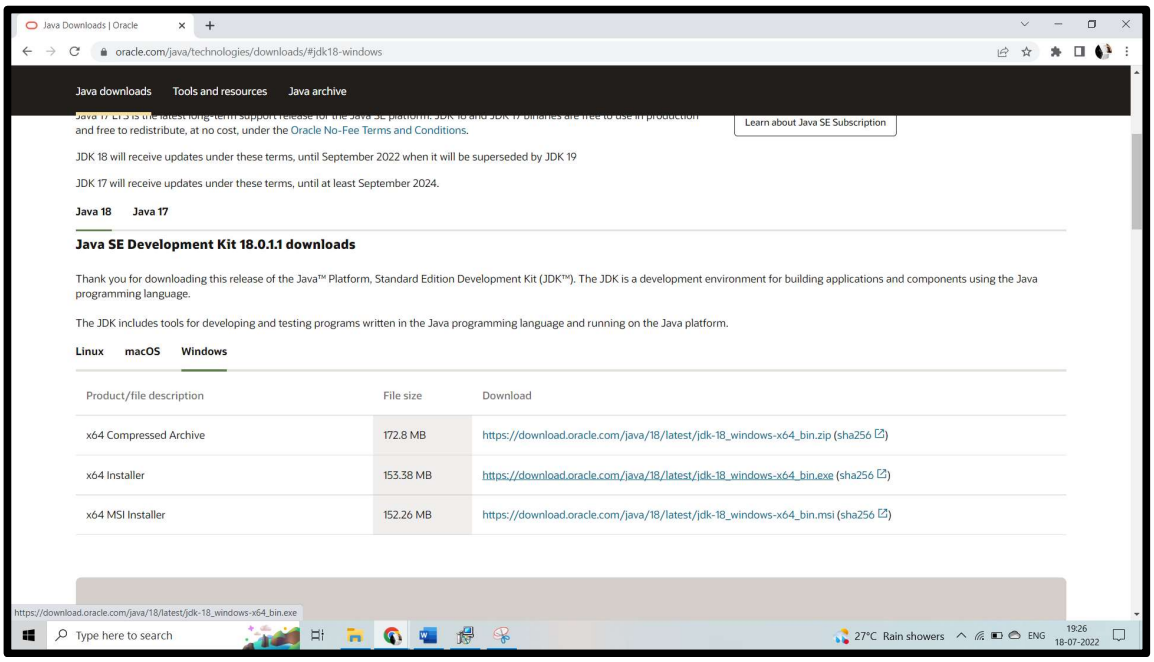
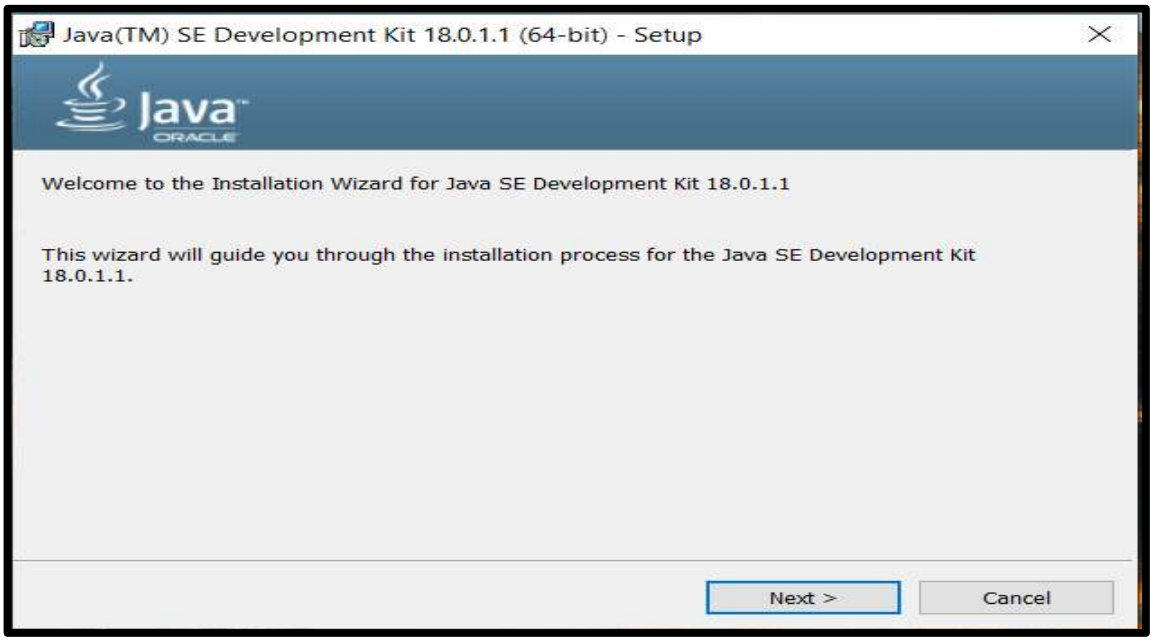
Practical: - 1

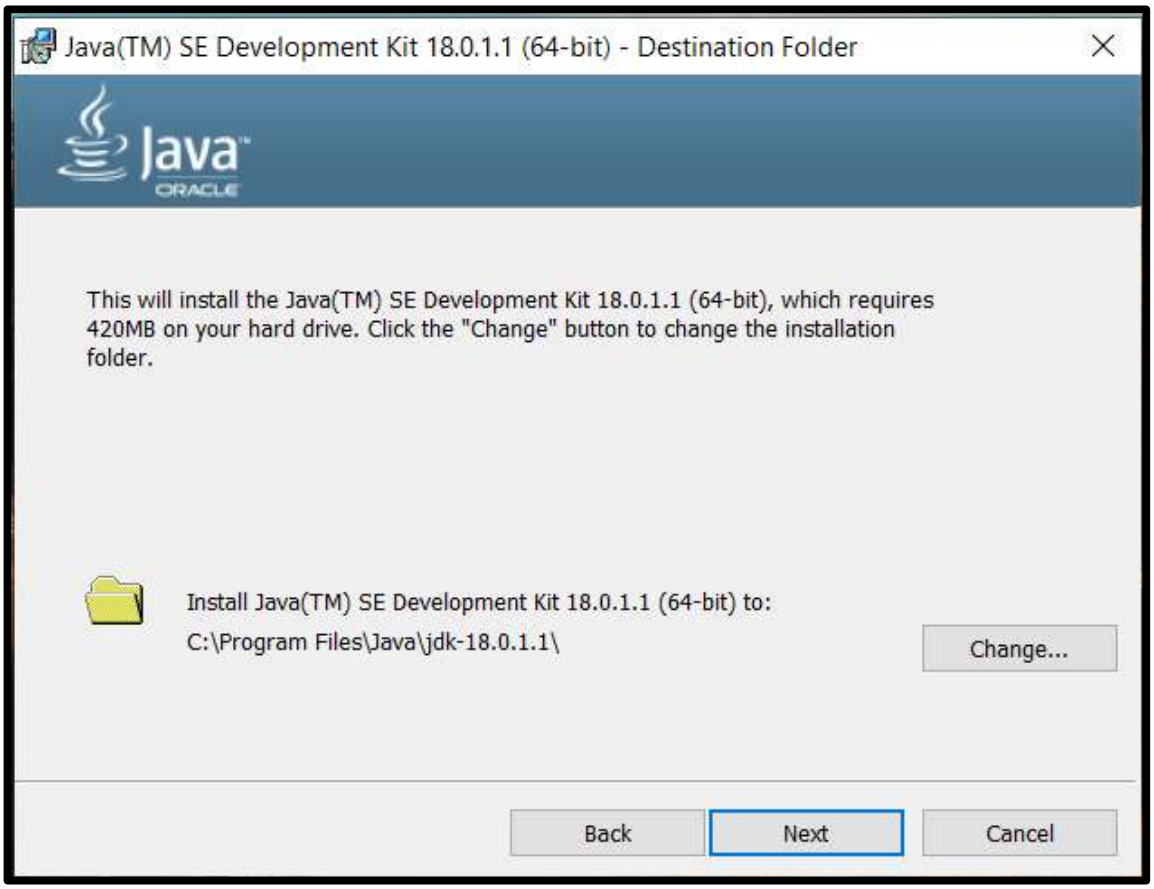
Aim: - Install JDK and write a java program to print your name.

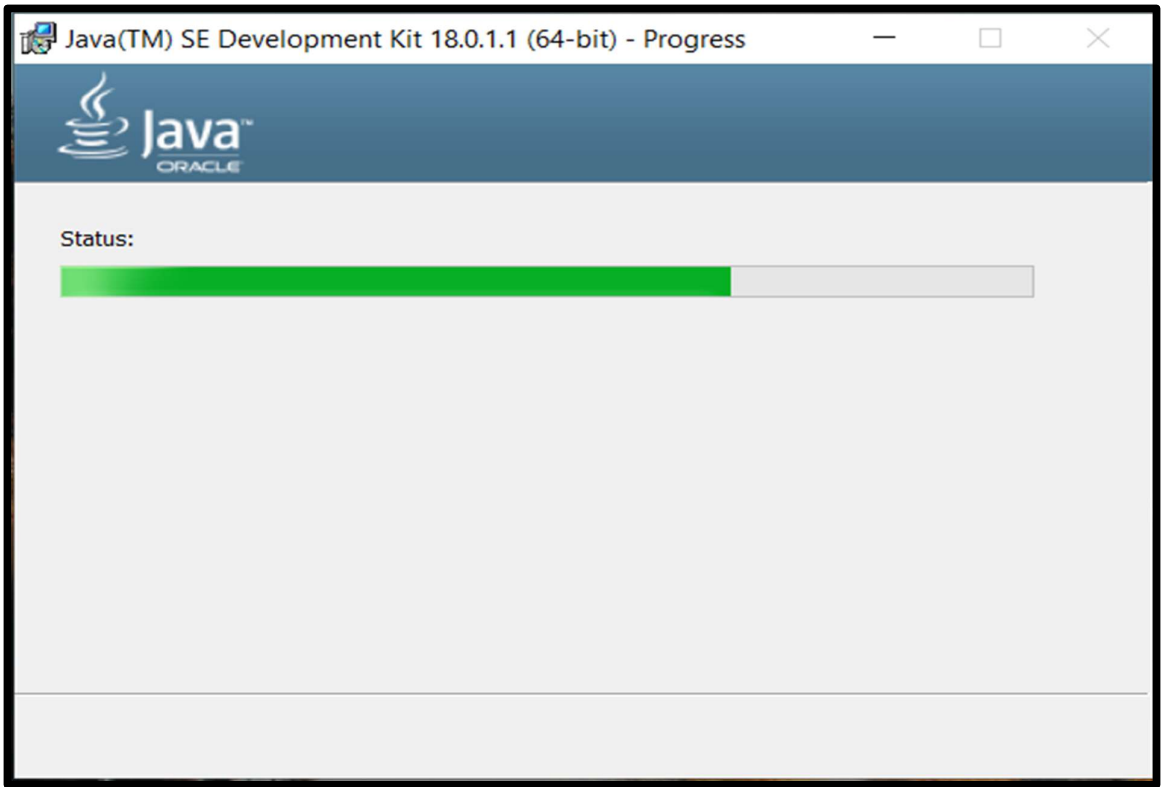
Step 1: - Click here to download JDK Java 18 on windows.

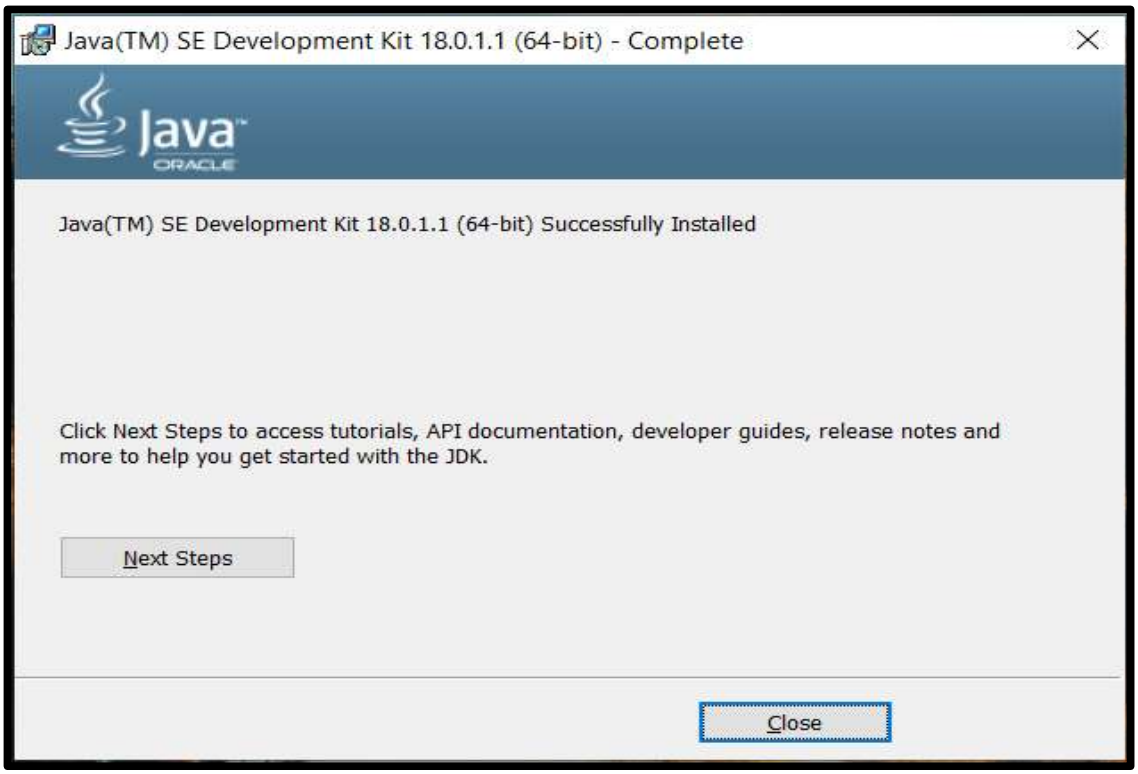
X64 Installer



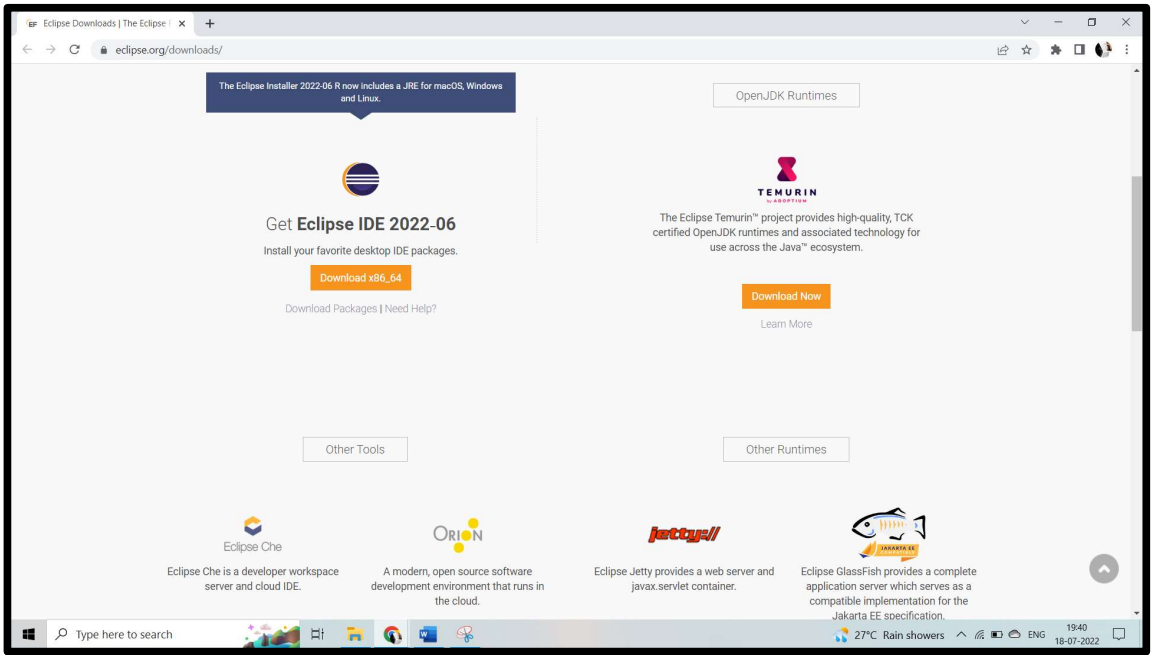


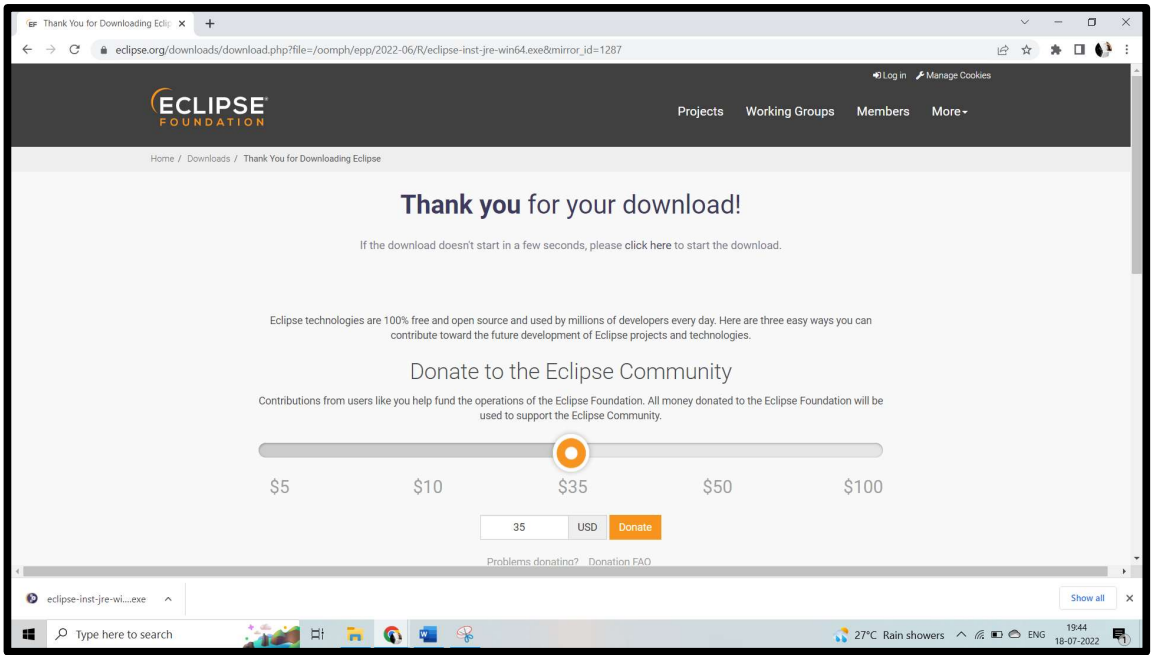

Java(TM) SE Development Kit 18.0.1.1 (64-bit) - Destination Folder
Java
ORACLE
This will install the Java(TM) SE Development Kit 18.0.1.1 (64-bit), which requires 420MB on your hard drive. Click the "Change" button to change the installation folder.
Install Java(TM) SE Development Kit 18.0.1.1 (64-bit) to:
C:\Program Files\Java\jdk-18.0.1.1\
Change...
Back
Next
Cancel


Java(TM) SE Development Kit 18.0.1.1 (64-bit) - Progress
Java
ORACLE
Status:



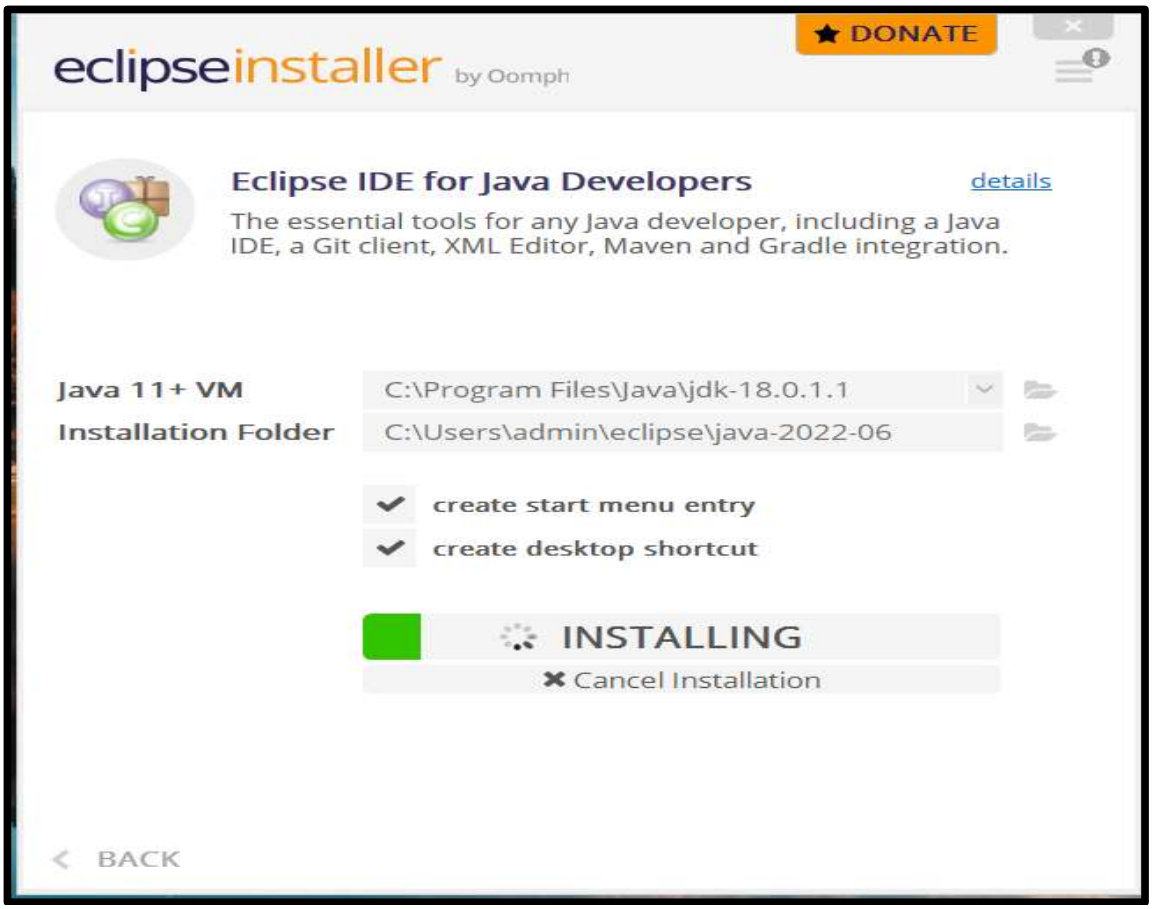
Step 2: - Click here to download Eclipse IDE on windows.



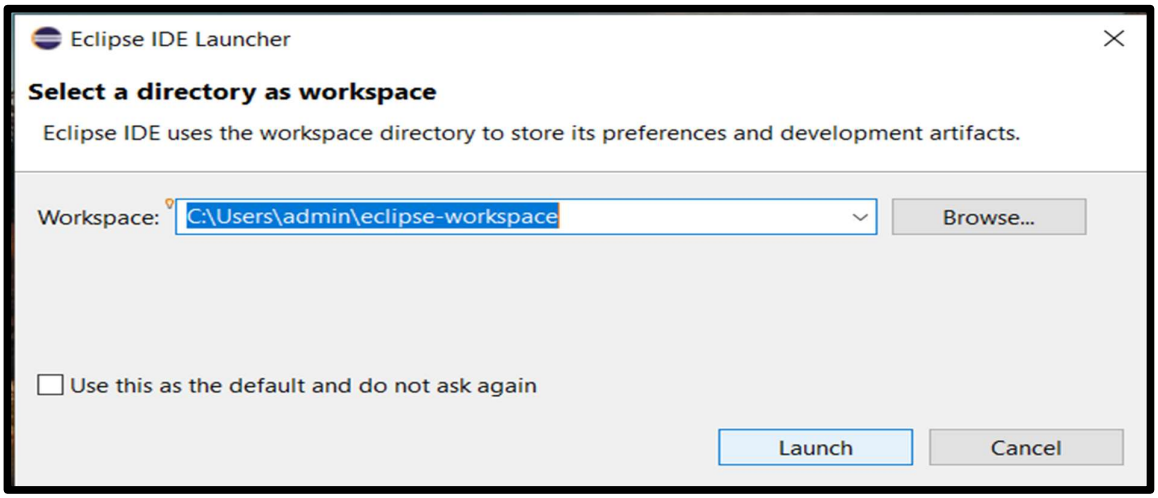

Thank You for Downloading Eclip
eclipse.org/downloads/download.php?file=/oomph/epp/2022-06/R/eclipse-inst-jre-win64.exe&mirror_id=1287
Log in
Manage Cookies
ECLIPSE FOUNDATION
Projects
Working Groups
Members
More
Home / Downloads / Thank You for Downloading Eclipse
Thank you for your download!
If the download doesn't start in a few seconds, please click here to start the download.
Eclipse technologies are 100% free and open source and used by millions of developers every day. Here are three easy ways you can contribute toward the future development of Eclipse projects and technologies.
Donate to the Eclipse Community
Contributions from users like you help fund the operations of the Eclipse Foundation. All money donated to the Eclipse Foundation will be used to support the Eclipse Community.
$5
$10
$35
$50
$100
35
USD
Donate
Problems donating? Donation FAQ
eclipse-inst-jre-wi....exe
Show all
Type here to search
27°C Rain showers
ENG
19:44
18-07-2022


eclipseinstaller
by Oomph

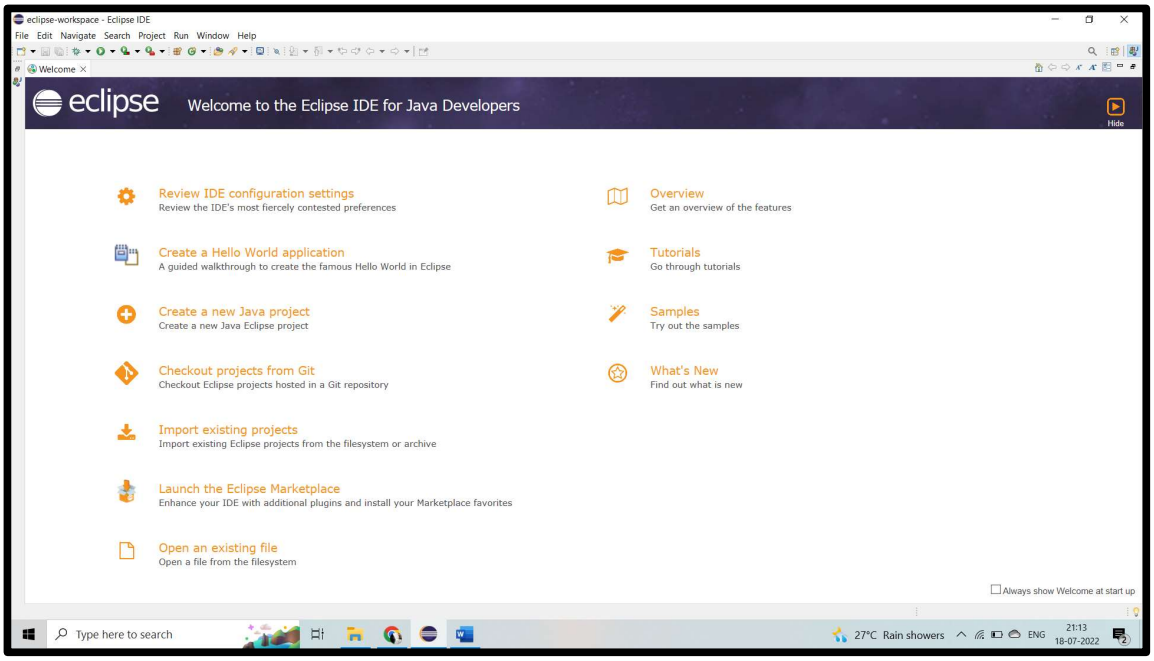
Step 3: - Click on Eclipse IDE for Java Developers



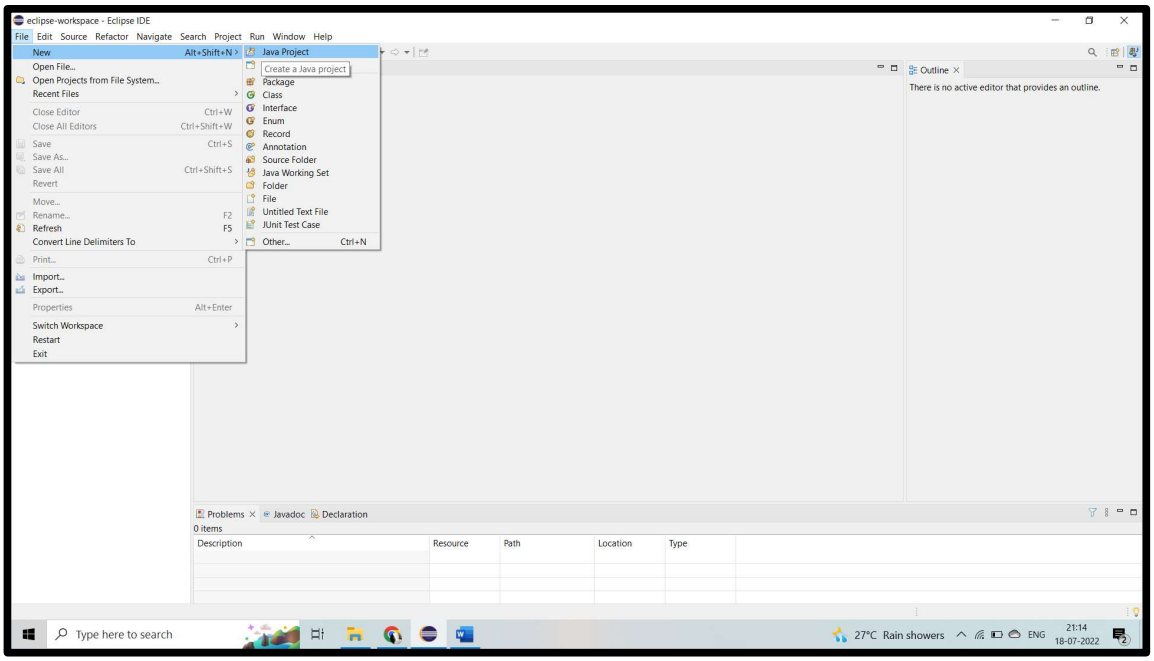
Step 4: - After installing the java developer than click on launch and set the path

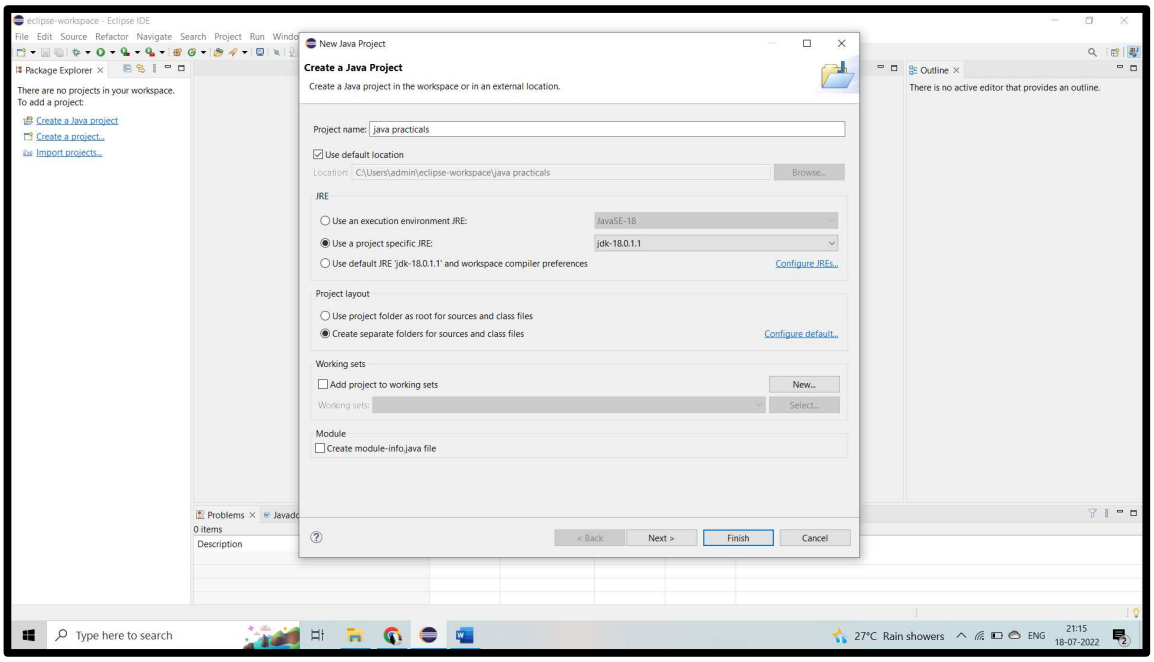



eclipseIDE
2022-06
Starting Eclipse IDE

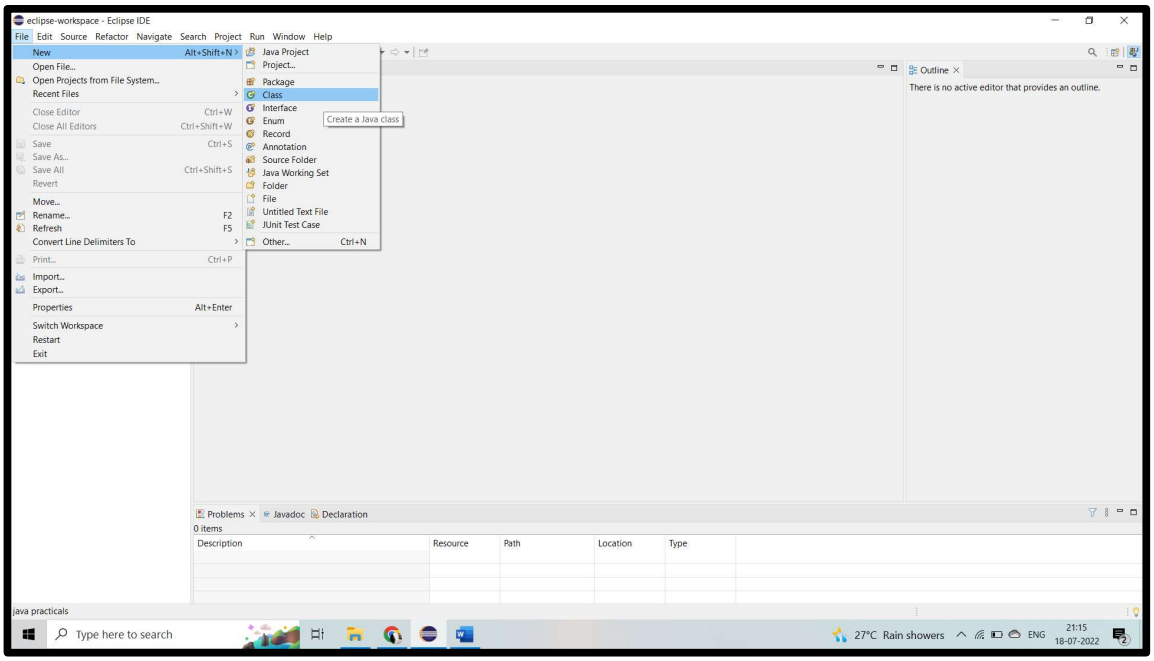

eclipse-workspace - Eclipse IDE
File Edit Navigate Search Project Run Window Help
Welcome
eclipse
Welcome to the Eclipse IDE for Java Developers
Hide
Review IDE configuration settings
Review the IDE's most fiercely contested preferences
Overview
Get an overview of the features
Create a Hello World application
A guided walkthrough to create the famous Hello World in Eclipse
Tutorials
Go through tutorials
Create a new Java project
Create a new Java Eclipse project
Samples
Try out the samples
Checkout projects from Git
Checkout Eclipse projects hosted in a Git repository
What's New
Find out what is new
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
Always show Welcome at start up
Type here to search
27°C Rain showers
ENG
21:13
18-07-2022

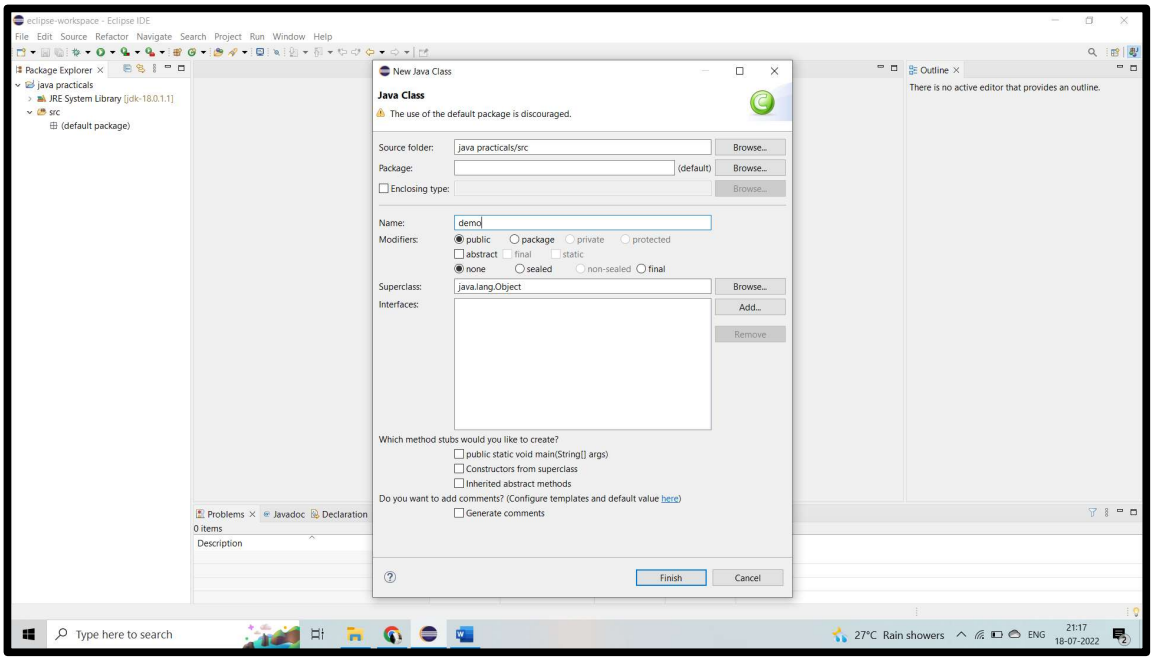
Step 5: - create a java project





Step 6: - create a java class





Create a java program to print hello world with my self

Input: -

```
public class demo {
public static void main(String args[])
{
	System.out.println("Hello World");
	System.out.println("MY NAME IS : RUDRA SAIKIRAN");
	System.out.println("ENROLLMENT NO : 20270106138");
	System.out.println("DEPARTMENT : COMPUTER");
	System.out.print("TAG LINE:- WEBSITE DEVELOPER");

}

}
```

Output: -

Practical: - 2

Aim: - Create three variables to store marks of three subjects and generate mark sheet. (Use if condition & switch case).

Input: -

```
import java.util.Scanner;

public class result
{
public static void main(String args[])
{
String name;
int enrollno,is,python,java,total;
float per;
Scanner s=new Scanner(System.in);

System.out.println("Enter Student Name:");
name=s.nextLine();
System.out.println("Enter Student's Enrollment No:");
enrollno=s.nextInt();
System.out.println("Enter Marks for information security:");
is=s.nextInt();
System.out.println("Enter Marks for python programming:");
python=s.nextInt();
System.out.println("Enter Marks for programming");
java=s.nextInt();

total=is+python+java;
per=total/3;

System.out.println("\t\t\tMarksheet");
System.out.println("**************************************************");
System.out.println("Name:"+name);
System.out.println("Enrollment No:"+enrollno);
System.out.println("-------------------------------------------------------");
System.out.println("Subjects"+"\t\t"+"Marks");
System.out.println("-------------------------------------------------------");
System.out.println("information security"+"\t\t\t"+is);
System.out.println("python programming"+"\t\t\t"+python);
System.out.println("java programming"+"\t\t\t"+java);
System.out.println("-------------------------------------------------------");
System.out.println("Total Out Of 300"+"\t"+total+"\t"+"Percentage"+per);
System.out.println("-------------------------------------------------------");

if((per>=70)&&(per<=99))
{
```

```
System.out.println("Distinction...");
}
else if((per>=60)&&(per<=69))
{
System.out.println("First Class...");
}
else if((per>=50)&&(per<=59))
{
System.out.println("Second Class...");
}
else if((per>=35)&&(per<=49))
{
System.out.println("Pass Class...");
}
else if(is<35 ||python<35||java<35)
{
System.out.println("Fail...");
}
else
{
System.out.println("Fail...");
}
System.out.println("-------------------------------------------------------");
    }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac result.java

C:\Users\admin\Desktop\Java practicals>java result
Enter Student Name:
SAIKIRAN RUDRA
Enter Student's Enrollment No:
138
Enter Marks for information security:
70
Enter Marks for python programming:
80
Enter Marks for programming
70
                        Marksheet
**************************************************
Name:SAIKIRAN RUDRA
Enrollment No:138
-------------------------------------------------------
Subjects                Marks
-------------------------------------------------------
information security                    70
python programming                      80
java programming                        70
-------------------------------------------------------
Total Out Of 300        220     Percentage73.0
-------------------------------------------------------
Distinction...
-------------------------------------------------------
```

Practical: - 3

Aim: - Write a program in Java to reverse the digits of a number using while loop.

Input: -

```
import java.util.Scanner;
class rev
{
   public static void main(String args[])
   {
         int n,i,temp=0;
         Scanner s = new Scanner(System.in);
         System.out.println("**********************");
         System.out.print("Enter value of n : ");
         n = s.nextInt();
         System.out.println("**********************");
         while(n > 0)
         {
         i = n%10;
         n = n/10;
         temp = (temp*10) + i;
         }
         System.out.println("Reverse Number = "+temp);
         System.out.println("**********************");
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac rev.java

C:\Users\admin\Desktop\Java practicals>java rev
**********************
Enter value of n : 842371
**********************
Reverse Number = 173248
**********************
```

Practical: - 4

Aim: - Write a program to demonstrate use of wrapper class.

Input: -

```
public class wraper {

   public static void main(String args[])
{
//Converting int into Integer or autoboxing

System.out.println("Converting primitive type to Object");
System.out.println("*******************************");

int a=30;

Integer i=a;                          //converting int into Integer explicitly
Integer j=a;//autoboxing, now compiler will write Integer.valueOf(a) internally
System.out.println("a="+a+" "+"i="+i+" "+"j="+j);

//Converting Integer to int or unboxing

System.out.println("Converting Object to primitive type ");
System.out.println("*******************************");

Integer b=13;
int i1=b;                 //converting Integer to int explicitly
int j1=b;//unboxing, now compiler will write a.intValue() internally
System.out.println("b="+b+" "+"i1="+i1+" "+"j1="+j1);
}
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac wraper.java

C:\Users\admin\Desktop\Java practicals>java wraper
Converting primitive type to Object
*******************************
a=30 i=30 j=30
Converting Object to primitive type
*******************************
b=13 i1=13 j1=13
```

Practical: - 5

Aim: - Write a program in Java to perform addition of two matrix.

Input: -

```
import java.util.Scanner;
class matrix
{
public static void main(String args[])
{
Scanner s = new Scanner(System.in);
int i,j,row,column;

System.out.println("***********************************");
System.out.print("Enter number of rows : ");
row = s.nextInt();
System.out.print("Enter number of columns : ");
column = s.nextInt();
System.out.println("***********************************");
int [][]a = new int[row][column]; System.out.println("Enter Matrix A :");
for(i=0;i<row;i++)
{
        for(j=0;j<column;j++)
        {
        a[i][j]=s.nextInt();
        }
}
System.out.println("***********************************");
int [][]b = new int[row][column];
System.out.println("Enter B Matrix :");
for(i=0;i<row;i++)
{
        for(j=0;j<column;j++)
        {
        b[i][j]=s.nextInt();
}
}

System.out.println("***********************************");
System.out.println("Matrix A is :");
for(i=0;i<row;i++)
{
for(j=0;j<column;j++)
{
System.out.print(" "+a[i][j]);
}
System.out.println();
}
```

```
System.out.println("**********************************");
System.out.println("B Matrix is :");
for(i=0;i<row;i++)
{
        for(j=0;j<column;j++)
        {
           System.out.print(" "+b[i][j]);
        }
        System.out.println();
}

System.out.println("**********************************");
int [][]c = new int[row][column]; System.out.println("Addition of two Matrix are :");
for(i=0;i<row;i++)
{
        for(j=0;j<column;j++)
   {
        c[i][j] = (a[i][j] + b[i][j]) ;
        System.out.print(" "+c[i][j]);
    }
        System.out.println();
}

System.out.println("**********************************");
}
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac matrix.java

C:\Users\admin\Desktop\Java practicals>java matrix
**********************************
Enter number of rows : 2
Enter number of columns : 3
**********************************
Enter Matrix A :
5
4
3
2
1
9
**********************************
Enter B Matrix :
7
8
5
1
6
2
**********************************
Matrix A is :
 5 4 3
 2 1 9
**********************************
B Matrix is :
 7 8 5
 1 6 2
**********************************
Addition of two Matrix are :
 12 12 8
 3 7 11
**********************************
```

Practical: - 6

Aim: - Using command line argument perform addition of two values.

Input: -

```
class commandDemo
{
public static void main(String args[])
{
        int a=Integer.parseInt(args[0]);
        int b=Integer.parseInt(args[1]);
        int c=a+b;
        System.out.println("a+b="+c);
}

}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac commandDemo.java

C:\Users\admin\Desktop\Java practicals>java commandDemo 10 20
a+b=30
```

Practical: - 7

Aim: - Write a program in Java to demonstrate use of this keyword.

Input: -

```
class point1
{
    int x,y;
    void init(int x,int y)
  {
     this.x=x;
     this.y=y;
  }
void disp()
  {
    System.out.println("x="+x);
    System.out.println("y="+y);
   }
}

class point
{
    public static void main(String[] args)
  {
    point1 p = new point1();
    p.init(59,91);
    p.disp();
  }

}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac point.java

C:\Users\admin\Desktop\Java practicals>java point
x=59
y=91
```

Practical: - 8

Aim: - Write a program in java to demonstrate use of default constructor, copy constructor and parameterized constructor.

Input: -

```
class copy
{
        int a,b;
        copy() //Default Constructor
  {
        a = 0;
        b = 0;
  }
copy(int a1,int b1) //Parameterized Constructor

  {
        a = a1; b =
        b1;
  }
copy(copy c1) //Copy Constructor

  {
        a = c1.a; b =
        c1.b;
  }

}

class copyd
{
public static void main(String args[])
  {
        copy c = new copy(); //Invoke Default Constructor
        copy c2 = new copy(30,45); //Invoke Parameterized Constructor
        copy c3 = new copy(c2); //Invoke Copy Constructor
        System.out.println("***************************");
        System.out.println("a = "+c2.a);
        System.out.println("b = "+c2.b);
        System.out.println("***************************");
        System.out.println("c3.a = "+c3.a);
        System.out.println("c3.b = "+c3.b);
        System.out.println("***************************");
  }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac copyd.java

C:\Users\admin\Desktop\Java practicals>java copyd
*************************
a = 30
b = 45
*************************
c3.a = 30
c3.b = 45
*************************
```

Practical: - 9

Aim: - Write a program in Java to demonstrate use of final keyword at variable level and class level.

Input: -

```
final class a //parent class declared as final class
{
        final int a = 10; //variable 'a' as final variable
        final void display() //'display' method as final method
        {
        a += 10; //value can't be incremented because variable 'a' is a final variable
   System.out.print("Value of a from Parent class = "+a);
        }
}
class b extends a //child class can't inherit parent class because parent class is final class
{
      void display() //method can't be override because method in parent class declared as final
   {
       System.out.print("Value of a from Child class = "+a);
   }
}
class findemo
{
   public static void main(String args[])
 {
    b b1 = new b(); b1.display();
  }

}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac a.java
a.java:10: error: cannot inherit from final a
class b extends a //child class can't inherit parent class because parent class is final class
                ^
a.java:6: error: cannot assign a value to final variable a
a += 10; //value can?t be incremented because variable 'a' is a final variable
^
a.java:12: error: display() in b cannot override display() in a
void display() //method can't be override because method in parent class declared as final
     ^
  overridden method is final
3 errors
```

Practical: - 10

Aim: - Write a program in Java to demonstrate use of static keyword.

Input: -

```
class d1
{
     int a=10;
     static int b=10;
     d1()
      {
          a=  a+10;
          b=b+10;
      }
   void disp1()
    {
       System.out.println("a="+a);
        System.out.println("b="+b);
    }
  static void disp2()
    {
       System.out.println("static method called...");
    }
 static
  {
      System.out.println("static block1 called...");
  }
  static
  {
      System.out.println("static block2 called...");
  }
}
class d11
{
    public static void main(String args[])
      {
          d1 d=new d1();
          d.disp1();
          d1.disp2();

          d1 d2=new d1();
          d2.disp1();
          d1.disp2();
      }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac d11.java

C:\Users\admin\Desktop\Java practicals>java d11
static block1 called...
static block2 called...
a=20
b=20
static method called...
a=20
b=30
static method called...
```

Practical: - 11

Aim: - Develop minimum 4 program based on variation in methods i.e. passing by value, passing by reference, returning values and returning objects from methods.

Develop a program based on variation in methods i.e. passing by value from methods.

Input: -

```
class x
{
        int a;
        void change (int a)
        {
                a=a+10;
        }
public static void main(String args[])
        {
                x x1=new x();
                x1.a=50;
                System.out.println("before change a is :- "+x1.a);
                x1.change(10);
                System.out.println("after change a is :- "+x1.a);
        }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac x.java

C:\Users\admin\Desktop\Java practicals>java x
before change a is :- 50
after change a is :- 50
```

Practical: - 11.1

Aim: - Develop a program based on variation in methods i.e. passing by reference from methods.

Input: -

```
class x
{
        int a;
        void change (x x2)
        {
                x2.a=x2.a+10;
        }
public static void main(String args[])
        {
                x x1=new x();
                x1.a=50;
                System.out.println("before change a is :-"+x1.a);
                x1.change(x1);
                System.out.println("after change a is :-"+x1.a);
        }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac refernce.java

C:\Users\admin\Desktop\Java practicals>java x
before change a is :-50
after change a is :-60
```

Practical: - 11.2

Aim: - Develop a program based on variation in methods i.e. returning values from methods.

Input: -

```
class x
{
	int sum (int i)
	{
		int n;
		if (i==1)
		{
			return 1;
		}
	else
		{
		n=i+ sum (i-1);
		return n;
		}
	}
}

class xy
{
	public static void main (String args[])
	{
		x x1=new x();
		System.out.println("sum of 1to9 no is :-"+x1.sum(9));
	}
}
```

Output:

```
C:\Users\admin\Desktop\Java practicals>javac return.java

C:\Users\admin\Desktop\Java practicals>java xy
sum of 1to9 no is :-45
```

Practical: - 11.3

Aim: - Develop a program based on variation in methods i.e. returning objects from methods.

Input: -

```
class A
{
   int a,b,A,B;
   A()
   {
      a=10;
      b=20;
   }

   A(int x ,int y)
   {
      A=x;
      B=y;
   }

   void Display()
   {
      System.out.println("A="+A+"\nB="+B);
   }
}

public class rudra
{
   public static void main(String[] args)
   {
      A a1=new A(40,30);
      a1.Display();
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac rudra.java

C:\Users\admin\Desktop\Java practicals>java rudra
A=40
B=30
```

Practical: - 12

Aim: - Write a program in Java to demonstrate single inheritance, multilevel inheritance, and hierarchical inheritance.

Input: -

```
class A
{

void displayA()
        {
            System.out.println("Base class A method");
        }

}

class B extends A
{

void displayB()
        {
            System.out.println("Child class B method");
        }

}


class sing
{
   public static void main(String args[])
   {
         B b1=new B();
          b1.displayA();
          b1.displayB();
    }

}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac sing.java

C:\Users\admin\Desktop\Java practicals>java sing
Base class A method
Child class B method
```

Practical: - 12.1

Aim: - Write a program in Java to demonstrate multilevel inheritance.

Input: -

```
class A
{

   void displayA()
    {
     System.out.println("base class a method");
    }
}

class B extends A
{
  void displayB()
  {
    System.out.println("class b method");
  }
}

class C extends B
{
   void displayC()
   {
    System.out.println("class c method");
   }
}

class mut
{
   public static void main (String args[])
   {
      B b1=new B();
       b1.displayA();
       b1.displayB();

       C c1=new C();
       c1.displayA();
       c1.displayC();
    }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac mut.java

C:\Users\admin\Desktop\Java practicals>java mut
base class a method
class b method
base class a method
class c method
```

Practical: - 12.2

Aim: - Write a program to show Hierarchical inheritance

Input: -

```
class A
{

   void displayA()
    {
     System.out.println("base class a method");
    }
}

class B extends A
{
  void displayB()
  {
    System.out.println("class b method");
  }
}

class C extends A
{
   void displayC()
   {
    System.out.println("class c method");
   }
}
class D extends A
{
   void displayD()
   {
    System.out.println("class d method");
   }
}
class multi
{
   public static void main (String args[])
   {
      B b1=new B();
       b1.displayA();
       b1.displayB();

       C c1=new C();
       c1.displayA();
       c1.displayC();
```

```
        D d1=new D();
        d1.displayA();
        d1.displayD();
    }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac multi.java

C:\Users\admin\Desktop\Java practicals>java multi
base class a method
class b method
base class a method
class c method
base class a method
class d method
```

Practical: - 13

Aim: - Write a program in Java in which a subclass constructor invokes the constructor of the super class and instantiate the values.

Input: -

```
class a
{
a(String s)
    {
        System.out.println("Base Class Constructor : "+s);
    }
}
class b extends a
{
b(String s)
    {
        super(s); //access parent class constructor using super keyword
        System.out.println("Child Class Constructor : "+s);
        System.out.println("*****************************************");
    }
}
class superr
{
public static void main(String args[])
   {
        System.out.println("*****************************************");
        b b1 = new b("Saikiran Rudra");
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac p13.java

C:\Users\admin\Desktop\Java practicals>java superr
*****************************************
Base Class Constructor : Saikiran Rudra
Child Class Constructor : Saikiran Rudra
*****************************************
```

Practical: - 14

Aim: - Write a program that illustrates interface inheritance. Interface P12 inherits from both P1 and P2. Each interface declares one constant and one method. The class Q implements P12. Instantiate Q and invoke each of its methods. Each method displays one of the constants.

Input: -

```
interface P
{
   final int p=55;
   void dispP();
}
interface P1 extends P
{
   final int p1=80;
   void dispP1();
}
interface P2 extends P
{
   final int p2=70;
   void dispP2();
}
interface P12 extends P1,P2
{
   final int p12=90;
   void dispP12();
}
class Q implements P12
{
   public void dispP()
   {
      System.out.println("dispP : "+p1);
   }
   public void dispP1()
   {
      System.out.println("dispP1 : "+p2);
   }
   public void dispP2()
   {
      System.out.println("dispP2 : "+p12);
   }
   public void dispP12()
   {
      System.out.println("dispP12 : "+p);
   }

}
```

```
class Interface
{
   public static void main(String arg[])
   {
      Q q=new Q();
      q.dispP();
      q.dispP1();
      q.dispP2();
      q.dispP12();
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac interface.java

C:\Users\admin\Desktop\Java practicals>java Interface
dispP : 80
dispP1 : 70
dispP2 : 90
dispP12 : 55
```

Practical: - 15

Aim: - Write a program in Java to demonstrate implementation of multiple inheritance using interfaces.

Input: -

```
interface a1 //1st interface a1
{
    int a = 90;
    void add();
}
interface a2 //2nd interface a2
{
   int b = 10;
   void sub();
}
class b implements a1,a2 //class b multiple implements a1,a2 interfaces
{
public void add()
  {
      System.out.println("Addition of two numbers = "+(a+b));
  }
public void sub()
  {
      System.out.println("Subtraction of two numbers = "+(a-b));
  }
void mul()
  {
      System.out.println("Multiplication of two numbers = "+(a*b));
   }
}
class multi_inter
{
public static void main(String args[])
 {
      b b1 = new b();
      System.out.println("**********************************");
      b1.add();
      b1.sub();
      b1.mul();
      System.out.println("**********************************");
  }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac multi_inter.java

C:\Users\admin\Desktop\Java practicals>java multi_inter
********************************
Addition of two numbers = 100
Subtraction of two numbers = 80
Multiplication of two numbers = 900
********************************
```

Practical: - 16

Aim: - Describe abstract class called Shape which has three subclasses say Triangle, Rectangle, and Circle. Define one method area () in the abstract class and override this area () in these three subclasses to calculate for specific object i.e., area () of Triangle subclass should calculate area of triangle etc. Same for Rectangle and Circle.

Input: -

```
abstract class shape //abstract class shape
{
      double l,w,r;
      final double pi = 3.14; //fix the value of pi using final keyword

   shape(double l,double w)
     {
          this.l = l;
         this.w = w;
     }
    shape(double r)
     {
           this.r = r;
     }
    abstract double area(); //abstract method, it can't contain body
}

class rectangle extends shape //1st child class to calculate the area of rectangle
{
   rectangle(double a,double b)
     {
          super(a,b); //access parent class constructor using super keyword
     }
   double area()
     {
           System.out.println("Area of Rectangle = "+(l * w));
            return 0;
      }
}

class triangle extends shape //2nd child class to calulate the area of triangle
{
    triangle(double i,double j)
    {
         super(i,j);
     }
    double area()
     {
         System.out.println("Area of Triangle = "+(0.5 * l * w));
```

```
            return 0;
        }
}
class circle extends shape //3rd child class to calulate the area of circle
{
     circle(double k)
     {
         super(k);
     }
double area()
     {
          System.out.println("Area of Circle = "+(pi * r * r));
          return 0;
      }
}
class areasum
{
     public static void main(String args[])
     {
          System.out.println("****************************");
          shape s; //create the reference of parent class

          rectangle r = new rectangle(8,5); //create object of 1st child class
          s = r; // refer child class object to the reference of parent class
          r.area(); //call abstract method

          triangle t = new triangle(5,8); //create object of 2nd child class
          s = t;
          s.area();

          circle c = new circle(8); //create object of 3rd child class
          s = c;
          s.area();
          System.out.println("****************************");
     }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac multi_inter.java

C:\Users\admin\Desktop\Java practicals>java multi_inter
**********************************
Addition of two numbers = 100
Subtraction of two numbers = 80
Multiplication of two numbers = 900
**********************************
```

Practical: - 17

Aim: - Write an application that illustrates method overriding in the same package and different packages. Also demonstrate accessibility rules in inside and outside packages.

Input: -

Package 1

```
package pk1;
class s
{
   public static void main(String args[])
        {
                System.out.print("this is from package1");
        }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>cd p17

C:\Users\admin\Desktop\Java practicals\p17>javac -d . s.java

C:\Users\admin\Desktop\Java practicals\p17>java pk1.s
this is from package1
```

Package 2

```
package pk2;

public class s1

{

   public void disp()

            {

                System.out.print("this is from packag2");

            }

}
```

Main

```
import pk2.s1;
class x
{
  public static void main(String args[])
      {
          s1 s11=new s1();
          s11.disp();
      }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals\p17>javac -d . s1.java

C:\Users\admin\Desktop\Java practicals\p17>javac x.java

C:\Users\admin\Desktop\Java practicals\p17>java x
this is from packag2
```

```
C:\Users\admin\Desktop\Java practicals>cd p17

C:\Users\admin\Desktop\Java practicals\p17>javac -d . s.java

C:\Users\admin\Desktop\Java practicals\p17>java pk1.s
this is from package1
C:\Users\admin\Desktop\Java practicals\p17>javac -d . s1.java

C:\Users\admin\Desktop\Java practicals\p17>javac x.java

C:\Users\admin\Desktop\Java practicals\p17>java x
this is from packag2
```

Practical: - 18

Aim: - Write a program in Java to demonstrate multiple try block and multiple catch exception and include 'divide by zero' and 'Arithmetic exception'.

Input: -

```
class P18
 {
  public static void main(String args[])
  {
   System.out.println("-----------------------------------------------------");
   try
   {
    try
    {
     System.out.println("Going to divide by 0");
     int a = 2 / 0;
    }
    catch (ArithmeticException e)
    {
     System.out.println("Divide by Zero Error...");
    }
    System.out.println("-----------------------------------------------------");
    try
    {
     int b[] = { 0, 1, 2 };
     System.out.println(b[3]);
    }
    catch (ArrayIndexOutOfBoundsException e)
    {
     System.out.println("Array Index Error...");
    }
    System.out.println("-----------------------------------------------------");
   }
   catch (Exception e)
   {
    e.printStackTrace();
   }
  }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac p18.java

C:\Users\admin\Desktop\Java practicals>java P18
-------------------------------------------------------
Going to divide by 0
Divide by Zero Error...
-------------------------------------------------------
Array Index Error...
---------------------------------------------------------
```

Practical: - 19

Aim: - Write a program in java to demonstrate use defined exception.

Input: -

```
class div
{
   static void div(int a,int b) throws uexc //static division method
   {
      if(b == 0) //if second number is zero then throw exception
      {
         throw new uexc(b);
      }
      else
      {
         System.out.println(a / b); //if no error generated then simply division isperformed
      }
   }

   public static void main(String args[])
   {
      System.out.println("------------------------------------------------------------");
      try
      {
         div(2,0); //second number is zero so it throws exception
      }
      catch(uexc c)
      {
         c.printStackTrace();
      }
      System.out.println("------------------------------------------------------------");
   }
}
class uexc extends Exception //extends Exception class for the userdefined exception
{
   int b;
   uexc(int b)
   {
      this.b = b;
   }
   public String toString() //method for user defined exception message
   {
      return "User defined exception divide by zero...";
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac div.java

C:\Users\admin\Desktop\Java practicals>java div
-----------------------------------------------------------
User defined exception divide by zero...
        at div.div(div.java:7)
        at div.main(div.java:20)
-----------------------------------------------------------
```

Practical: - 20

Aim: - Write a small application in Java to develop Banking Application in which user deposits the amount Rs 2000.00 and then start withdrawing of Rs 1500.00, Rs 400.00 and it throws exception "Not Sufficient Fund" when user withdraws Rs. 500 thereafter.

Input: -

```
import java.util.Scanner;

class banks
{
   public static void main(String args[])
   {
      Scanner s = new Scanner(System.in);
      int opt, amount, balance = 2000;
      do
      {
         System.out.println("-------------------------------------------------------------------");
         System.out.println("1.Deposit Money");
         System.out.println("2.Withdraw Money");
         System.out.println("3.Check Balance");
         System.out.println("4.Exit");
         System.out.println("-------------------------------------------------------------------");
         System.out.print("Enter Option : ");
         opt = s.nextInt();
         System.out.println("-------------------------------------------------------------------");
         switch (opt)
         {
            case 1:
            System.out.print("Enter Amount to Deposit : ");
            amount = s.nextInt();

            if (amount > 0)
            {
               balance = balance + amount;
               System.out.println(
               amount + " has been Deposited in yourAccount..."
               );
            }
            else
            {
               System.out.println("Negative Amount can't be deposit");
            }
            break;

            case 2:
```

```
            System.out.print("Enter amount to Withdraw : ");
            amount = s.nextInt();
            if (amount <= 0)
            {
               System.out.println("Enter Positive Amount ...");
            }
            else if (amount > balance)
            {
               System.out.println("Insufficient Money ...");
            }
            else
            {
               balance =  balance - amount;
               System.out.println(amount + " has been Withdrawn... ");
            }
            break;

            case 3:
            System.out.println("Your Account Balance = " + balance);
            break;

            case 4:
            System.exit(0);
            default:
            System.out.println("Wrong Option Entered. ..");
            System.out.println("-------------------------------------------------------");
            break;
         }
      }
      while (opt <= 4);
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac banks.java

C:\Users\admin\Desktop\Java practicals>java banks
------------------------------------------------------------
1.Deposit Money
2.Withdraw Money
3.Check Balance
4.Exit
------------------------------------------------------------
Enter Option : 3
------------------------------------------------------------
Your Account Balance = 2000
------------------------------------------------------------
1.Deposit Money
2.Withdraw Money
3.Check Balance
4.Exit
------------------------------------------------------------
Enter Option : 2
------------------------------------------------------------
Enter amount to Withdraw : 1500
1500 has been Withdrawn...
------------------------------------------------------------
1.Deposit Money
2.Withdraw Money
3.Check Balance
4.Exit
------------------------------------------------------------
Enter Option : 3
------------------------------------------------------------
Your Account Balance = 500
------------------------------------------------------------
```

```
-----------------------------------------------
1.Deposit Money
2.Withdraw Money
3.Check Balance
4.Exit
-----------------------------------------------
Enter Option : 2
-----------------------------------------------
Enter amount to Withdraw : 400
400 has been Withdrawn...
-----------------------------------------------
1.Deposit Money
2.Withdraw Money
3.Check Balance
4.Exit
-----------------------------------------------
Enter Option : 3
-----------------------------------------------
Your Account Balance = 100
-----------------------------------------------
1.Deposit Money
2.Withdraw Money
3.Check Balance
4.Exit
-----------------------------------------------
Enter Option : 2
-----------------------------------------------
Enter amount to Withdraw : 500
Insufficient Money ...
-----------------------------------------------
1.Deposit Money
2.Withdraw Money
```

```
3.Check Balance
4.Exit
----------------------------------------
Enter Option : 4
----------------------------------------
```

Practical: - 21

Aim: - Write a program that executes two threads. One thread displays "Thread1" every 3,000 milliseconds, and the other displays "Thread2" every 5,000 milliseconds. Create the threads by extending the Thread class

Input: -

```
class pthreads
{
   public static void main(String args[])
   {
      tt t1 = new tt("Thread 1");
      tt t2 = new tt("Thread 2");
   }
}

class tt extends Thread //extend Thread class
{
   tt(String s)
   {
      super(s);
      start();
   }
   public void run()
   {
      for(int i=0;i<5;i++)
      {
         System.out.println(getName());
         {
            if(getName() == "Thread 1")
            try
            {
               Thread.sleep(3000);
            }
            catch (Exception e) {}
            else
            try
            {
               Thread.sleep(5000);
            }
            catch (Exception e) {}
         }
      }
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac pthreads.java

C:\Users\admin\Desktop\Java practicals>java pthreads
Thread 2
Thread 1
Thread 1
Thread 2
Thread 1
Thread 1
Thread 2
Thread 1
Thread 2
Thread 2
```

Practical: - 22

Aim: - Write a program in Java to demonstrate use of synchronization of threads when multiple threads are trying to update common variable

Input: -

```
class number1
{
   public static void main(String args[])
   {
      num nu = new num();
      numb num1 = new numb(nu);
      numb num2 = new numb(nu);
      numb num3 = new numb(nu);
      numb num4 = new numb(nu);
      numb num5 = new numb(nu);
   }
}

class num
{
   int n;

   synchronized void increment()
   {
      n++;
      try
      {
         Thread.sleep(2000);
      }
      catch (Exception e) {}
      System.out.println("Number = " + n);
   }
}

class numb extends Thread
{
   num n1;

   numb(num n1)
   {
      this.n1 = n1;
      start();
   }

   public void run()
   {
      n1.increment();
```

```
    }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac number1.java

C:\Users\admin\Desktop\Java practicals>java number1
Number = 1
Number = 2
Number = 3
Number = 4
Number = 5
```

Practical: - 23

Aim: - Write a program in java to use String class and compare two strings

Input: -

```
class True
{

   public static void main(String args[])
   {
      String s1 = "Saikiran";
      String s2 = "Saikiran";
      String s3 = new String("Saikiran");
      String s4 = "jayesh";
      String s5 = "Nikhal";
      System.out.println(s1.equals(s2)); //true
      System.out.println(s1.equals(s3)); //true
      System.out.println(s1.equals(s4)); //false
      System.out.println(s1.equals(s2)); //false
      System.out.println(s1.equalsIgnoreCase(s2)); //true
      System.out.println(s1 == s2); //true
      System.out.println(s1 == s3); //false
      System.out.println(s1.compareTo(s2)); //0
      System.out.println(s1.compareTo(s3)); //1(because s1>s3)
      System.out.println(s3.compareTo(s1)); //-1(because s3 < s1 )
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac True.java

C:\Users\admin\Desktop\Java practicals>java True
true
true
false
true
true
true
false
0
0
0
```

Practical: - 24

Aim: - Write a program in java to use String Buffer class and perform concatenation of two Strings.

Input: -

```
class buf
{
   public static void main(String args[])
   {
      StringBuffer sb=new StringBuffer("Hello ");
      sb.append("Java"); //now original string is changed
      System.out.println(sb); //prints Hello Java
   }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac buf.java

C:\Users\admin\Desktop\Java practicals>java buf
Hello Java
```

Practical: - 25

Aim: - Write a program in Java to create, write, modify, read operations on a Text file

Input: -

```
import java.io.*; //Import all classes from io package

class read
{

   public static void main(String args[])
   {
      try
      {
         File f = new File("C:/pp/test1.txt");
         f.createNewFile();
         System.out.println("----------------------------------------------------------------------");
         System.out.println("File created Successfully...");
         System.out.println("----------------------------------------------------------------------");
      }
      catch (Exception e)
      {
         e.printStackTrace();
      }
      try
      {
         FileOutputStream w = new FileOutputStream("test1.txt");
         String s = "narikiaS arduR";
         byte b[] = s.getBytes();
         w.write(b);
         w.close();
         System.out.println("Writing Complete");
         System.out.println("----------------------------------------------------------------------");
      }
      catch (Exception e)
      {
         e.printStackTrace();
      }
      try
      {
         FileInputStream r = new FileInputStream("test1.txt");
         int i = 0;
         while ((i = r.read()) != -1) System.out.print((char) i);
         System.out.println();
         r.close();
         System.out.println("Reading Complete");
```

```
            System.out.println("------------------------------------------------------------");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
        try
        {
            FileOutputStream ww = new FileOutputStream("test1.txt");
            String s1 = "Saikiran Rudra";
            byte b1[] = s1.getBytes();
            ww.write(b1);
            ww.close();
            System.out.println("Modification Complete");
            System.out.println("------------------------------------------------------------");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
        try
        {
            FileInputStream rr = new FileInputStream("test1.txt");
            int j = 0;
            while ((j = rr.read()) != -1) System.out.print((char) j);
            System.out.println();
            rr.close();
            System.out.println("Reading Complete");
            System.out.println("------------------------------------------------------------");
        }
        catch (Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

Output: -

```
C:\Users\admin\Desktop\Java practicals>javac read.java

C:\Users\admin\Desktop\Java practicals>java read
-----------------------------------------------------------
File created Successfully...
-----------------------------------------------------------
Writing Complete
-----------------------------------------------------------
narikiaS arduR
Reading Complete
-----------------------------------------------------------
Modification Complete
-----------------------------------------------------------
Saikiran Rudra
Reading Complete
-----------------------------------------------------------
```