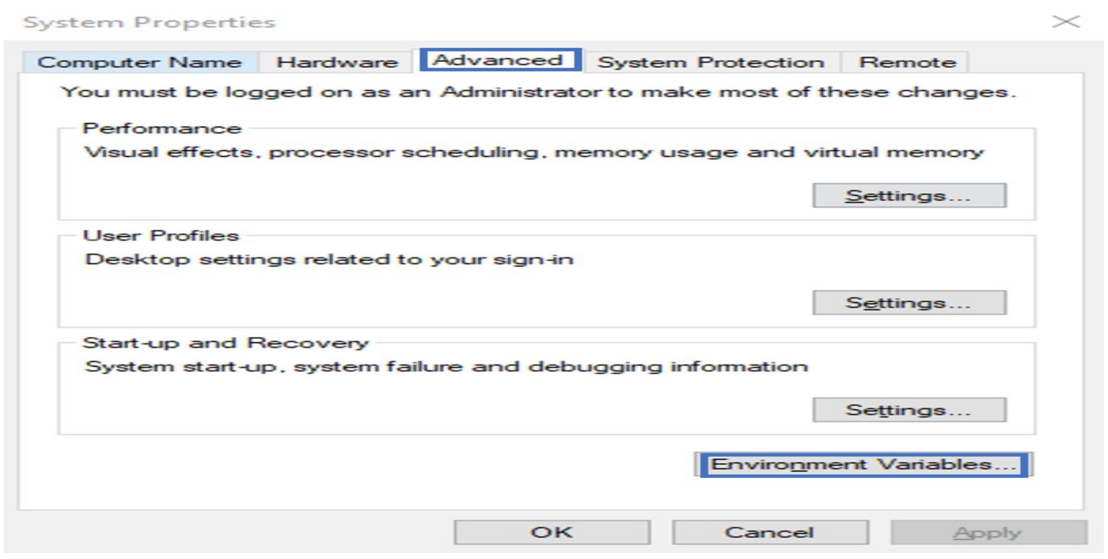
System
Processor: Intel(R) Core(TM) i3-6006U CPU @ 2.00GHz 1.99 GHz
Installed memory (RAM): 16.0 GB (15.9 GB usable)
System type: 64-bit Operating System, x64-based processor
Pen and Touch: No Pen or Touch Input is available for this Display

Computer name, domain and workgroup settings
Computer name: DESKTOP-NOIFVE0
Full computer name: DESKTOP-NOIFVE0
Computer description:
Workgroup: WORKGROUP
Change settings

Windows activation
Windows is activated Read the Microsoft Software Licence Terms
Product ID:
Change product key

See also
Security and Maintenance

- Now, click on '**Environment Variables**' under '**Advanced**' tab as shown below:



You must be logged on as an Administrator to make most of these changes.

Performance

Visual effects, processor scheduling, memory usage and virtual memory

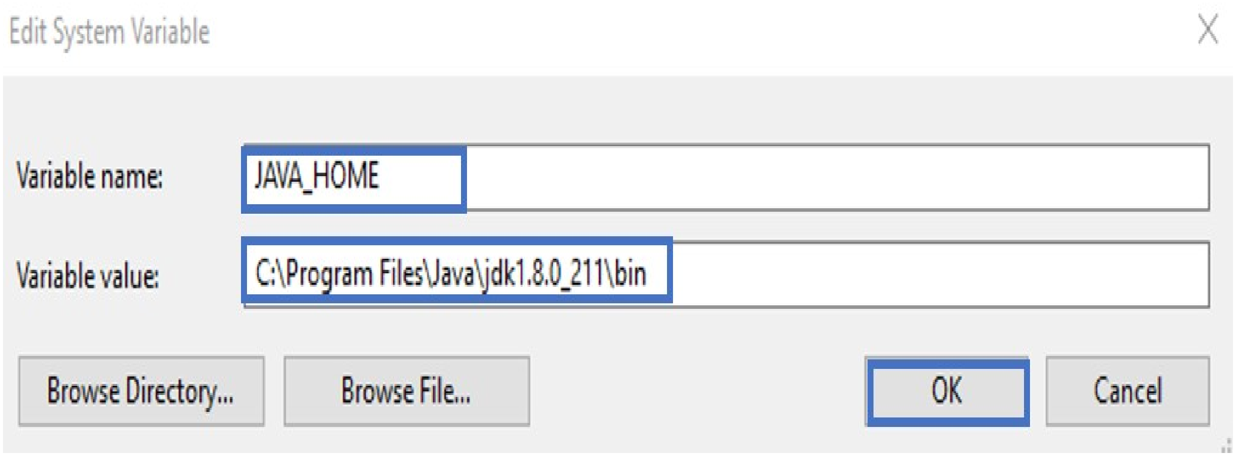
User Profiles

Desktop settings related to your sign-in

Start-up and Recovery

System start-up, system failure and debugging information

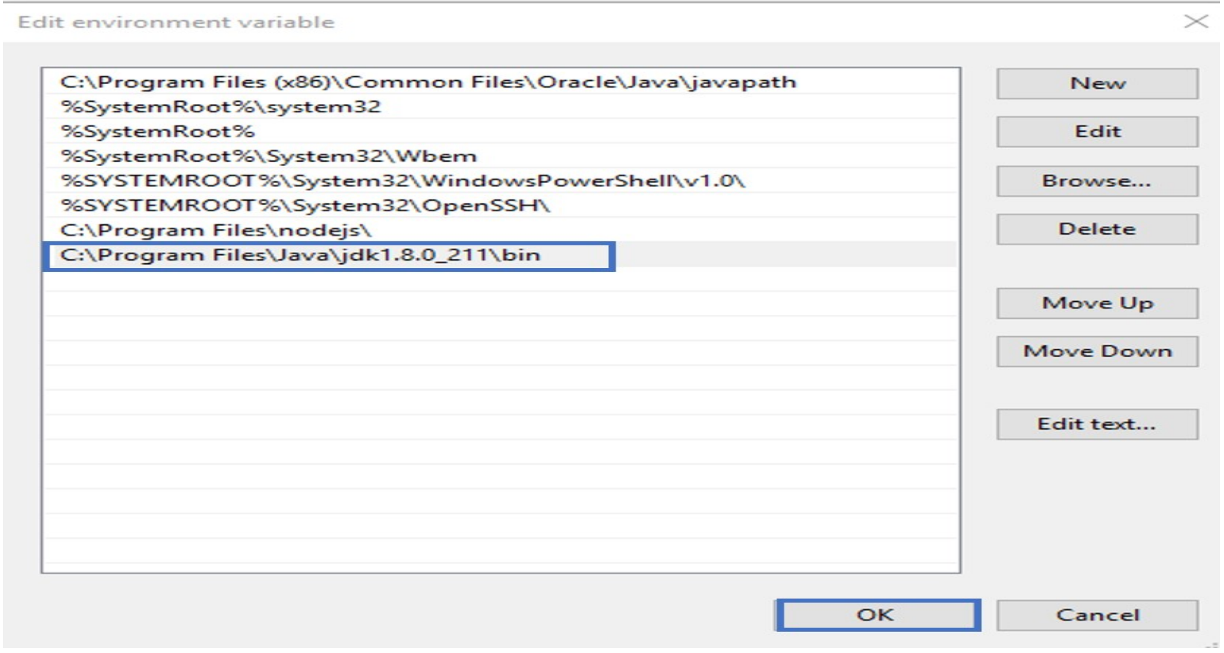
➢ Next, under **System Variables** choose **New.** Enter the variable name as '**JAVA_HOME**' and the full path to Java installation directory as per your system as shown below:



Variable name: JAVA_HOME

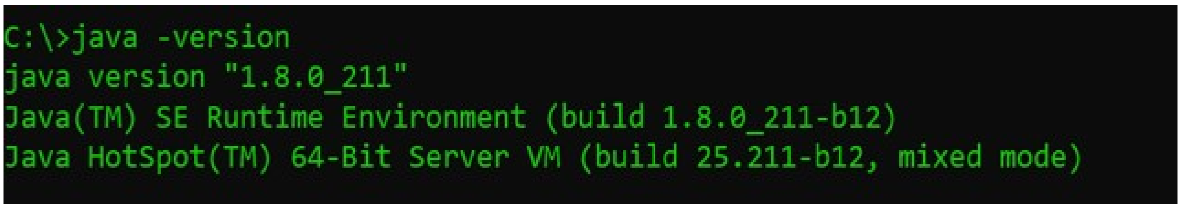
Variable value: C:\Program Files\Java\jdk1.8.0_211\bin

➢ Next thing that you have to do is to configure your environment variables. Let's see how to do that. Here, you have to edit the path of the system variable as shown below. Then click OK.



```
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\nodejs\
C:\Program Files\Java\jdk1.8.0_211\bin
```

➢ Now to cross-check the installation, just run following command in cmd – ***java -version***. It should display the installed version of Java in your system.



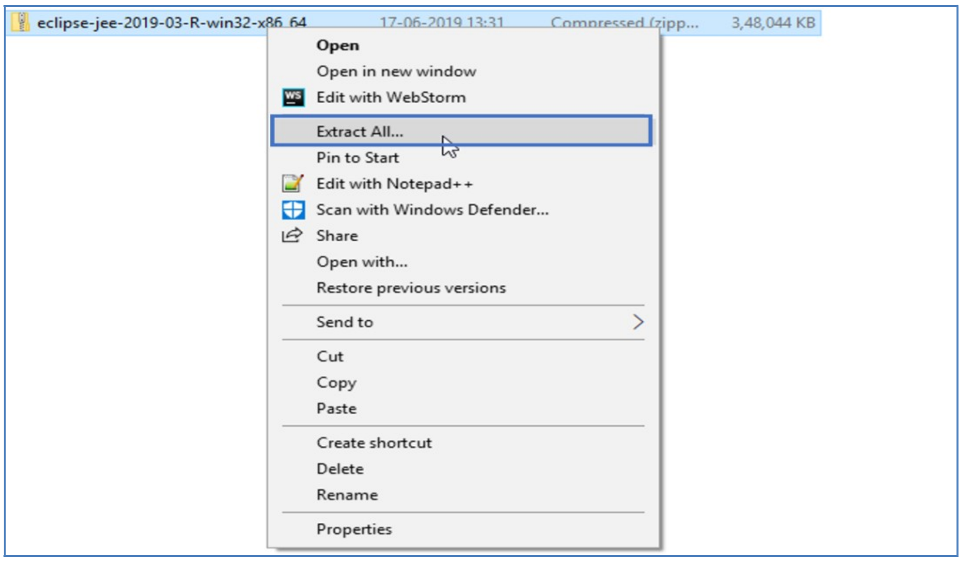
```
C:\>java -version
java version "1.8.0_211"
Java(TM) SE Runtime Environment (build 1.8.0_211-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.211-b12, mixed mode)
```

Install Eclipse:

Follow the below steps to configure Eclipse on your system:

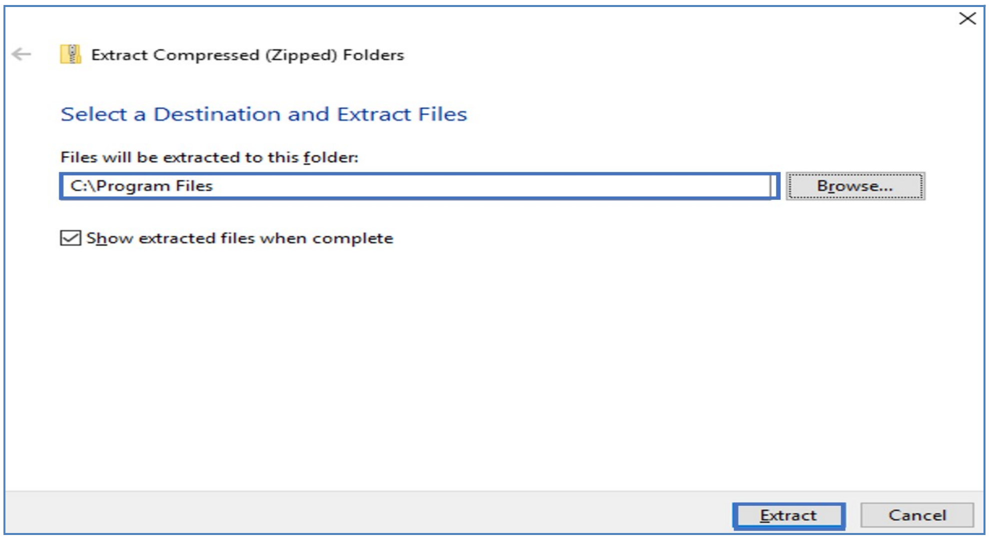
Step 1: Navigate to the following URL – https://www.eclipse.org/downloads/packages/ and select the download link depending on your system architecture – (Windows, Mac OS or Linux) and download it.

Step 2: Once the download is over, extract the zipped file by right-clicking on the folder and choose **Extract All**. Refer below.



eclipse-jee-2019-03-R-win32-x86_64 17-06-2019 13:31 Compressed (zipp... 3,48,044 KB

Step 3: You will be then redirected to a dialog box, where you have to choose the directory in which you wish to extract the files. Then click on **Extract**. Refer below.



Extract Compressed (Zipped) Folders

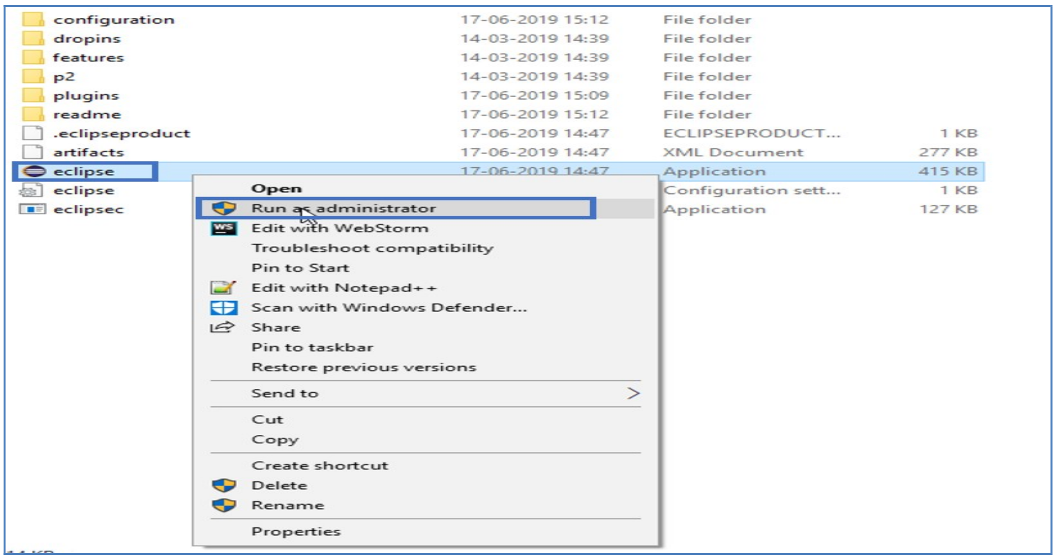
Select a Destination and Extract Files

Files will be extracted to this folder:

C:\Program Files

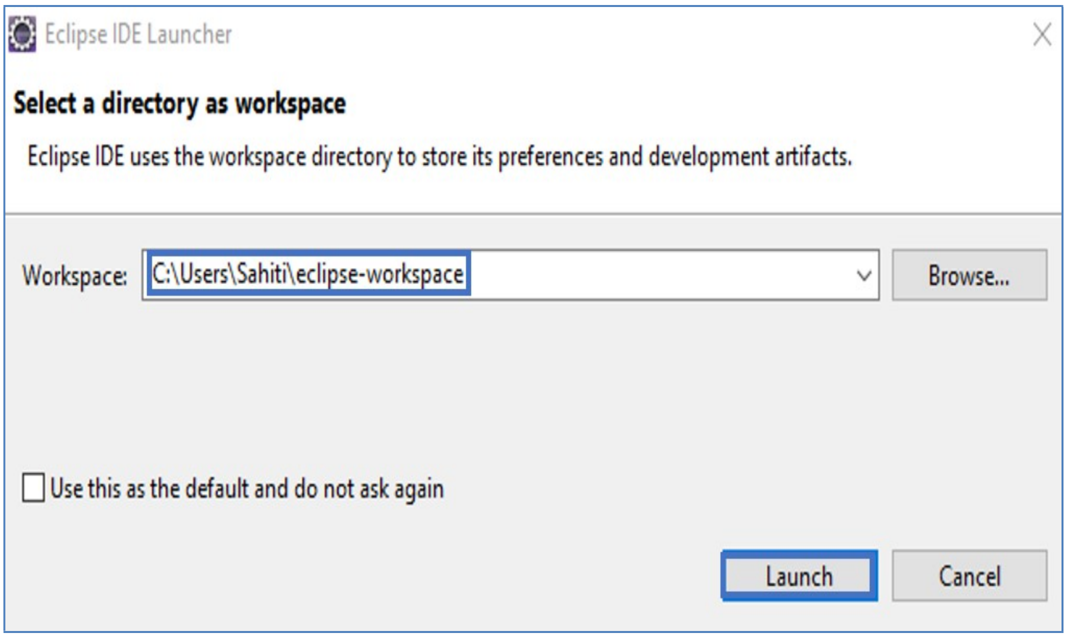
☑ Show extracted files when complete

Step 4: After extracting files, open the folder and launch **eclipse.exe.**



configuration	17-06-2019 15:12	File folder	
dropins	14-03-2019 14:39	File folder	
features	14-03-2019 14:39	File folder	
p2	14-03-2019 14:39	File folder	
plugins	17-06-2019 15:09	File folder	
readme	17-06-2019 15:12	File folder	
.eclipseproduct	17-06-2019 14:47	ECLIPSEPRODUCT...	1 KB
artifacts	17-06-2019 14:47	XML Document	277 KB
eclipse	17-06-2019 14:47	Application	415 KB
eclipse		Configuration sett...	1 KB
eclipsec		Application	127 KB

Step 5: Then, you have to choose the Launch directory for Eclipse and then click on Launch. Refer below.



Eclipse IDE Launcher

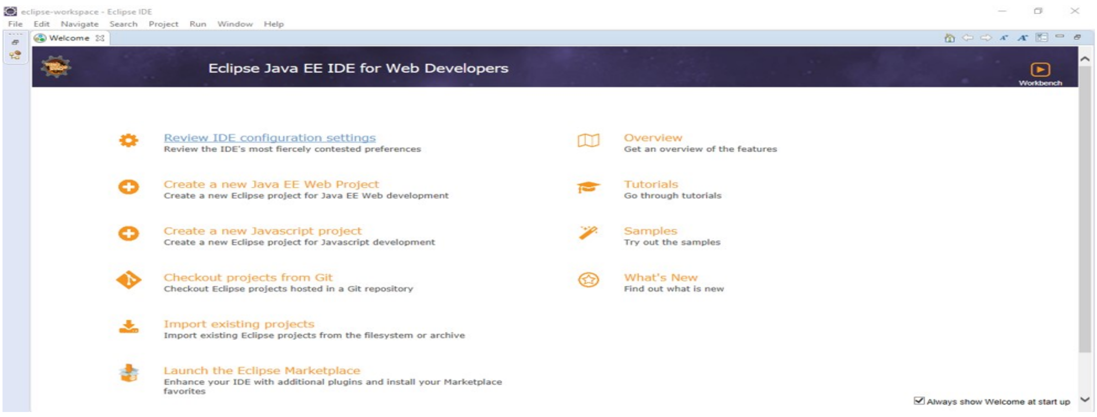
Select a directory as workspace

Eclipse IDE uses the workspace directory to store its preferences and development artifacts.

Workspace: C:\Users\Sahiti\eclipse-workspace

☐ Use this as the default and do not ask again

Step 6: Once Eclipse launches, you will see the below window:



eclipse-workspace - Eclipse IDE

Eclipse Java EE IDE for Web Developers

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

Create a new Java EE Web Project
Create a new Eclipse project for Java EE Web development

Create a new Javascript project
Create a new Eclipse project for Javascript development

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

Overview
Get an overview of the features

Tutorials
Go through tutorials

Samples
Try out the samples

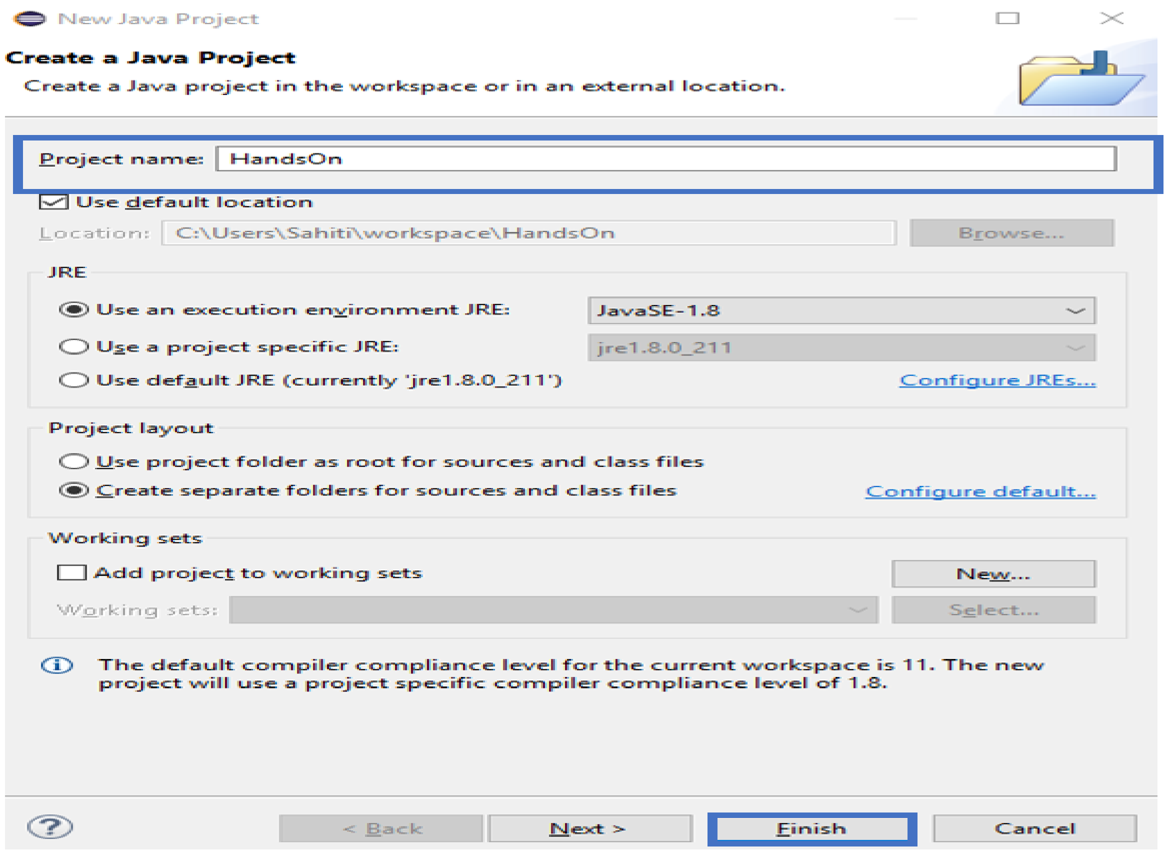
What's New
Find out what is new

☑ Always show Welcome at start up

Executing first Java Program : Hello World Program:

Step 1: Launch **Eclipse IDE** and go to **File ->New -> Java Project**

Step 2: Mention the **project name** and click on **Finish.**



New Java Project

Create a Java Project

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

Project name: HandsOn

☑ Use default location

Location: C:\Users\Sahiti\workspace\HandsOn Browse...

JRE

● Use an execution environment JRE: JavaSE-1.8

○ Use a project specific JRE: jre1.8.0_211

○ Use default JRE (currently 'jre1.8.0_211') Configure JREs...

Project layout

○ Use project folder as root for sources and class files

● Create separate folders for sources and class files Configure default...

Working sets

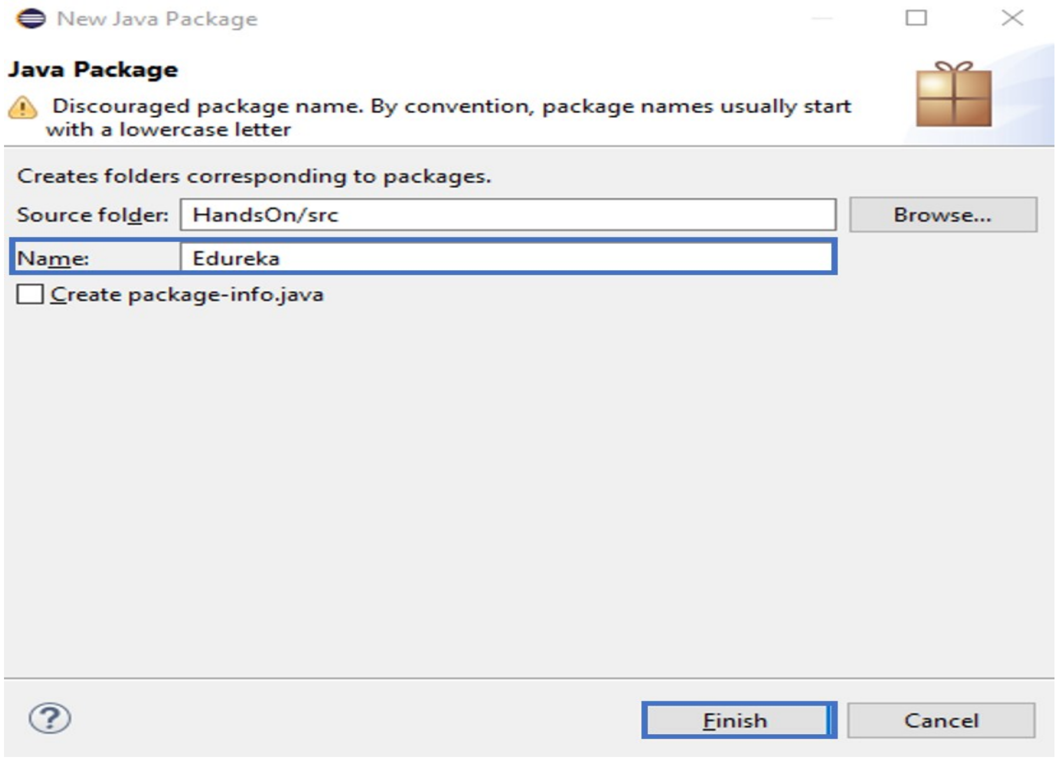
☐ Add project to working sets New...

Working sets: Select...

The default compiler compliance level for the current workspace is 11. The new project will use a project specific compiler compliance level of 1.8.

< Back Next > Finish Cancel

Step 3: Now, go to the **Project**, Right-Click on the Project and choose **Package**. In the dialog box, which opens up, mention the **Package name** as below and click on **Finish.**



New Java Package

Java Package

Discouraged package name. By convention, package names usually start with a lowercase letter

Creates folders corresponding to packages.

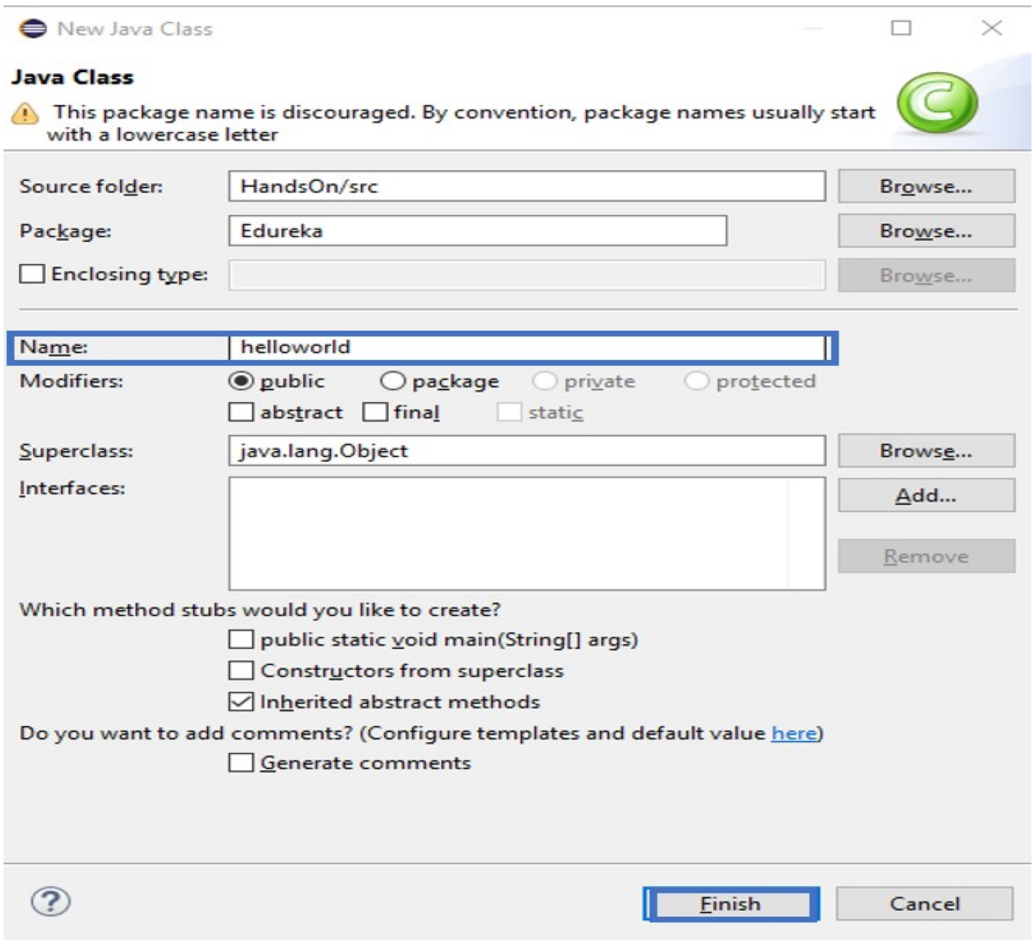
Source folder: HandsOn/src Browse...

Name: Edureka

☐ Create package-info.java

Finish Cancel

Step 4: Now, right click on the **Package**, go to **New** and choose **Class**. Mention the **class name** and click on **Finish**. Refer below.



New Java Class

Java Class

This package name is discouraged. By convention, package names usually start with a lowercase letter

Source folder: HandsOn/src

Package: Edureka

☐ Enclosing type:

Name: helloworld

Modifiers: ● public ○ package ○ 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 5: Now, mention the following code in the workspace.

```
public class helloworld
{
  public static void main(String args[])
        {

                    System.out.println("Hello World");
        }
}
```

Step 6: Now, execute your file, by right-clicking on the **helloworld.java file** and choose **Run As -> Java Application.**

Program No.2: Code, execute and debug programs that uses different types of variables and datatypes; Identify and resolve issues in the given code snippet.

```
class Datatypes
{
 public static void main(String[] args)
  {
        byte myByte1,myByte2;
        myByte1 = 120;
        myByte2 = -48;
        System.out.println("Byte1: " +myByte1);
        System.out.println("Byte2: " +myByte2);
        myByte1++; // Looping back within the range
        System.out.println("Incremented Value of Byte1:" +myByte1);

        short myShort = 6000;
        System.out.println("\nShort:" +myShort);

        int myInteger1, myInteger2, result;
        myInteger1 = -7000;
        myInteger2 = 9000;
        result = myInteger1 + myInteger2;
        System.out.println("\nInteger1:"+myInteger1);
        System.out.println("Integer2:"+myInteger2);
        System.out.println("Integer1 + Integer2: " +result);

        long myLong1, myLong2, result1;
        myLong1 = 100000000L;
        myLong2 = 200L;
        result1 = myLong1 * myLong2;
        System.out.println("\nLong1: " +myLong1);
        System.out.println("Long2: " +myLong2);
        System.out.println("Long1 * Long2: " +result1);

        float myFloat1,myFloat2,result2;
        myFloat1=1000.666f;
        myFloat2=110.77f;
        result2=myFloat1-myFloat2;
        System.out.println("\nFloat1: "+myFloat1);
        System.out.println("Float2: "+myFloat2);
        System.out.println("Float1-Float2: "+result2);

        double myDouble1, myDouble2, result3;
        myDouble1 = 48976.8987;
        myDouble2 = 29513.7812d;
        result3 = myDouble1 + myDouble2;
        System.out.println("\nDouble1: " +myDouble1);
        System.out.println("Double2: " +myDouble2);
        System.out.println("Double1 + Double2: " +result3);
        boolean myBool = true;
        if(myBool == true)
        System.out.println("\nI am using a Boolean data type");
        System.out.println(myBool);

        char myChar1 = 'A';
        char myChar2 = 66;
        System.out.println("\nmyChar1: " +myChar1);
        System.out.println("myChar2: " +myChar2);
        myChar2++; // valid increment operation
        System.out.println("The Incremented value of myChar2: " +myChar2);

        String string1 = "\nGPT ATHANI"; // declaring string using string literal
        System.out.println(string1);
}
}
```

Output:

```
        Byte1: 120
        Byte2: -48
        Incremented Value of Byte1:121

        Short: 6000

        Integer1:-7000
        Integer2:9000
        Integer1 + Integer2: 2000

        Long1: 100000000
        Long2: 200
        Long1 * Long2: 20000000000

        Float1: 1000.666
        Float2: 110.77
        Float1-Float2: 889.896

        Double1: 48976.8987
        Double2: 29513.7812
        Double1 + Double2: 78490.6799

        I am using a Boolean data type
        true

        myChar1: A
        myChar2: B
        The Incremented value of myChar2: C

        Hello World
```

Program No.3: Code, execute and debug programs that uses different types of constructors. Identify and resolve issues in the given code snippet.

```
class Student
{
        String name;
        int regno;
        Student()    //Constructor
          {
                name="Raju";
                regno=1234;
          }
     Student(String n, int  r) // parameterized constructor
         {
                name=n;
                regno=r;
          }
     Student(Student s)   // copy constructor
        {
                name=s.name;
                regno=s.regno;
        }
        void display()
        {
                System.out.println(name + "\t" +regno);
        }
  }
class StudentDemo
   {
        public static void main(String args[])
          {
                Student s1=new Student();
                Student s2=new Student("Ravi",1489);
                Student s3=new Student(s1);
                s1.display();
                s2.display();
                s3.display();
          }
    }
```

Output:

```
Raju 1234
Ravi 1489
Raju 1234
```

Program No.4: Code, execute and debug program to perform autoboxing and unboxing. Identify and resolve issues in the given code snippet.

```
class Conversion
    {
     public static void main(String args[])
       {
           byte b=10;
           short s=20;
           int i=30;
           long l=40;
           float f=50.0F;
           double d=60.0D;
           char c='a';
           boolean b2=true;

           //Autoboxing: Converting primitives into objects
           Byte byteobj=b;
           Short shortobj=s;
           Integer intobj=i;
           Long longobj=l;
           Float floatobj=f;
           Double doubleobj=d;
           Character charobj=c;
           Boolean boolobj=b2;

           //Printing objects
           System.out.println("---Printing object values---");
           System.out.println("Byte object: "+byteobj);
           System.out.println("Short object: "+shortobj);
           System.out.println("Integer object: "+intobj);
           System.out.println("Long object: "+longobj);
           System.out.println("Float object: "+floatobj);
           System.out.println("Double object: "+doubleobj);
           System.out.println("Character object: "+charobj);
           System.out.println("Boolean object: "+boolobj);

           //Unboxing: Converting Objects to Primitives
           byte bytevalue=byteobj;
           short shortvalue=shortobj;
           int intvalue=intobj;
           long longvalue=longobj;
           float floatvalue=floatobj;
           double doublevalue=doubleobj;
           char charvalue=charobj;
           boolean boolvalue=boolobj;

           //Printing primitives
           System.out.println("---Printing primitive values---");
           System.out.println("byte value: "+bytevalue);
           System.out.println("short value: "+shortvalue);
```

```
                System.out.println("int value: "+intvalue);
                System.out.println("long value: "+longvalue);
                System.out.println("float value: "+floatvalue);
                System.out.println("double value: "+doublevalue);
                System.out.println("char value: "+charvalue);
                System.out.println("boolean value: "+boolvalue);
        }
}
```

Output:

```
---Printing object values---
Byte object: 10
Short object: 20
Integer object: 30
Long object: 40
Float object: 50.0
Double object: 60.0
Character object: a
Boolean object: true

---Printing primitive values---
byte value: 10
short value: 20
int value: 30
long value: 40
float value: 50.0
double value: 60.0
char value: a
boolean value: true
```

Program No.5: Code, execute and debug program to perform evaluation of expression. Identify and resolve issues in the given code snippet.

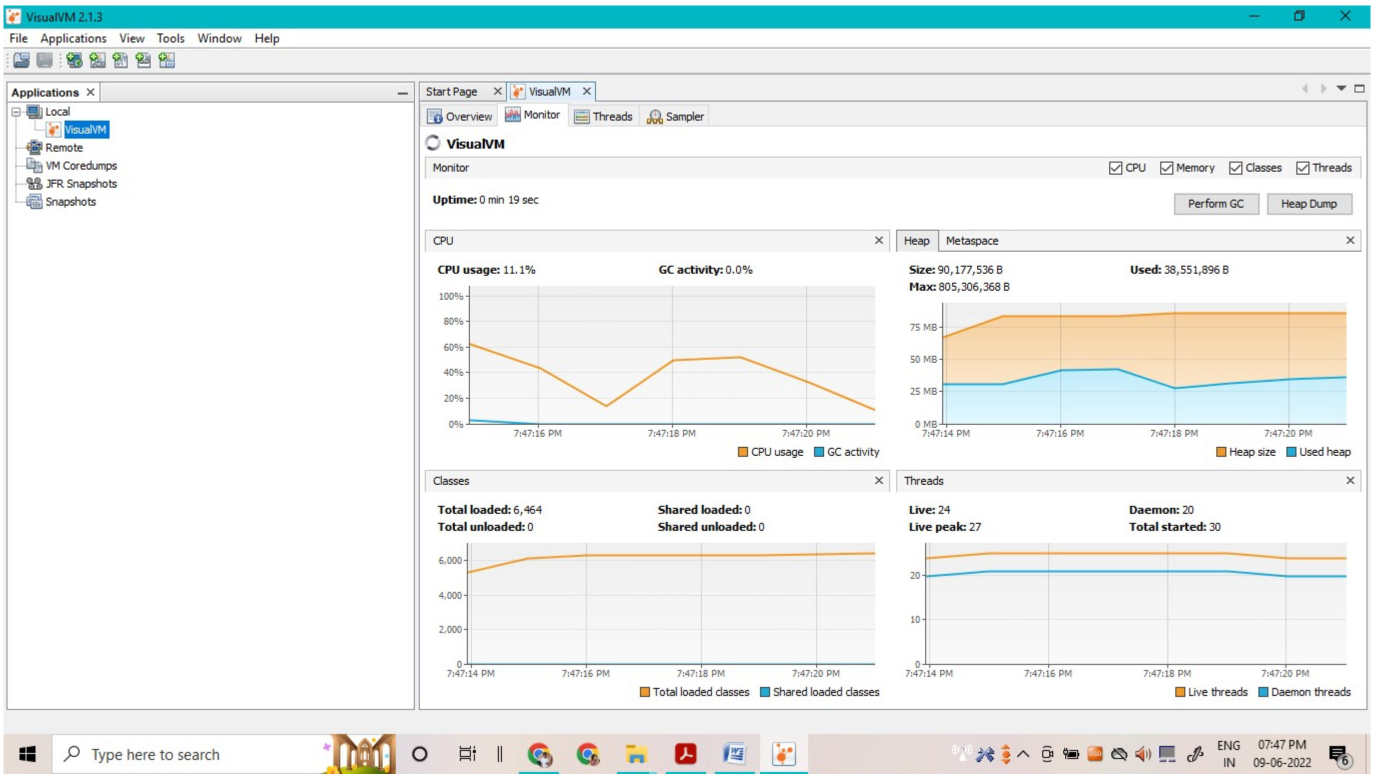
```
import java.util.Scanner;
class ExpressionEvaluation
{
 public static void main(String[] args)
 {
   Scanner input = new Scanner(System.in);
   System.out.print("Enter Equation you want to evaluate : ");
   String string = input.nextLine();
   String a = string.replaceAll(" ","");

  if (a.length() < 3)
   {
     System.out.println( "Please Enter Minimum One Opearator and Two Opearands");
     System.exit(0);
   }
  int result = 0;
  int i = 0;
  while(a.charAt(i)!='+' && a.charAt(i)!='-' && a.charAt(i)!='*' && a.charAt(i)!='/')
   {
      i++;
   }
  switch (a.charAt(i))
    {
           case '+' :
            result = Integer.parseInt(a.substring(0,i))+Integer.parseInt(a.substring(i+1,a.length()));
            break;
           case '-' :
            result = Integer.parseInt(a.substring(0,i))-Integer.parseInt(a.substring(i+1,a.length()));
            break;
           case '*' :
            result = Integer.parseInt(a.substring(0,i))*Integer.parseInt(a.substring(i+1,a.length()));
            break;
           case '/' :
            result = Integer.parseInt(a.substring(0,i))/Integer.parseInt(a.substring(i+1,a.length()));
            break;
    }
  System.out.println(a.substring(0,i) + ' ' + a.charAt(i) + ' ' + a.substring(i+1,a.length())+ " = " + result);
 }
}
```

Output:

```
Enter Equation: 10+20
10 + 20 = 30
```

Program No.6: Install memory monitoring tool and observe how JVM allocates Memory.

- **VisualVM** provides detailed information about Java applications while they are running on the Java Virutal Machine (JVM). VisualVM's graphical user interface enables us to quickly and easily see information about multiple Java applications.
- **Installing VisualVM**
  - ➢ Download the VisualVM installer from the VisualVM project page.
  - ➢ Extract the VisualVM installer to an empty directory on local system.
- **Starting VisualVM**
  - ➢ To start VisualVM on Windows, run the **visualvm.exe** program that is in the \bin folder under the VisualVM install folder.
  - ➢ After opening VisualVM, double click on VisualVM icon which appears on left panel.
- **Exploring VisualVM:**
  - ➢ Click on overview tab to know more about JVM arguments, System Properties etc.
  - ➢ Click on monitor tab to understand about CPU usage, Heap Space, Classes and Threads. Observe how memory is managed by JVM.
  - ➢ Click on Heap Dump(right-side corner) in monitor tab to know about classes, instances, and environment.
  - ➢ Click on threads to know how many types threads are running in live and finished threads.
  - ➢ Click on sampler to know CPU samples, Memory Samples.



Uptime: 0 min 19 sec

CPU usage: 11.1% GC activity: 0.0%

Size: 90,177,536 B Used: 38,551,896 B
Max: 805,306,368 B

Total loaded: 6,464 Shared loaded: 0
Total unloaded: 0 Shared unloaded: 0

Live: 24 Daemon: 20
Live peak: 27 Total started: 30

Program No.7: Explain memory allocation through Java programs.

What is Stack Memory?

- Stack in Java is a section of memory which contains methods, local variables, and reference variables. Stack memory is always referenced in Last-In-First-Out order. Local variables are created in the stack.

What is Heap Memory?

- Heap is a section of memory which contains Objects and may also contain reference variables. Instance variables are created in the heap.

Memory Allocation in Java:

- **Memory Allocation in Java** is the process in which the virtual memory sections are set aside in a program for storing the variables and instances of structures and classes. However, the memory isn't allocated to an object at declaration but only a reference is created. For the memory allocation of the object, **new()** method is used, so the object is always allocated memory on the heap.
- The Java Memory Allocation is divided into following sections :
  1. Heap
  2. Stack
  3. Code
  4. Static
- This division of memory is required for its effective management.
  - The **code** section contains your **bytecode**.
  - The **Stack** section of memory contains **methods, local variables, and reference variables.**
  - The **Heap** section contains **Objects** (may also contain reference variables).
  - The **Static** section contains **Static data/methods**.

Difference between Local and Instance Variable

Instance variable is declared **inside a class but not inside a method**

```
class Student
 {
        int num; // num is  instance variable
        public void showData{}
 }
```

Local variables are declared **inside** a **method including** method **arguments**.

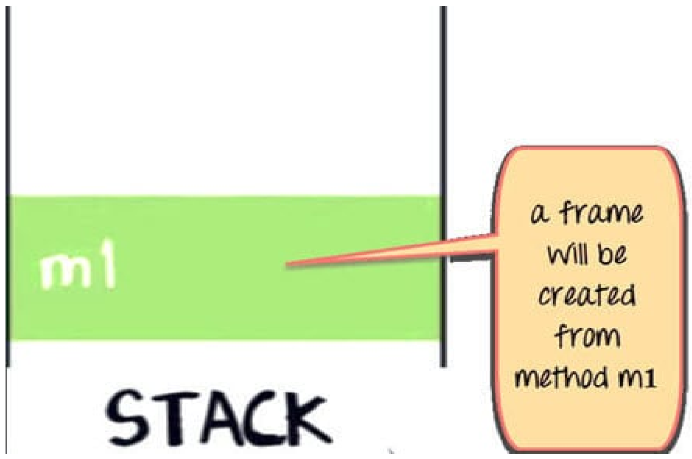
```
public void sum(int a)
{
    int x = int a +  3;    // a , x are local variables;
 }
```

Difference between Stack and Heap

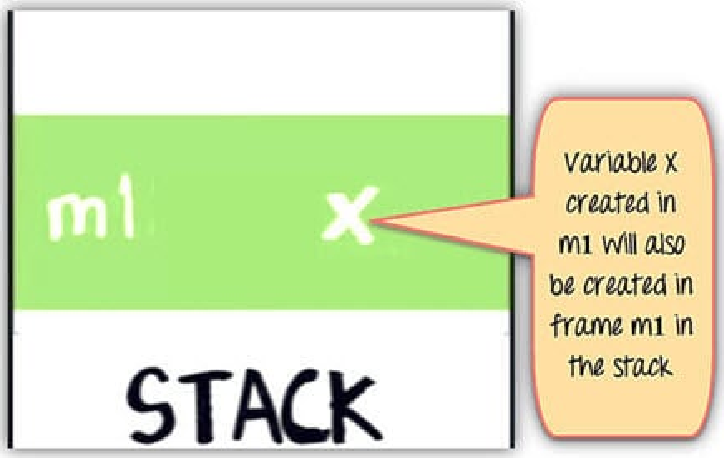
- Let's take an example to understand this better.
- Consider that the main method calling method **m1**

```
public void m1
{
   int x=20;
}
```

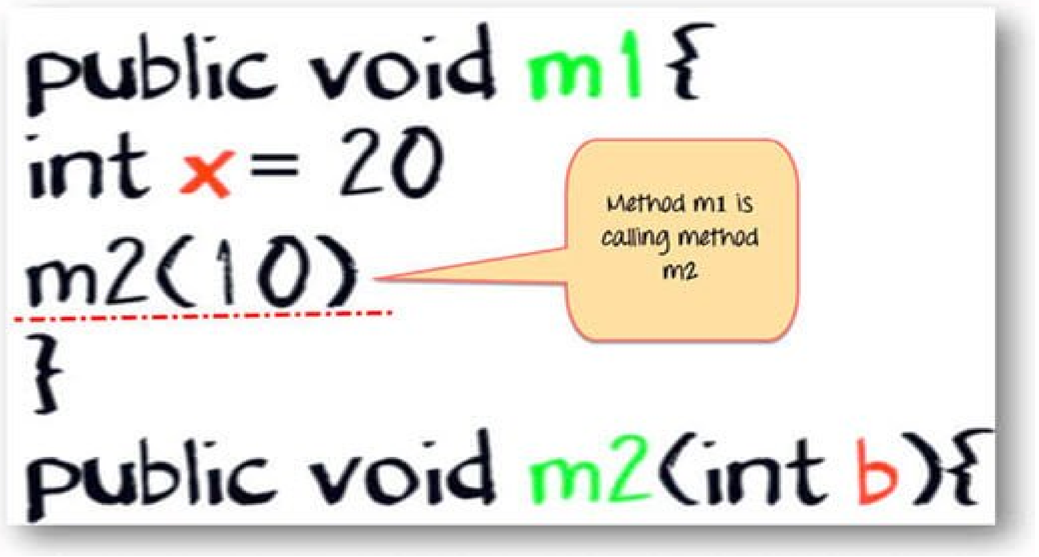
In the stack java, a frame will be created from method m1.



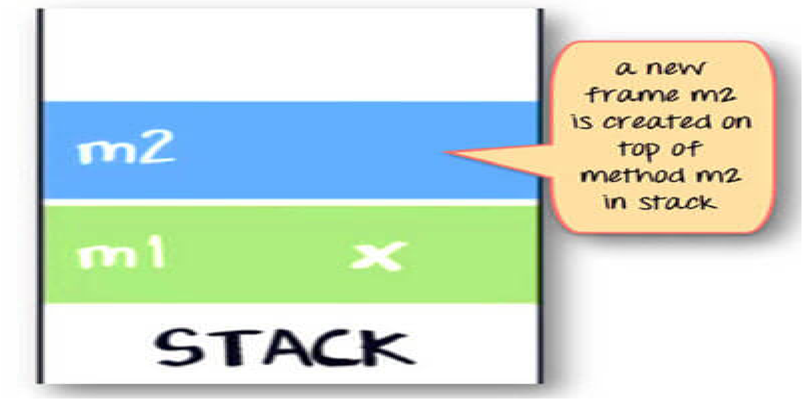
The variable **x** in m1 will also be created in the frame for m1 in the stack. (See image below).


m1
x
Variable x created in m1 will also be created in frame m1 in the stack
STACK

As shown below Method m1 is calling method m2. In the stack Java, a new frame is created for m2 on top of the frame m1.

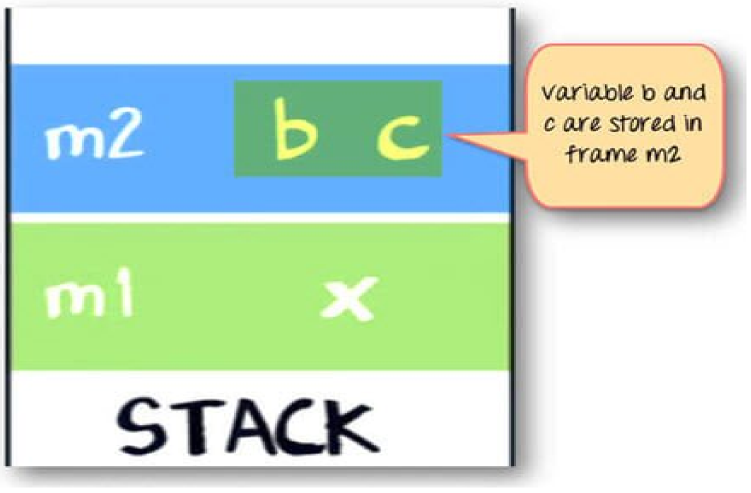


```
public void m1{
int x= 20
m2(10)
}
public void m2(int b){
```

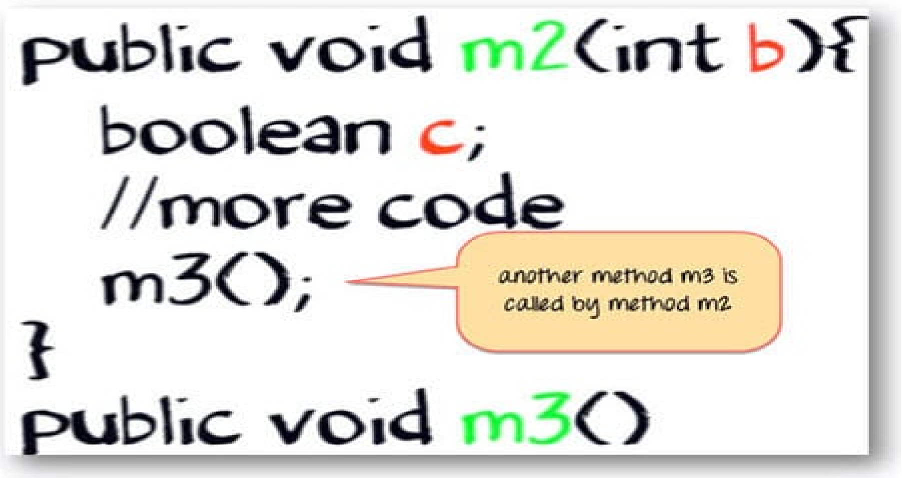


Variable b and c will also be created in a frame m2 in a stack.

```
public void m2(int b)
{
   boolean c;
}
```

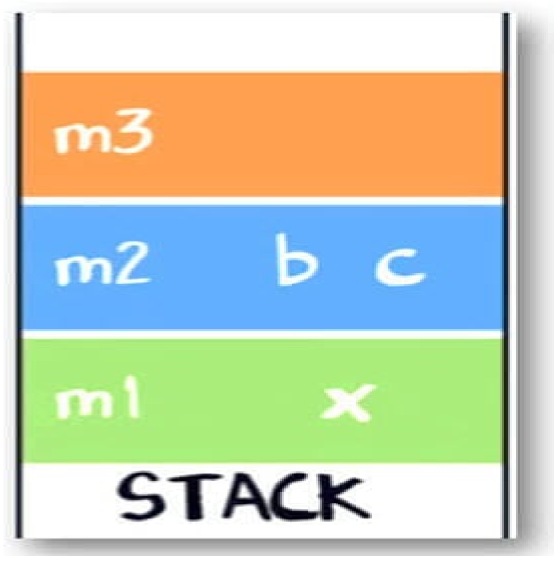


As shown below same method m2 is calling method m3. Again a frame m3 is created on the top of the stack (see image below).



```
public void m2(int b){
    boolean c;
    //more code
    m3();
}
public void m3()
```

another method m3 is called by method m2



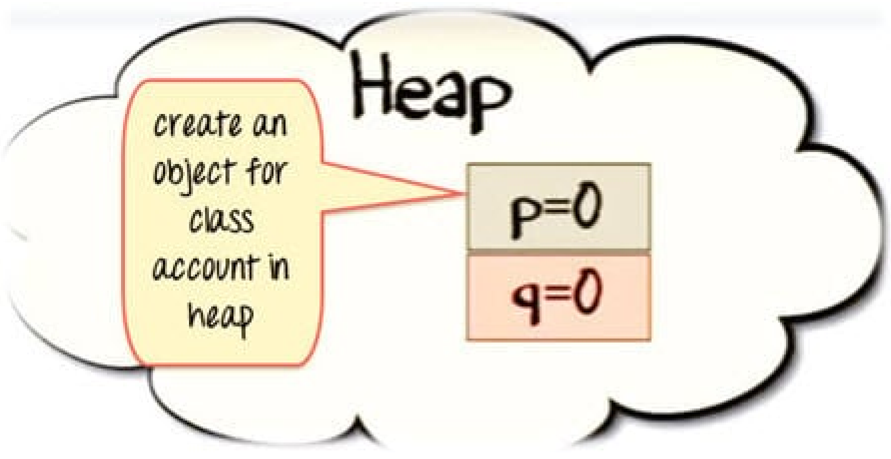
Now let say our method m3 is creating an object for class **"Account,"** which **has two instances variables int p and int q.**

```
Account
   {
          int p;
          int q;
      }
```

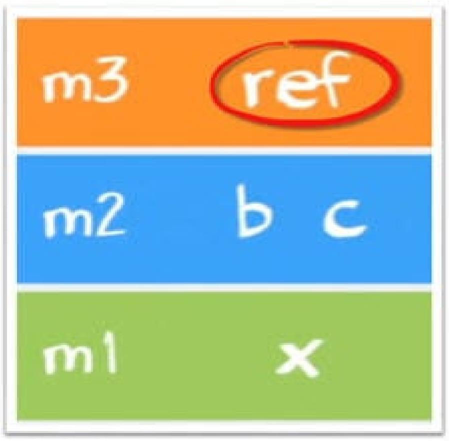
Here is the code for method m3

```
public void m3()
  {
           Account ref = new Account();
           // more code
  }
```

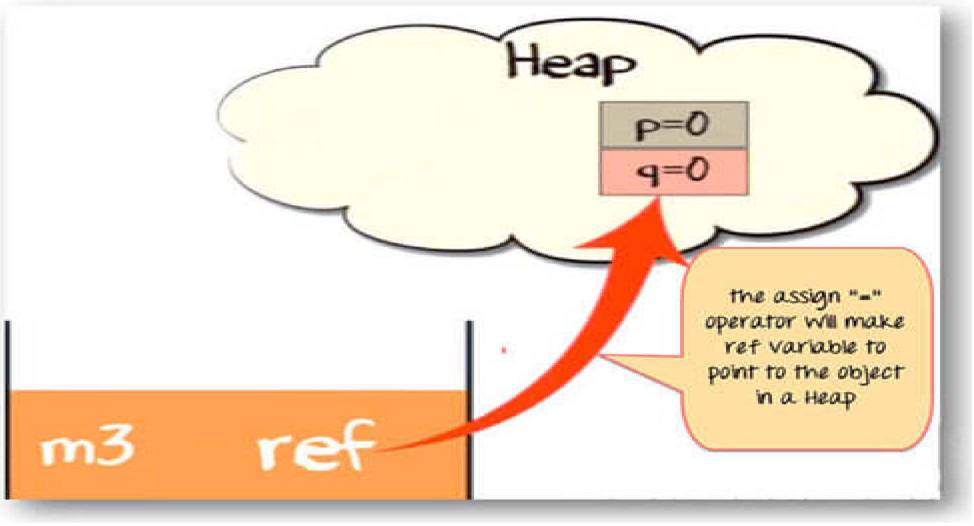
The statement new Account() will create an object of account in **heap.**



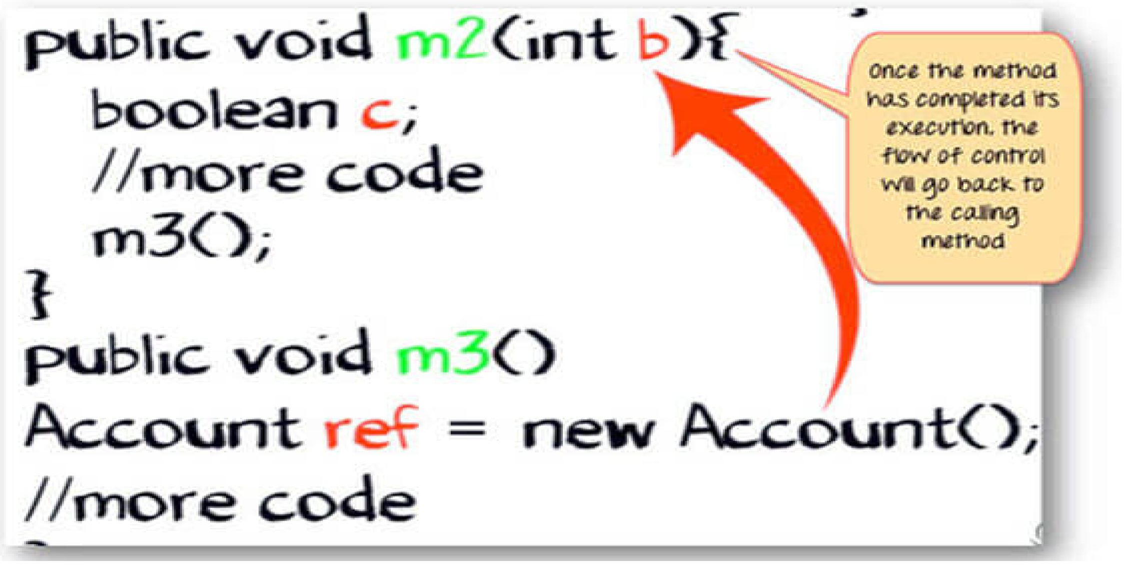
The reference variable "ref" will be created in a stack java.



The assignment "=" operator will make a reference variable to point to the object in the Heap.


Heap
p=0
q=0
the assign "=" operator will make ref variable to point to the object in a Heap
m3 ref

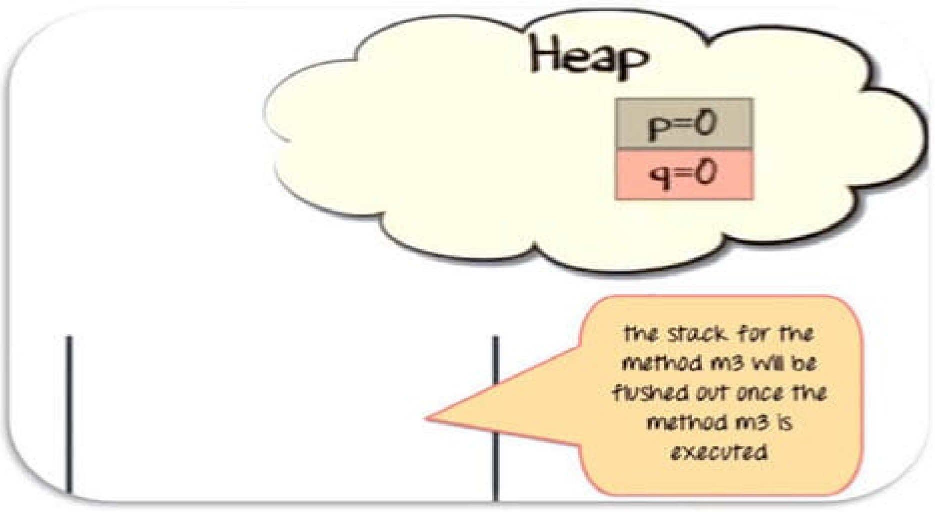
Once the method has completed its execution. The flow of control will go back to the calling method. Which in this case is method m2.



```
public void m2(int b){
   boolean c;
   //more code
   m3();
}
public void m3()
Account ref = new Account();
//more code
```

once the method has completed its execution. the flow of control will go back to the calling method

The stack from method m3 will be flushed out.


Heap
p=0
q=0
the stack for the method m3 will be flushed out once the method m3 is executed

Since the reference variable will no longer be pointing to the object in the heap, it would be eligible for garbage collection.

Eligible for garbage collection
Heap
p=0
q=0

- Once method m2 has completed its execution. It will be popped out of the stack, and all its variable will be flushed and no longer be available for use.
- Likewise for method m1.Eventually, the flow of control will return to the start point of the program. Which usually, is the “main” method.

Summary:

- When a method is called, a frame is created on the top of the stack.
- Once a method has completed execution, the flow of control returns to the calling method and its corresponding stack frame is flushed.
- Local variables are created in the stack.
- Instance variables are created in the heap & are part of the object they belong to.
- Reference variables are created in the stack.

Program No.8: Code, execute and debug programs that uses different control statements. Identify and resolve issues in the given code snippet.

a) Write a Program to check whether the given Integer is Odd or Even using if-else statement.

```
import java.util.Scanner;
class OddEven
      {
            public static void main(String[] args)
              {
                  int n;
                  Scanner s = new Scanner(System.in);
                  System.out.println("Enter the number you want to check:");
                  n = s.nextInt();
                  if(n % 2 == 0)
                    {
                          System.out.println("The given number "+n+" is Even ");
                    }
                  else
                  {
                          System.out.println("The given number "+n+" is Odd ");
                  }
            }
      }
```

Output:

```
Enter the number you want to check:12
The given number 12 is Even

Enter the number you want to check:7
The given number 7 is Odd
```

b) Write a program to illustrate switch statement

```
public class SwitchDemo
  {
        public static void main(String[] args)
         {
           int day = 4;
           switch (day)
             {
                 case 1:
                 System.out.println("Monday");
                 break;
                 case 2:
                 System.out.println("Tuesday");
                 break;
                 case 3:
                 System.out.println("Wednesday");
                 break;
                 case 4:
                 System.out.println("Thursday");
                 break;
```

```
                        case 5:
                        System.out.println("Friday");
                        break;
                        case 6:
                        System.out.println("Saturday");
                        break;
                        case 7:
                        System.out.println("Sunday");
                        break;
                }
            }
        }
```

Output:
Thursday

c) Write a Program to Generate n Fibonacci Numbers using for loop.

```
import java.util.Scanner;
class Fibonacci
    {
        public static void main(String[] args)
            {
                int n, a = 0, b = 0, c = 1;
                Scanner s = new Scanner(System.in);
                System.out.println("Enter value of n:");
                n = s.nextInt();
                System.out.print("Fibonacci Series:");
                for(int i = 1; i <= n; i++)
                  {
                        a = b;
                        b = c;
                        c = a + b;
                        System.out.print(a+" ");
                  }
            }
    }
```

Output:

```
Enter value of n:8
Fibonacci Series:0 1 1 2 3 5 8 13
```

d) Write a Program to Reverse a Number and Check if it is a Palindrome using while loop.

```
import java.util.Scanner;
class Palindrome
   {
     public static void main(String args[])
      {
         int n, m, a = 0,x;
         Scanner s = new Scanner(System.in);
         System.out.println("Enter any number:");
         n = s.nextInt();
         m = n;
```

```
            while(n > 0)
            {
                    x = n % 10;
                    a = a * 10 + x;
                    n = n / 10;
            }
            if(a == m)
            {
                System.out.println("Given number "+m+" is Palindrome");
            }
            else
            {
                 System.out.println("Given number "+m+" is Not Palindrome");
            }
        }
    }
```

Output:

```
Enter any number:565
Given number 565 is Palindrome

Enter any number:234
Given number 234 is Not Palindrome
```

e) Write a Program to Reverse a Number and find the Sum of its Digits using do-while Loop.

```
import java.util.Scanner;
class DoWhile
  {
     public static void main(String[] args)
       {
        int n, a, m = 0, sum = 0;
        Scanner s = new Scanner(System.in);
        System.out.print("Enter any number:");
        n = s.nextInt();
        do
          {
             a = n % 10;
             m = m * 10 + a;
            sum = sum + a;
            n = n / 10;
          }
        while( n > 0);
        System.out.println("Reverse:"+m);
        System.out.println("Sum of digits:"+sum);
     }
  }
```

Output:

```
Enter any number:35
Reverse:53
Sum of digits:8
```

f) Write a Program to Check whether the given Number is Prime Number. (Use break statement)

```
import java.util.Scanner;
class CheckPrime
  {
    public static void main(String args[])
      {
         int j, x, flag = 1;
         System.out.print("Enter any number :");
         Scanner s = new Scanner(System.in);
         x = s.nextInt();
         for( j = 2; j < x; j++)
           {
             if(x % j == 0)
               {
                   flag = 0;
                   break;
               }
           }
         if(flag == 1)
          {
              System.out.println("The "+x+" is a prime number.");
          }
         else
         {
             System.out.println("The "+x+" is not a prime number.");
         }
      }
  }
```

Output:

```
Enter any number :45
The 45 is not a prime number.

Enter any number :23
The 23 is a prime number.
```

Program No.9: Code, execute and debug programs that uses encapsulation concept.

```
class Encapsulate

  {
     // private variables declared these can only be accessed by public methods of class
      private String geekName;
      private int geekRoll;
      private int geekAge;

    public void setAge(int newAge) // set method for age to access private variable geekage
         {
           geekAge = newAge;
          }
   public void setName(String newName)   // set method for name to access private variable geekName
         {
           geekName = newName;
         }
    public void setRoll(int newRoll)  // set method for roll to access private variable geekRoll
           {
               geekRoll = newRoll;
           }
    public String getName()  // get method for name to access private variable geekName
          {
            return geekName;
           }

    public int getRoll()        // get method for roll to access private variable geekRoll
          {
           return geekRoll;
          }
      public int getAge()         // get method for age to access private variable geekAge
           {
            return geekAge;
           }

    }

public class EncapsulationDemo
     {
      public static void main(String[] args)
         {
             Encapsulate obj = new Encapsulate(); // Creating object
             // setting values of the variables using set methods
```

```
                obj.setName("Harish");
                obj.setAge(19);
                obj.setRoll(51);
                // Displaying values of the variables using get methods
                System.out.println("Geek's name: " + obj.getName());
                System.out.println("Geek's age: " + obj.getAge());
                System.out.println("Geek's roll: " + obj.getRoll());

                // Direct access of geekRoll is not possible due to encapsulation

                // System.out.println("Geek's roll: " + obj.geekName);  // Not possible

        }
    }
```

Output:

```
Geek's name: Harish
Geek's age: 19
Geek's roll: 51
```

Program No.10: Define class & implement like simple calculator and check compliance with SRP.

```
import java.util.Scanner;
public class BasicCalculator
{
  public static void main(String[] args)
    {
     double num1;
     double num2;
     double ans;
     char op;
     Scanner reader = new Scanner(System.in);
     System.out.print("Enter two numbers: ");
     num1 = reader.nextDouble();
     num2 = reader.nextDouble();
     System.out.print("\nEnter an operator (+, -, *, /): ");
     op = reader.next().charAt(0);
     switch(op) {
       case '+': ans = num1 + num2;
         break;
       case '-': ans = num1 - num2;
         break;
       case '*': ans = num1 * num2;
         break;
       case '/': ans = num1 / num2;
         break;
      default: System.out.printf("Error! Enter correct operator");
       return;
     }
     System.out.print("\nThe result is given as follows:\n");
     System.out.printf(num1 + " " + op + " " + num2 + " = " + ans);
  }
}
```

Output:

```
Enter two numbers:
12
78
Enter an operator (+, -, *, /): *

The result is given as follows:
12.0 * 78.0 = 936.0
```

Program No.11: Code, execute and debug programs that uses array concept.

a) Java Program to illustrate how to declare, instantiate, initialize and traverse the Java array.

```
class OneDimArray
  {
    public static void main(String args[])
      {
        int a[]=new int[5];//declaration and instantiation
        a[0]=10;//initialization
        a[1]=20;
        a[2]=70;
        a[3]=40;
        a[4]=50;
        //traversing array
        System.out.println("Elements of array are");
        for(int i=0;i<a.length;i++) //length is the property of array
        System.out.println(a[i]);
   }
}
```

Output:

```
Elements of array are
10
20
70
40
50
```

b) Java Program to illustrate the use of multidimensional array

```
class MultiDimArray
 {
  public static void main(String args[])
   {
   int arr[][]={{1,2,3},{2,4,5},{4,4,5}};  //declaring and initializing 2D array
   //printing 2D array
   System.out.println("Elements of 2D array are");
   for(int i=0;i<3;i++)
   {
    for(int j=0;j<3;j++)
      {
        System.out.println(arr[i][j]+" ");
      }
   System.out.println();
   }
```

```
        }
  }
```

Output:

```
Elements of 2D array are
1
2
3

2
4
5

4
4
5
```

Program No.12: Code, execute and debug programs to perform string manipulation.

```
import java.lang.String;
class StringDemo
{
        public static void main(String arg[])
        {
           String s1 = new String("gpt athani");
           String s2 = "GPT ATHANI";
           System.out.println("The string s1 is : " + s1);
           System.out.println("The string s2 is : " + s2);
           System.out.println("Length of the string s1 is : " + s1.length());
           System.out.println("Length of the string s2 is : " + s2.length());
           System.out.println("The String s1 in Upper Case : " + s1.toUpperCase());
           System.out.println("The String s2 in Lower Case : " + s2.toLowerCase());
           System.out.println("The first occurrence of a is at the position : "+ s1.indexOf('a'));
           System.out.println("s1 equals to s2 : " + s1.equals(s2));
          System.out.println("s1 equals ignore case to s2 : " + s1.equalsIgnoreCase(s2));
           System.out.println("Character at an index of 6 is :" + s1.charAt(6));
           String s3 = s1.substring(4, 8);
           System.out.println("Extracted substring is :" + s3);
           System.out.println("After Replacing a with b in s1 : "+ s1.replace('a', 'b'));
           System.out.println("After string concat :" + s1.concat(" Karnataka"));
           String s4 = "  This is a book "; //White space before This word.
           System.out.println("The string s4 is :" + s4);
           System.out.println("After string trim :" + s4.trim());
           int result = s1.compareTo(s2);
           System.out.println("After compareTo");
           if (result == 0)
           System.out.println(s1 + " is equal to " + s2);
           else if (result > 0)
                System.out.println(s1 + " is greater than " + s2);
           else
                System.out.println(s1 + " is smaller than " + s2);
        }
   }
```

Output:

```
The string s1 is : gpt athani
The string s2 is : GPT ATHANI
Length of the string s1 is : 10
Length of the string s2 is : 10
The String s1 in Upper Case : GPT ATHANI
The String s2 in Lower Case : gpt athani
The first occurrence of a is at the position : 4
s1 equals to s2 : false
s1 equals ignore case to s2 : true
Character at an index of 6 is :h
Extracted substring is :atha
After Replacing a with b in s1 : gpt bthbni
After string concat :gpt athani Karnataka
The string s4 is :  This is a book
After string trim :This is a book
After compareTo
gpt athani is greater than GPT ATHANI
```

Program No.13: Code, execute and debug a program that implements the concept of inheritance.

```
    class Room
    {
            int length,breadth;
            Room(int x, int y)
            {
                length = x;
               breadth = y;
            }
            int area()
            {
               return (length * breadth);
            }
    }
   class ClassRoom extends Room
    {
         int height;
       ClassRoom(int x, int y, int z)
        {
           super(x, y);
           height = z;
        }
    int volume()
    {
       return (length * breadth * height);
    }
  }
 class SubClass
    {
    public static void main(String args[])
      {
            ClassRoom cr = new ClassRoom(20, 30, 10);
            int area = cr.area();
            int volume =cr.volume();

            System.out.println("Area=" + area);
            System.out.println("Volume=" + volume);
        }
    }
```

Output

```
Area = 600
Volume = 6000
```

Program No.14: Design a class & implement like file parser and check compliance with OCP.

```
class Cuboid
 {
    public double length;
    public double breadth;
    public double height;
}
 class Application
 {
  public double get_total_volume(Cuboid geo_objects[])
    {

        double vol_sum = 0;
       for (Cuboid geo_obj : geo_objects)
         {
           vol_sum += geo_obj.length * geo_obj.breadth * geo_obj.height;
         }
        return vol_sum;
    }
}

public class OCP
{
    public static void main(String args[])
    {
        Cuboid cb1 = new Cuboid();
        cb1.length = 5;
        cb1.breadth = 10;
        cb1.height = 15;

        Cuboid cb2 = new Cuboid();
        cb2.length = 2;
        cb2.breadth = 4;
        cb2.height = 6;

        Cuboid cb3 = new Cuboid();
        cb3.length = 3;
        cb3.breadth = 12;
        cb3.height = 15;

        Cuboid  c_arr[] = new Cuboid[3];
        c_arr[0] = cb1;
        c_arr[1] = cb2;
        c_arr[2] = cb3;

        Application app = new Application ();
        double volume = app.get_total_volume(c_arr);
        System.out.println ("The total volume is " + volume);
    }
}
```

Output:

```
The total volume is 1338.0
```

Program No.15: Code, execute and debug programs that uses
a. static binding
b. dynamic binding

a) Static binding

```
class Dog
{
    private void eat()
      {
        System.out.println("Dog is eating...");
      }
public static void main(String args[])
   {
      Dog d1=new Dog();
     d1.eat();
   }
}
```

Output:

```
Dog is eating...
```

b) Dynamic binding

```
class Animal
  {
     void eat()
     {
       System.out.println("animal is eating...");
     }
  }
class Dog1 extends Animal
   {
        void eat()
         {
           System.out.println("dog is eating...");
         }
        public static void main(String args[])
         {
           Animal a=new Dog1();
           a.eat();
         }
   }
```

Output:

```
Dog is eating...
```

Program No.16: Code, execute and debug program that uses abstract class to achieve abstraction.

```
abstract class Shape
{
   abstract void draw();
}
//In real scenario, implementation is provided by others i.e. unknown by end user
class Rectangle extends Shape
  {
     void draw()
        {
          System.out.println("drawing rectangle");
        }
   }
class Circle extends Shape
 {
     void draw()
        {
         System.out.println("drawing circle");
        }
}
//In real scenario, method is called by programmer or user
class TestAbstraction
    {
       public static void main(String args[])
         {
           Shape s=new Circle();
                //In a real scenario, object is provided through method, e.g., getShape() method
           s.draw();
         }
   }
```

Output: drawing circle

Program No.17: Code, execute and debug program that uses interface to achieve abstraction.

```
interface Area
{
        final static float pi = 3.142F;
        float compute(float x, float y);
}

class Rectangle implements Area
{
        public float compute(float x, float y)
        {
                return ( x * y);
        }
}
class Circle implements Area

{
        public float compute(float x, float y)
        {
                return (pi * x * x);
        }
}

class InterfaceTest
{
        public static void main(String args[])
        {
                Rectangle rect = new Rectangle();
                Circle cir = new Circle();
                Area area;
                area= rect;
                System.out.println("Area of Rectangle = " + area.compute(10, 20));
                 area = cir;
                System.out.println("Area of Circle = " + area.compute(30, 0));

        }
}
```

Output:

```
Area of Rectangle = 200
Area of Circle =3070.8
```

Program No.18: Code, execute and debug program to read the content of the file and write the content to another file.

(First create one text file- inputFile.txt and another text file outputFile.txt in C:drive\test folder)

```
 import java.io.File;
 import java.io.FileInputStream;
 import java.io.FileWriter;
 import java.io.IOException;
 import java.util.Scanner;
class CopyContent
 {
   public static void main(String[] args) throws IOException
   {
      File file = new File("C:\\test\\inputFile.txt");
      FileInputStream inputStream = new FileInputStream(file);
      Scanner sc = new Scanner(inputStream);
      StringBuffer buffer = new StringBuffer();
       while(sc.hasNext())
        {
          buffer.append(" "+sc.nextLine());
        }
     System.out.println("Contents of the file: "+buffer);
     File dest = new File("C:\\test\\outputFile.txt");
     FileWriter writer = new FileWriter(dest);
     writer.write(buffer.toString());
     writer.close();
     System.out.println("File copied successfully.......");
   }
 }
```

Output:

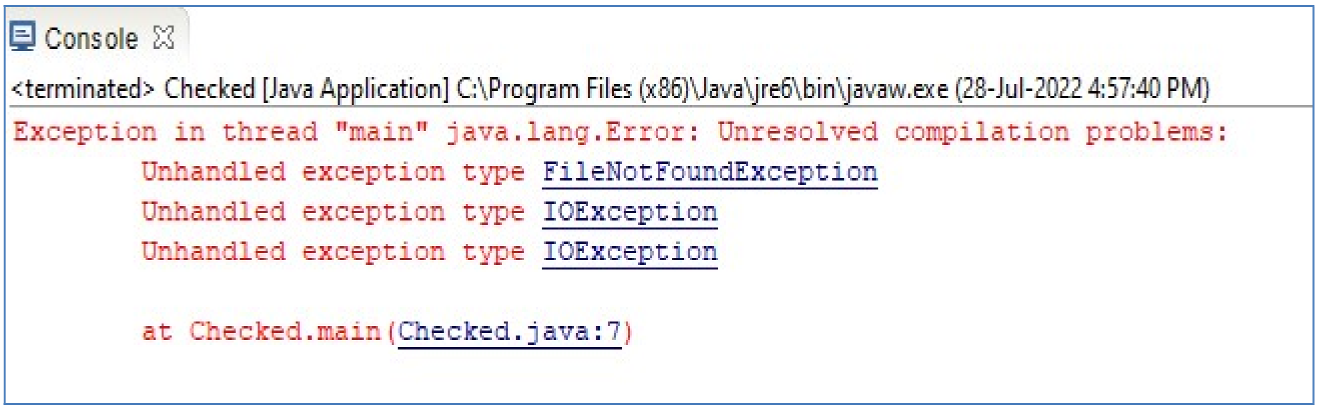
```
Contents of the file:  Welcome to GPT Athani This is example for checked exceptions. It uses throws keyword. Welcome to CS dept
File copied successfully.......
```

Program No.19: Code, execute and debug program that handles checked and unchecked exceptions

a) Checked Exceptions:

```
import java.io.*;
 class Checked
{
   public static void main(String[] args)
   {
      FileReader file = new FileReader("C:\\test\\gpta.txt");
      BufferedReader fileInput = new BufferedReader(file);
      for (int counter = 0; counter < 3; counter++)
      System.out.println(fileInput.readLine());
      fileInput.close();
   }
}
```

Output:



```
Console
<terminated> Checked [Java Application] C:\Program Files (x86)\Java\jre6\bin\javaw.exe (28-Jul-2022 4:57:40 PM)
Exception in thread "main" java.lang.Error: Unresolved compilation problems:
	Unhandled exception type FileNotFoundException
	Unhandled exception type IOException
	Unhandled exception type IOException

	at Checked.main(Checked.java:7)
```

- To fix the above program, we either need to specify a list of exceptions using **throws**, or we need to use a **try-catch block.** We have used throws in the below program. Since *FileNotFoundException* is a subclass of *IOException*, we can just specify *IOException* in the throws list and make the above program compiler-error-free.

```
import java.io.*;
 class Checked
{
   public static void main(String[] args) throws IOException
   {
       FileReader file = new FileReader("C:\\test\\gpta.txt");
       BufferedReader fileInput = new BufferedReader(file);
       for (int counter = 0; counter < 3; counter++)
       System.out.println(fileInput.readLine());
      fileInput.close();
   }
}
```

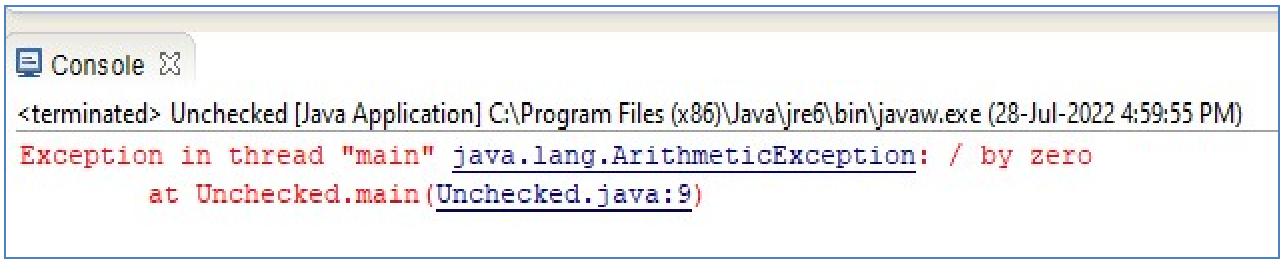
Output:

```
Welcome to GPT Athani
This is example for checked exceptions.
It uses throws keyword.
```

a) **Unchecked Exceptions:**

```
class Unchecked
 {
   public static void main(String args[])
   {
      // Here we are dividing by 0  which will not be caught at compile time
      // as there is no mistake but caught at runtime because it is mathematically incorrect
      int x = 0;
      int y = 10;
      int z = y / x;
   }
}
```

Output:



```
<terminated> Unchecked [Java Application] C:\Program Files (x86)\Java\jre6\bin\javaw.exe (28-Jul-2022 4:59:55 PM)
Exception in thread "main" java.lang.ArithmeticException: / by zero
	at Unchecked.main(Unchecked.java:9)
```

Program No.20: Code, execute and debug program to illustrate throwing our own exceptions or user defined exceptions.

```
import java.lang.Exception;
class MyException extends Exception
   {
         MyException(String message)
         {
           super(message);
         }
   }
class TestMyException
   {
   public static void main(String args[])
       {
         int x=5,y=1000;
         try
         {
            float z=(float) x/(float) y;
            if(z < 0.01)
              {
                  throw new MyException("Number is too small");
              }
         }
      catch(MyException e)
         {
            System.out.println("Caught my exception");
            System.out.println(e.getMessage());
         }
       finally
         {
            System.out.println("I am always here");
         }
      }
  }
```

Output:

```
Caught my exception
Number is too small
I am always here
```

Program No.21: Design an interface & implement it like one that builds different types of toys and check compliance with ISP.

```
        interface Toy
          {
                void setPrice(double price);
                void setColor(String color);
          }
        interface Movable
          {
                void move();
         }
        interface Flyable
         {
                void fly();
          }
        class ToyHouse implements Toy
          {
                double price;
                String color;
                @Override
                public void setPrice(double price)
                 {
                     this.price = price;
                 }
                @Override
                public void setColor(String color)
                {
                     this.color=color;
                 }
                @Override
                public String toString()
                {
                   return "ToyHouse: Toy house- Price: "+price+" Color: "+color;
                }
        }

class ToyCar implements Toy, Movable
    {
        double price;
        String color;
        @Override
        public void setPrice(double price)
        {
                this.price = price;
        }
        @Override
        public void setColor(String color)
        {
                this.color=color;
        }
        @Override
        public void move()
        {
              System.out.println("ToyCar: Start moving car.");
        }
        @Override
        public String toString()
         {
            return "ToyCar: Moveable Toy car- Price: "+price+" Color: "+color;
         }
   }

class ToyPlane implements Toy, Movable, Flyable
{
        double price;
        String color;
        @Override
        public void setPrice(double price)
        {
              this.price = price;
         }
        @Override
        public void setColor(String color)
        {
           this.color=color;
        }
        @Override
        public void move()
        {
                 System.out.println("ToyPlane: Start moving plane.");
        }
        @Override
        public void fly()
        {
                 System.out.println("ToyPlane: Start flying plane.");
        }
        @Override
        public String toString()
        {
                 return ("ToyPlane: Moveable and flyable toy plane- Price: "+price+"Color: "+color);
        }
   }

class ToyBuilder
  {
     public static ToyHouse buildToyHouse()
        {
        ToyHouse toyHouse=new ToyHouse();
        toyHouse.setPrice(15.00);
        toyHouse.setColor("green");
```

```
            return toyHouse;
          }
    public static ToyCar buildToyCar()
        {
            ToyCar toyCar=new ToyCar();
            toyCar.setPrice(25.00);
            toyCar.setColor("red");
            toyCar.move();
            return toyCar;
        }
public static ToyPlane buildToyPlane()
      {
            ToyPlane toyPlane=new ToyPlane();
            toyPlane.setPrice(125.00);
            toyPlane.setColor("white");
            toyPlane.move();
            toyPlane.fly();
            return toyPlane;
      }
}

public class ToyISPTest
 {
      public static void main(String[] args)
         {
            // TODO Auto-generated method stub
            ToyHouse toyHouse=ToyBuilder.buildToyHouse();
            System.out.println(toyHouse);
            ToyCar toyCar=ToyBuilder.buildToyCar();;
            System.out.println(toyCar);
            ToyPlane toyPlane=ToyBuilder.buildToyPlane();
            System.out.println(toyPlane);
       }
}
```

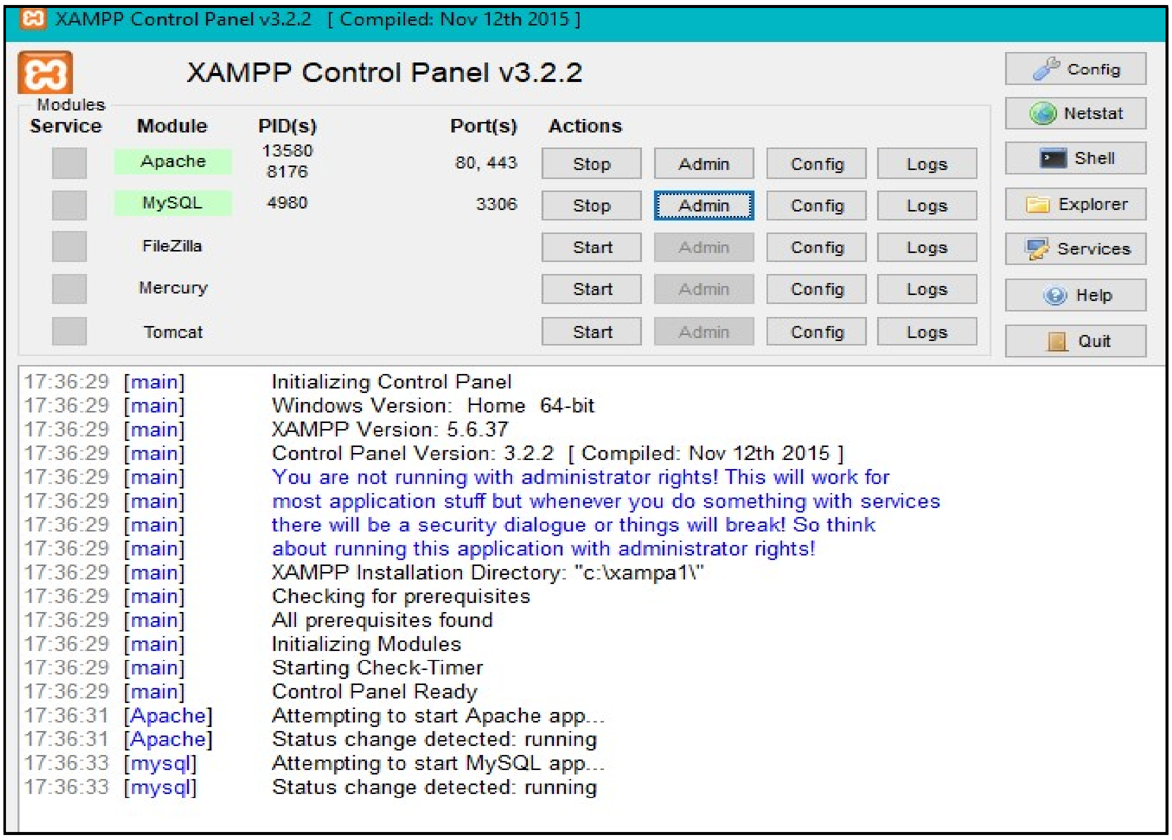
Output:

```
ToyHouse: Toy house- Price: 15.0 Color: green
ToyCar: Start moving car.
ToyCar: Moveable Toy car- Price: 25.0 Color: red
ToyPlane: Start moving plane.
ToyPlane: Start flying plane.
ToyPlane: Moveable and flyable toy plane- Price: 125.0 Color: white
```

Program No.22: Code, execute and debug programs to connect to database through JDBC and perform basic DB operations.

Step 1: In addition to JDK and Eclipse environment, install Xampp software for Apache server and MySql service.

Step 2: Now open Xampp control panel to start Apache and MySql services as shown below. Then click on MySql-Admin button to open MySql http://localhost/phpmyadmin/ in brower.

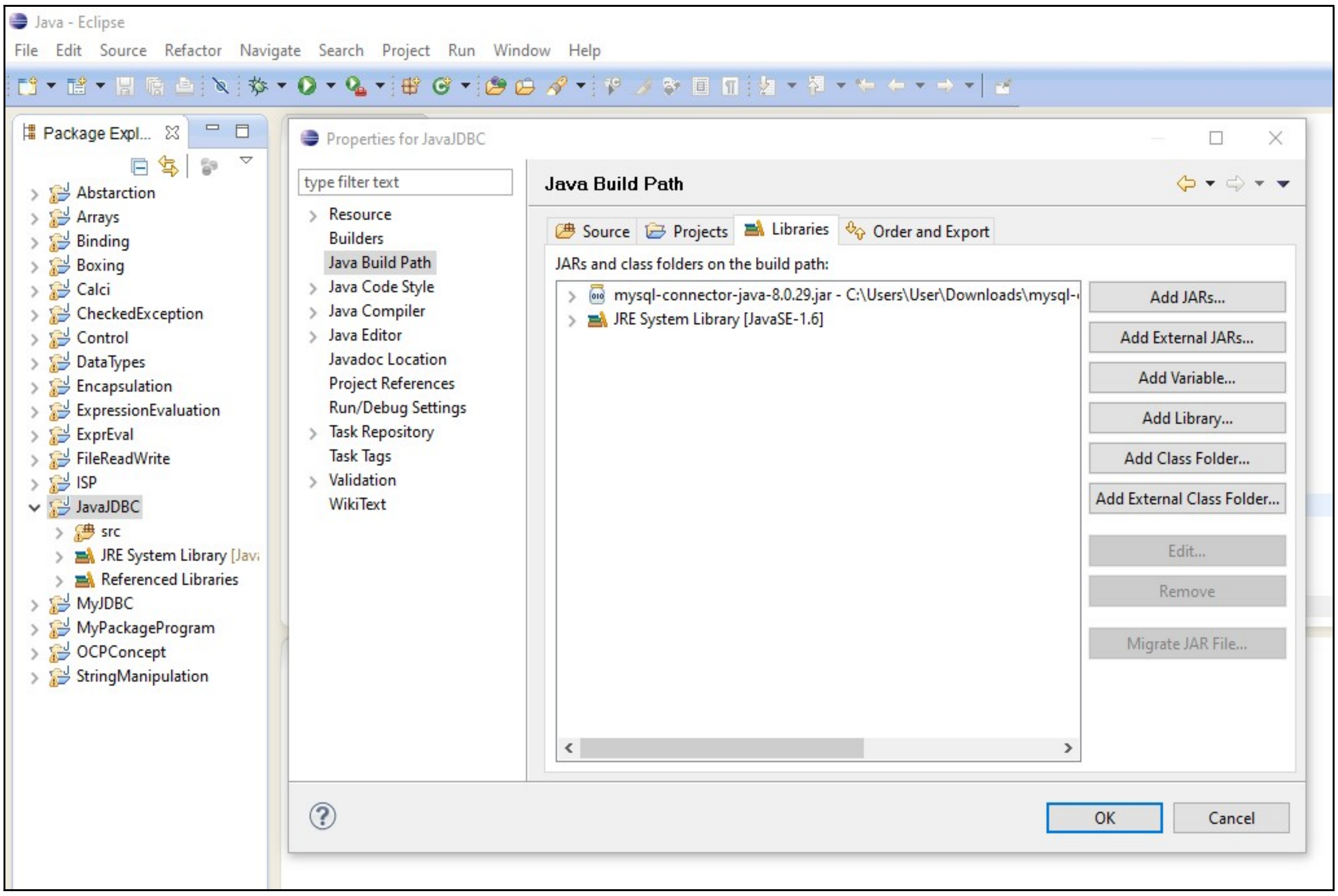

XAMPP Control Panel v3.2.2 [Compiled: Nov 12th 2015]
XAMPP Control Panel v3.2.2
Config
Netstat
Shell
Explorer
Services
Help
Quit

Service	Module	PID(s)	Port(s)	Actions			
	Apache	13580 8176	80, 443	Stop	Admin	Config	Logs
	MySQL	4980	3306	Stop	Admin	Config	Logs
	FileZilla			Start	Admin	Config	Logs
	Mercury			Start	Admin	Config	Logs
	Tomcat			Start	Admin	Config	Logs

```
17:36:29  [main]      Initializing Control Panel
17:36:29  [main]      Windows Version:  Home  64-bit
17:36:29  [main]      XAMPP Version: 5.6.37
17:36:29  [main]      Control Panel Version: 3.2.2  [ Compiled: Nov 12th 2015 ]
17:36:29  [main]      You are not running with administrator rights! This will work for
17:36:29  [main]      most application stuff but whenever you do something with services
17:36:29  [main]      there will be a security dialogue or things will break! So think
17:36:29  [main]      about running this application with administrator rights!
17:36:29  [main]      XAMPP Installation Directory: "c:\xampa1\"
17:36:29  [main]      Checking for prerequisites
17:36:29  [main]      All prerequisites found
17:36:29  [main]      Initializing Modules
17:36:29  [main]      Starting Check-Timer
17:36:29  [main]      Control Panel Ready
17:36:31  [Apache]    Attempting to start Apache app...
17:36:31  [Apache]    Status change detected: running
17:36:33  [mysql]     Attempting to start MySQL app...
17:36:33  [mysql]     Status change detected: running
```

Step 3: To connect MySql databse in Java using Eclipse, follow below steps.

- Open Eclipse IDE and create new Java project named JavaJDBC and click finish.
- Create a new Java class with DBTest and click on the finish button.
- In order to connect Java program (DBTest.java) with MySQL database, we need to download and include MySQL JDBC driver which is a JAR file, namely **mysql-connector-java-8.0.29.jar.**
- Now right click on JavaJDBC project to include connector and go to properties.
- Click on Java build path option-> click on libraries and then click on Add External JARS.
- Now select downloaded jar file **mysql-connector-java-8.0.29.jar.** & click open.
- Click on OK and close.



JARs and class folders on the build path:

mysql-connector-java-8.0.29.jar - C:\Users\User\Downloads\mysql-

JRE System Library [JavaSE-1.6]

Step 4: Now in browser go to myphpadmin page and create student table in test database with following fields as shown below and click save.

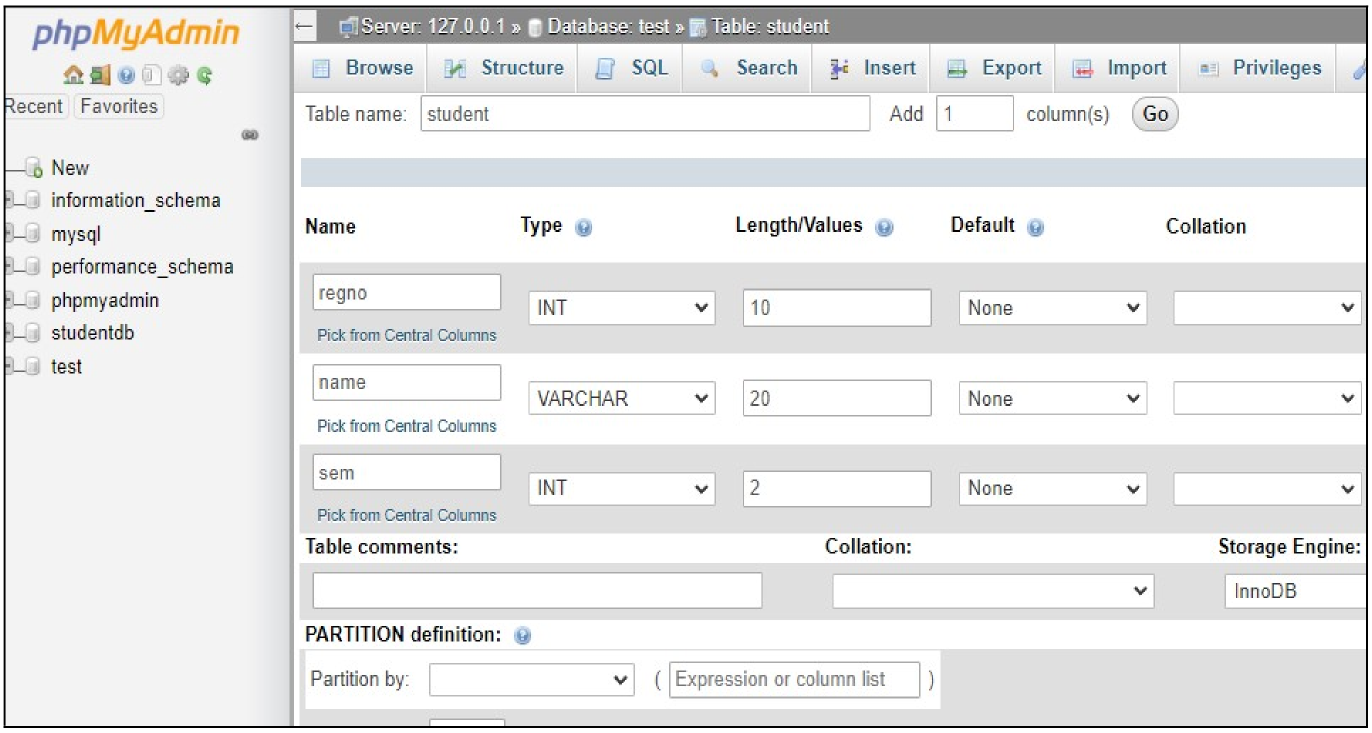


Table name: student Add 1 column(s)

Name	Type	Length/Values	Default	Collation
regno	INT	10	None	
name	VARCHAR	20	None	
sem	INT	2	None	

Table comments: Collation: Storage Engine: InnoDB

PARTITION definition:

Partition by: (Expression or column list)

Connecting Java Program with MySQL Database

- After adding jar file, connect the Java program with MySQL Database.

  i)Establish a connection using DriverManager.getConnection(String URL) and it returns a Connection reference.

  ii) In String URL parameter write like this :

  jdbc:mysql://localhost:3306/test", "root", "password"

Where,

- jdbc is the API.
- mysql is the database.
- localhost is the name of the server in which MySQL is running.
- 3306 is the port number.
- test is the database name. If the database name is different, then replace this name with the correct database name.
- root is the username of the MySQL database. It is the default username for the MySQL database.
- password is the password that is given while installing the MySQL database.

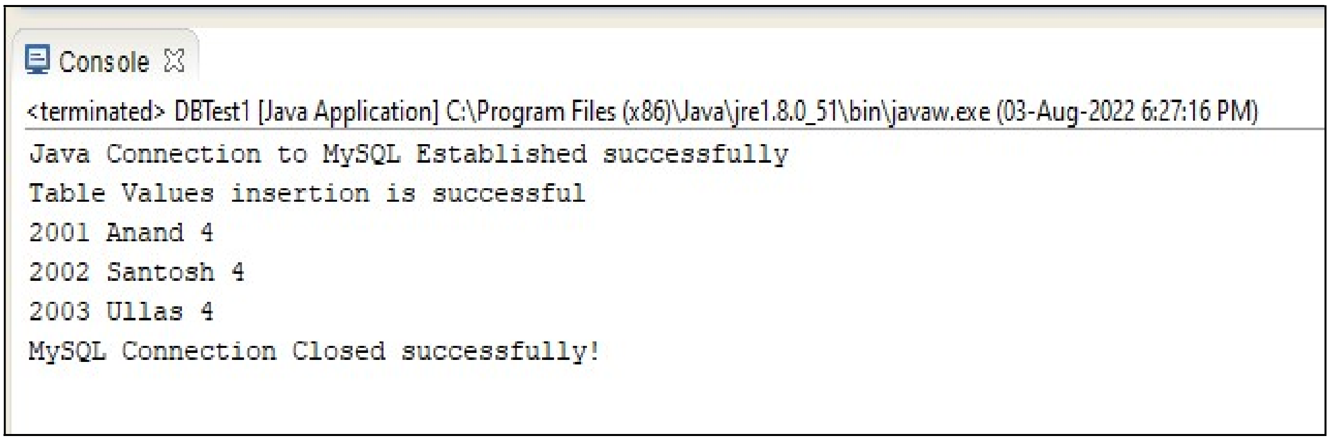
- SQL Exception might occur while connecting to the database, try-catch block must be used.

Step 5: Write below code in DBTest class Eclipse environment.

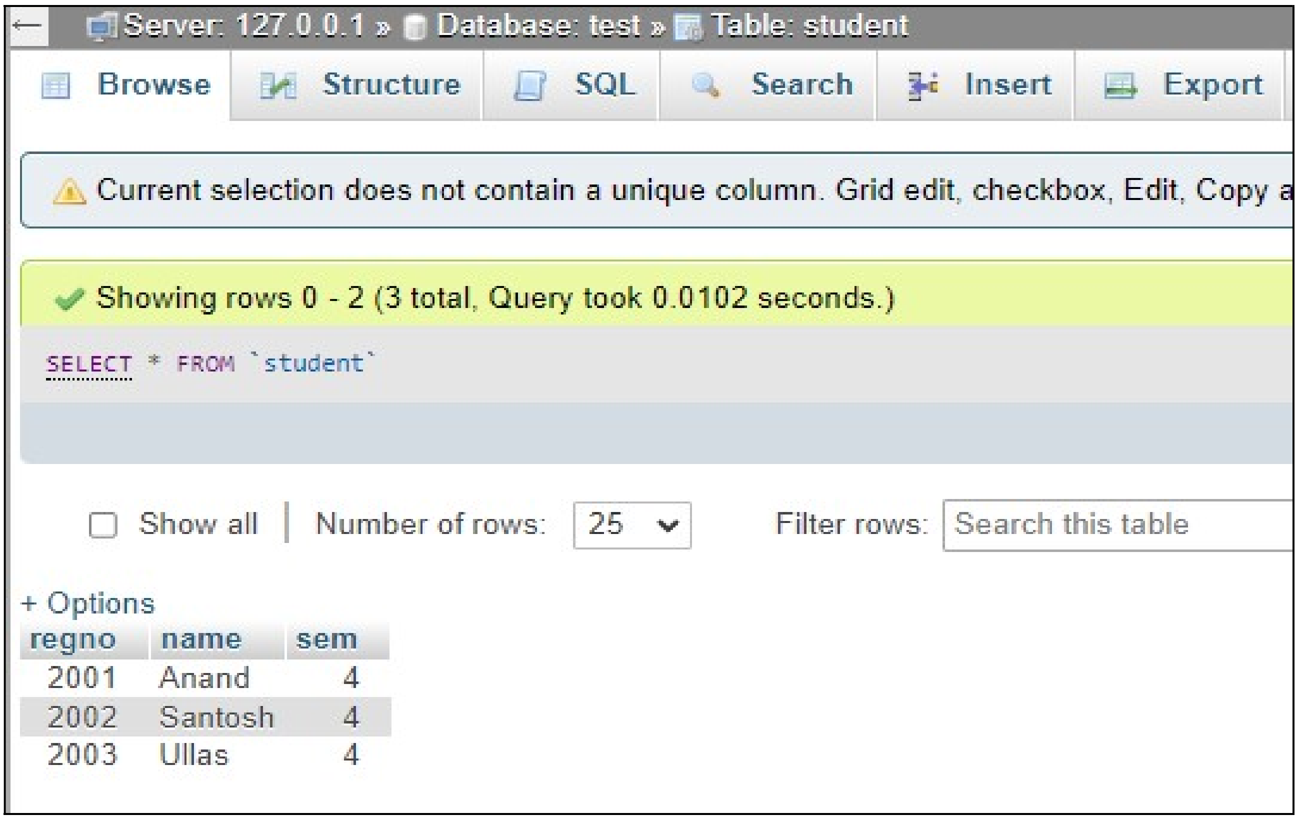
```
import java.sql.*;
public class DBTest
{
   public static void main(String[] args)
   {
   String url= "jdbc:mysql://localhost:3306/test"; // table URL
   String uname = "root"; // MySQL credentials
   String pw = "";
   try
   {
         //Loading MySQL Driver
         Class.forName("com.mysql.cj.jdbc.Driver");
         // Establishing connection with MySQL
         Connection con = DriverManager.getConnection(url,uname,pw);
         System.out.println("Java Connection to MySQL Established successfully");
         // Creating Statement object for query execution
         Statement st=con.createStatement();
         // Delete the table student if already present in the test database
         String deltbl= "DROP TABLE STUDENT";
         st.executeUpdate(deltbl);
         // Create a table STUDENT in database test
```

```
                String qrytbl= "CREATE TABLE STUDENT(regno int,name varchar(30),sem int)";
                st.executeUpdate(qrytbl);
                // Insert values into the STUDENT table
                String qry1="INSERT INTO STUDENT values(2001,'Anand',4)";
                st.executeUpdate(qry1);
                String qry2="INSERT INTO STUDENT values(2002,'Santosh',4)";
                st.executeUpdate(qry2);
                String qry3="INSERT INTO STUDENT values(2003,'Ullas',4)";
                st.executeUpdate(qry3);
                System.out.println("Table Values insertion is successful");
                // Query to retrieve values from table
                String query= "SELECT * FROM STUDENT";
                ResultSet rs = st.executeQuery(query);//Execute query
                while (rs.next())
                {
                        //Retrieve row-wise values of regno, name and sem columns
                        int regno = rs.getInt("regno");
                        String name= rs.getString("name");
                        int sem=rs.getInt("sem");
                        // Display the result on console
                        System.out.println(regno + " " + name+ " "+ sem);
                }
                st.close(); // close statement
                con.close(); // close connection
                System.out.println("MySQL Connection Closed successfully!");
            }
      catch(Exception e)
        {
          System.out.println("Error while executing program:" + e);
        }
      }
    }
```

Output:



```
<terminated> DBTest1 [Java Application] C:\Program Files (x86)\Java\jre1.8.0_51\bin\javaw.exe (03-Aug-2022 6:27:16 PM)
Java Connection to MySQL Established successfully
Table Values insertion is successful
2001 Anand 4
2002 Santosh 4
2003 Ullas 4
MySQL Connection Closed successfully!
```



Current selection does not contain a unique column. Grid edit, checkbox, Edit, Copy a

Showing rows 0 - 2 (3 total, Query took 0.0102 seconds.)

```
SELECT * FROM `student`
```

Show all | Number of rows: 25 Filter rows: Search this table

+ Options

regno	name	sem
2001	Anand	4
2002	Santosh	4
2003	Ullas	4