

# INDEX

S.No	TITLE OF THE EXPERIMENT	Page No.	Staff Sign.
1.	AREA AND PERIMETER OF A RECTANGLE	1	
2.	SUBSTRING FIND AND REPLACE	4	
3.	SORTING A GIVEN LIST OF NAMES IN ASCENDING ORDER	7	
4.	NESTED AND INNER CLASS	10	
5.	PASSING OBJECT AS PARAMETER	13	
6.	INHERITANCE	16	
7.	METHOD OVERLOADING	19	
8.	METHOD OVERRIDING USING SUPERKEYWORD	22	
9.	SIMPLE INTEREST PROGRAM USING THE INTERFACE	25	
10.	MARK STATEMENT OF A STUDENT USINGTHE PACKAGE	28	
11.	CREATE THREE THREADS AND DISPLAY MEESAGE	31	
12.	USER DEFIND ERROR HANDLING	34	
13.	ARRAY INDEX OUT OF BOUNDS EXECEPTION AND NUMBERFORMATEXECEPTION	37	
14.	APPLET THAT DISPLAYS A SIMPLE MESSAGE	40	
15.	FACTORIAL VALUE USING APPLETT	43	
16.	DRAW LINES, RECTANGLES AND OVALS	46	

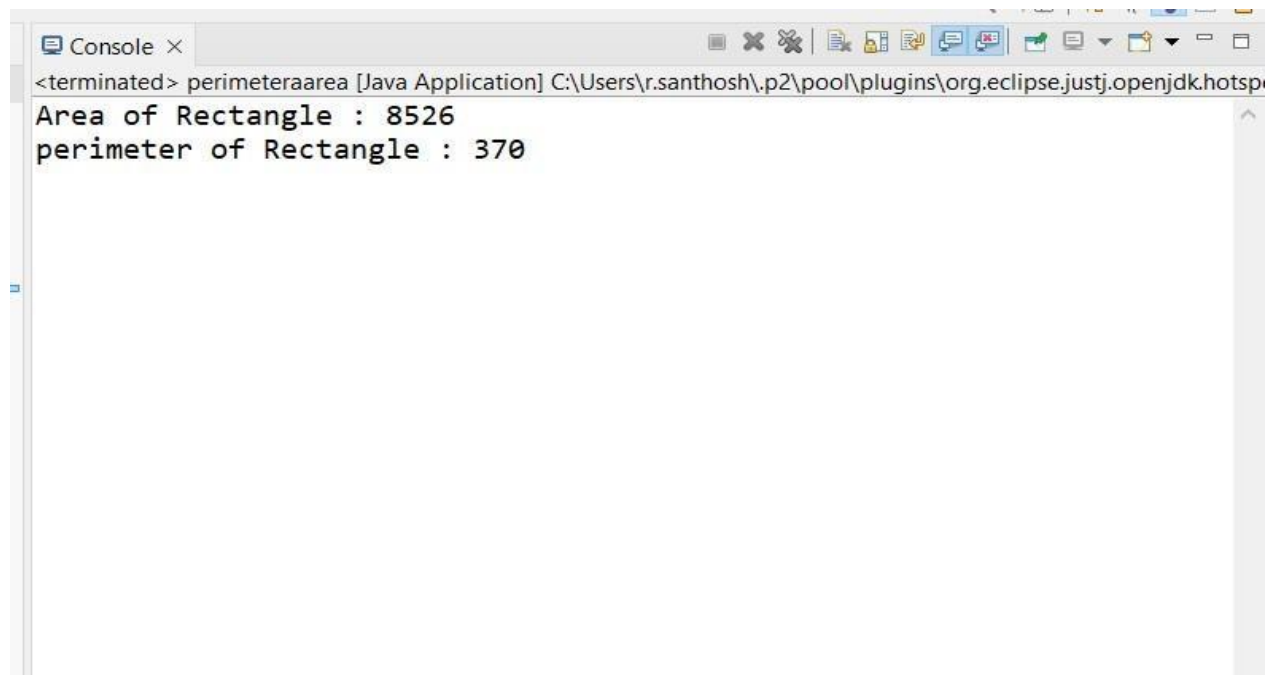
## **PROGRAM:-**

**//Ex:No:1. To find the area and perimeter of a Rectangle using  
BufferedReader class.**

```
class call
{ int w,h; call(int w1, int
    h1)
    { w=w1;
    h=h1; } int
    area()
    { int a=w*h;
        return a;
    }
    int perimeter()
    { int peri=2*(w+h);
        return peri;
    }
}

public class perimeterArea
{ public static void main(String args[])
    { call obj=new call(87,98); int
        i=obj.area();
        System.out.println("Area of Rectangle : " + i);
        int j=obj.perimeter();
        System.out.println("perimeter of Rectangle: " + j); }
}
```

## OUTPUT :



The screenshot shows the Eclipse IDE's console window. The title bar reads "Console x". The console content shows the application has terminated, followed by the output of a Java application named "perimetrearea". The output consists of two lines: "Area of Rectangle : 8526" and "perimeter of Rectangle : 370". The console window has a standard toolbar with icons for running, debugging, and other IDE functions.

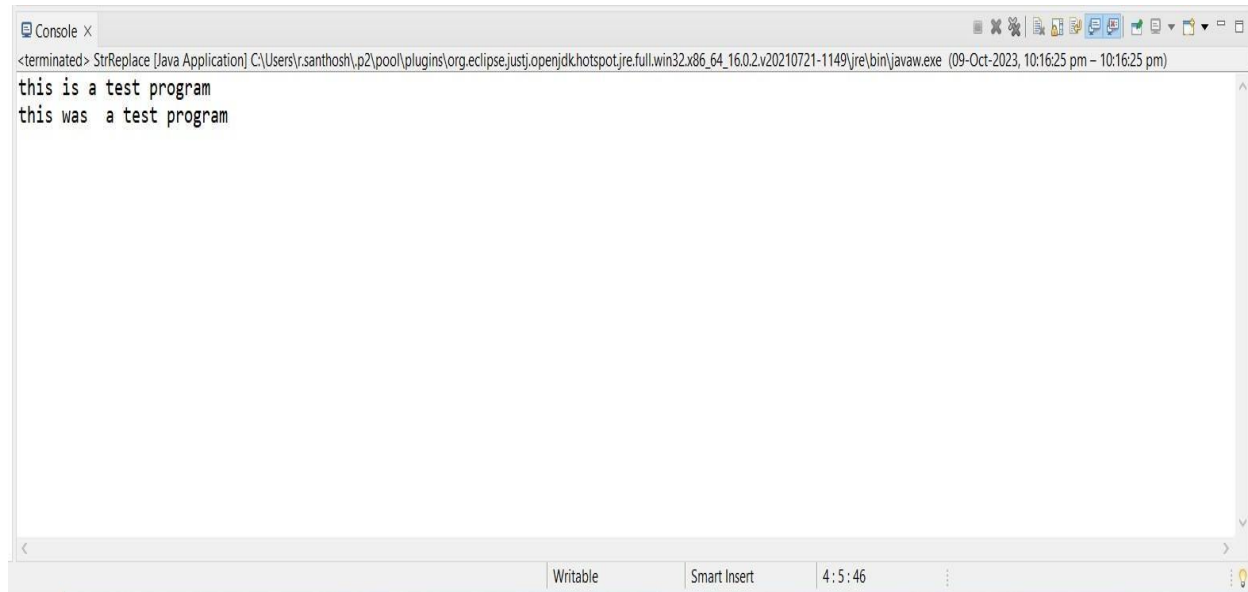
```
<terminated> perimetrearea [Java Application] C:\Users\r.santhosh\.p2\pool\plugins\org.eclipse.justj.openjdk.hotsp  
Area of Rectangle : 8526  
perimeter of Rectangle : 370
```

## **PROGRAM:**

### **//Ex.No:2. Substring replacement**

```
class StrReplace
{ public static void main(String args[])
{
    String org="this is a test program";
    String search=" is";
    String sub=" was "; String
    result=" ";
    int i; do
    {
        System.out.println(org);
        i =org.indexOf(search); if(i!=-1)
        {
            result=org.replace(search,sub);
            org=result;
        }
    }
    while(i!=-1);
}
}
```

## OUTPUT :



The screenshot shows the Eclipse IDE's console window. The title bar reads "Console x". The status bar at the top indicates the application has terminated: "<terminated> StrReplace [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_16.0.2.v20210721-1149\jre\bin\javaw.exe (09-Oct-2023, 10:16:25 pm - 10:16:25 pm)". The console output displays two lines of text: "this is a test program" followed by "this was a test program" on the next line. The status bar at the bottom shows "Writable", "Smart Insert", and a timer at "4:5:46".

```
<terminated> StrReplace [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (09-Oct-2023, 10:16:25 pm - 10:16:25 pm)
this is a test program
this was a test program
```

## PROGRAM:

### //Ex.No:3. sorting a given list of names in ascending order

```
import java.lang.*; import java.io.*; public class Sortng
{ public static void main(String args[]) throws IOException {
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
    System.out.print("enter how many strings"); int
    n=Integer.parseInt(br.readLine());
    String x[]=new String[n];
    System.out.println("enter    "+n+"    strings");
    for(int i=0;i<n;i++)
    {
        x[i]=br.readLine();
    }
    String s=new String();
    for(int i=0;i<n;i++)
    { for(int j=0;j<n;j++)
        { if(x[i].compareTo(x[j])<0)
            {
                s=x[i];
                x[i]=x[j];
                x[j]=s;
            }
        }
    }
    System.out.println("string in alphabetical order are");
    for(int i=0;i<n;i++)
    {
        System.out.println(x[i]);
    }
}
```

## OUTPUT :

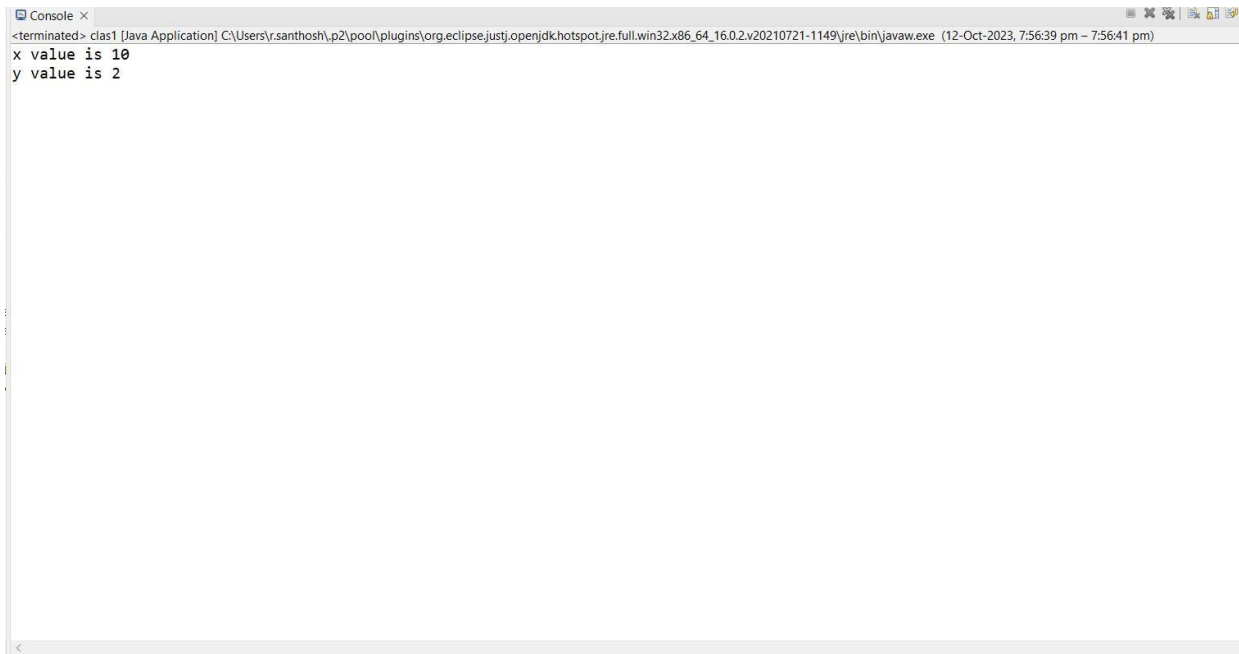
```
Console ×
<terminated> StrReplace [Java Application] C:\Users\r.santhosh\.p2\pool\plugins\org.eclipse.justj.openj
enter how many strings
3
enter 3 strings
tiger
dog
cat
string in alphabetical order are
cat
dog
tiger
```

## PROGRAM:

```
//Ex.No:4. To create nested class import java.io.*; class outer
{ int x; outer() { x=10;}
  void test()
  { class inner
    { int y; inner()
      { y=2;
        } void display()
        {
          System.out.println("y value is "+y);
        }
      }
    System.out.println("x value is "+ x);
    inner I=new inner();
    I.display();
  }
} public class nested
{ public static void main(String args[])
  { outer o=new outer();
    o.test();
    outer.inner i=outer.new inner();
    i.test();
  }
}
```



## OUTPUT :



```
Console x
<terminated> clas1 [Java Application] C:\Users\y.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_16.0.2.v20210721-1149\jre\bin\javaw.exe (12-Oct-2023, 7:56:39 pm - 7:56:41 pm)
x value is 10
y value is 2
```

## **PROGRAM:**

**//Ex.No:5. To pass object as parameter**

```
class test
{ int a; int b; test ( int i, int j)
  {
    a=i;
    b=j; } boolean equals(test o)
  { if(o.a==a && o.b==b) return true; else
    return false;
  }
} class passob
{ public static void main(String args[])
  { test ob1=new test (10,22); test ob2=new
    test(100,22); test ob3=new test(-1,-1);
    System.out.println("ob1==ob2:"+ob1.equals(ob2));
    System.out.println("ob1==ob3:"+ob3.equals(ob3));
  }
}
```

## OUTPUT :



The screenshot shows a console window titled "Console" with a close button. The text inside the console is as follows:

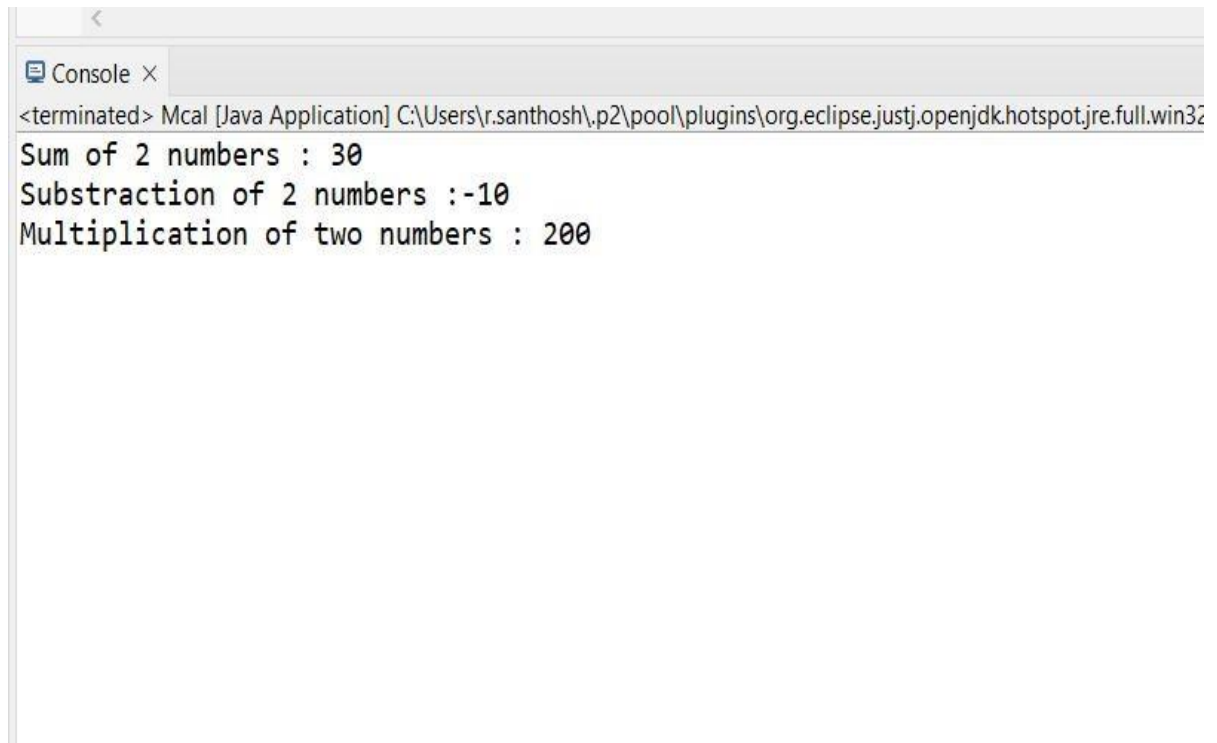
```
<terminated> passob [Java Application] C:\Users\r.santhosh\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64
ob1==ob2:false
ob1==ob3:true
```

## **PROGRAM:-**

**//Ex.No:6. To display arithmetic operations using inheritance.**

```
Class cal
{
    int a,b; void sum(int x,int y)
    { a=x+y;
      System.out.println("Sum of 2 numbers : " +a);
    } void sub(int x,int y)
    { b=x-y;
      System.out.println("Substraction of 2 numbers : " +a);
    }
}
class Mcal extends cal
{ int c;
  void mul(int x,int y)
  { c=x*y;
    System.out.println("Multiplication of two numbers : "+b);
  }
  public static void main(String args[])
  {
    Mcal d= new Mcal(); d.sum(10,20);
    d.sub(10,20);
    d.mul(10,20);
  }
}
```

## OUTPUT :



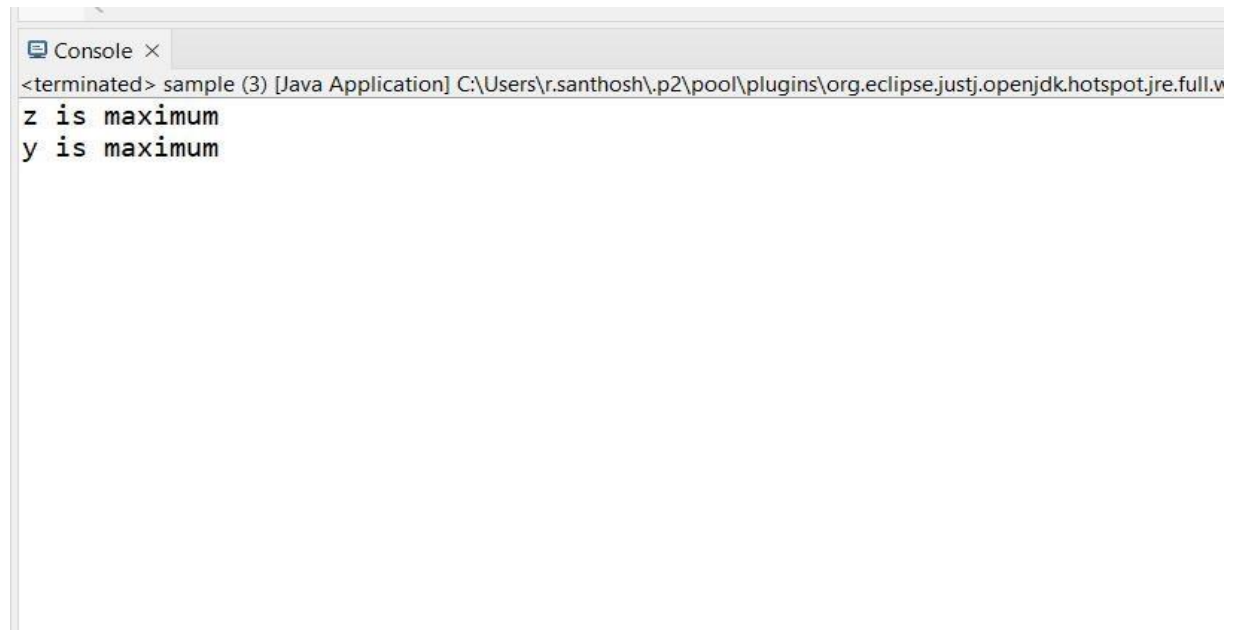
```
<terminated> Mcal [Java Application] C:\Users\r.santhosh\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32
Sum of 2 numbers : 30
Substraction of 2 numbers :-10
Multiplication of two numbers : 200
```

## PROGRAM:

**//Ex.No:7. To compare numbers using method overloading.**

```
import java.io.*; class methodOverloading
{
    int x,y,z;
    public void maxi (int a,int b,int c)
    { x=a; y=b; z=c;
        if(x>y & x>z)
            System.out.println("x is maximum"); if(y>x & y>z)
            System.out.println(" y is maximum"); else
            System.out.println("z is maximum");
    } void maxi(int a,int b)
    { x=a; y=b; if(x>y)
    {
        System.out.println("x is maximum");
    }
    else
    {
        System.out.println("y is maximum");
    }
    } } public class sample
{ public static void main(String args[])
{ methodOverloading m;
  m=newmethodOverloading();
  m.maxi(10,20,30);
  m.maxi(55,86);
}
}
```

## OUTPUT :



```
Console x
<terminated> sample (3) [Java Application] C:\Users\r.santhosh\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.v
z is maximum
y is maximum
```

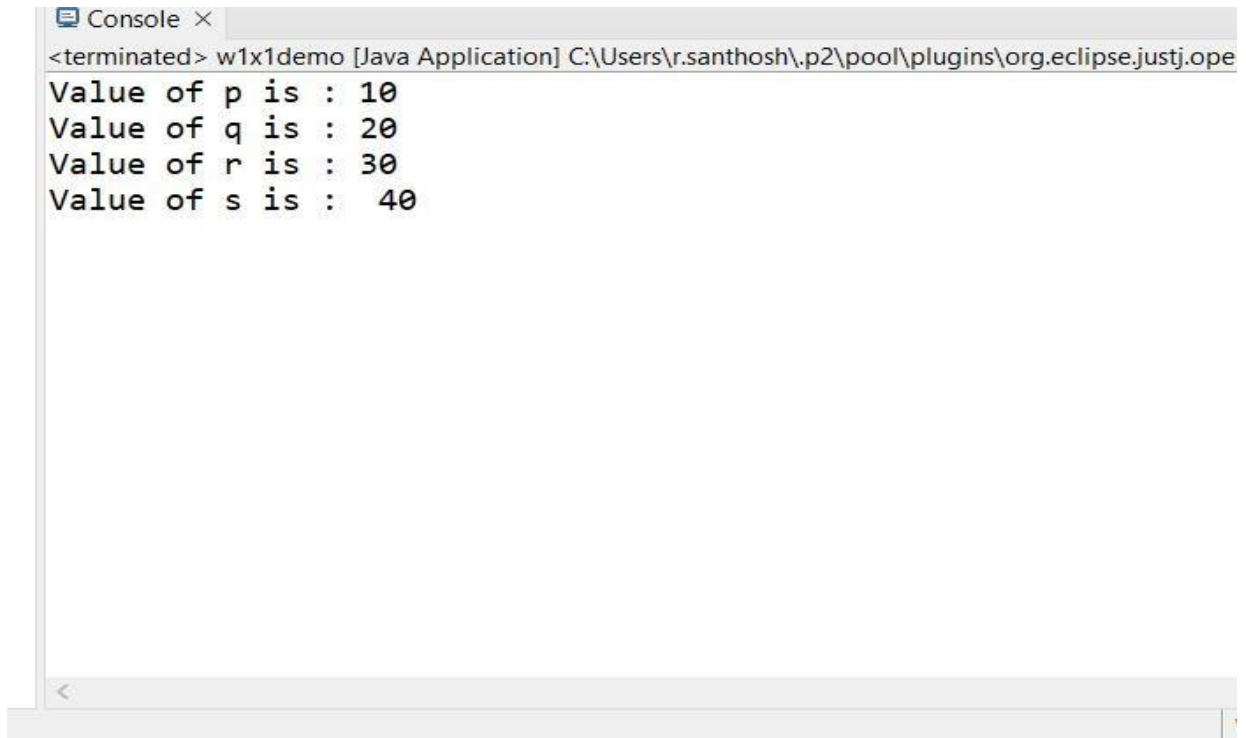
## PROGRAM:

**//Ex.NO:8. To override a method with super keyword**

```
class w1
{ int p,q; w1(int p,int q)
  {
    this.p=p;
    this.q=q;}
  void show()
  {
    System.out.println("Value of p is : " +p);
    System.out.println("Value of q is : " +q);
  }
} class x1 extends w1
{
  int r,s;
  x1(int p, int q, int r, int s)
  {
    super(p,q); this.r=r;
    this.s=s;
  } void show()
  {
    Super.show();
    System.out.println("Value of r is : " +r);
    System.out.println("Value of s is : " +s);
  }
} class wx1demo
{ public static void main(String args[])
  { x1 a=new x1(10,20,30,40); a.show();
  }
}
```



## OUTPUT :



The screenshot shows a console window titled "Console" with a close button. The text inside the console indicates that a Java application named "w1x1demo" has terminated. The output of the application is displayed as follows:

```
<terminated> w1x1demo [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.ope  
Value of p is : 10  
Value of q is : 20  
Value of r is : 30  
Value of s is : 40
```

## **PROGRAM:**

**//Ex.No:9. simple interest program using the interface**

interface deposit

```
{ void interest(double principal, int year, double rate);  
}
```

class simple implements deposit

```
{ public void interest(double p, int n, double r)  
    { double intr, total;  
      intr=p*n*r/100.  
      0; total=p+intr;  
      System.out.println("\nPrincipal amount: Rs. "+p);  
      System.out.println("No. of years: "+n);  
      System.out.println("Interest rate: "+r);  
      System.out.println("Interest amount:Rs. "+intr);  
      System.out.println("Total amount after "+n+ "years in simple interest: Rs.  
        "+total); }  
}
```

class interestcal

```
{ public static void main(String args[])  
    {   simple  sim=new  simple();  
        sim.interest(12500.0,    4,  
                    12.5);  
    }  
}
```

## OUTPUT :

Console ×  
<terminated> interestcal [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.f

Principal amount: Rs. 12500.0

No. of years: 4

Interest rate: 12.5

Interest amount:Rs. 6250.0

Total amount after 4years in simple interest: Rs. 18750.0

## PROGRAM:

```
//Ex.No:10. mark statement of a student using the package package p1;
import

java.io.*; public class stud
{ public int rno; public String name; public
  int m1, m2; public int total; public
  void input()
  { rno= 131;
    System.out.println("Roll number: "+rno); name= "sudha";
    System.out.println("Name: "+name); m1= 98;
    System.out.println("Mark1: "+m1); m2= 99;
    System.out.println("Mark2: "+m2);
  } public void process()
  { total= m1 + m2;
    System.out.println("Total: "+total);
  }
} package p2; import java.io.*;
import p1.*; public class details
{ public String game= "football"; public void display()
  {
    System.out.println("Game: "+game);
  }
}
package p3; import java.io.*; import p1.*; import p2.*;
class multi
{ public static void main(String args[])
  { stud s= new stud(); detailsd=new details(); s.input();
    s.process();
    d.display();
  }
}
```

## OUTPUT :

A screenshot of an Eclipse IDE console window. The window has a title bar with a close button and the text 'Console x'. Below the title bar, the console output is displayed. The first line is '<terminated> multi [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.j'. The subsequent lines are 'Roll number: 131', 'Name: sudha', 'Mark1: 98', 'Mark2: 99', 'Total: 197', and 'Game: football'.

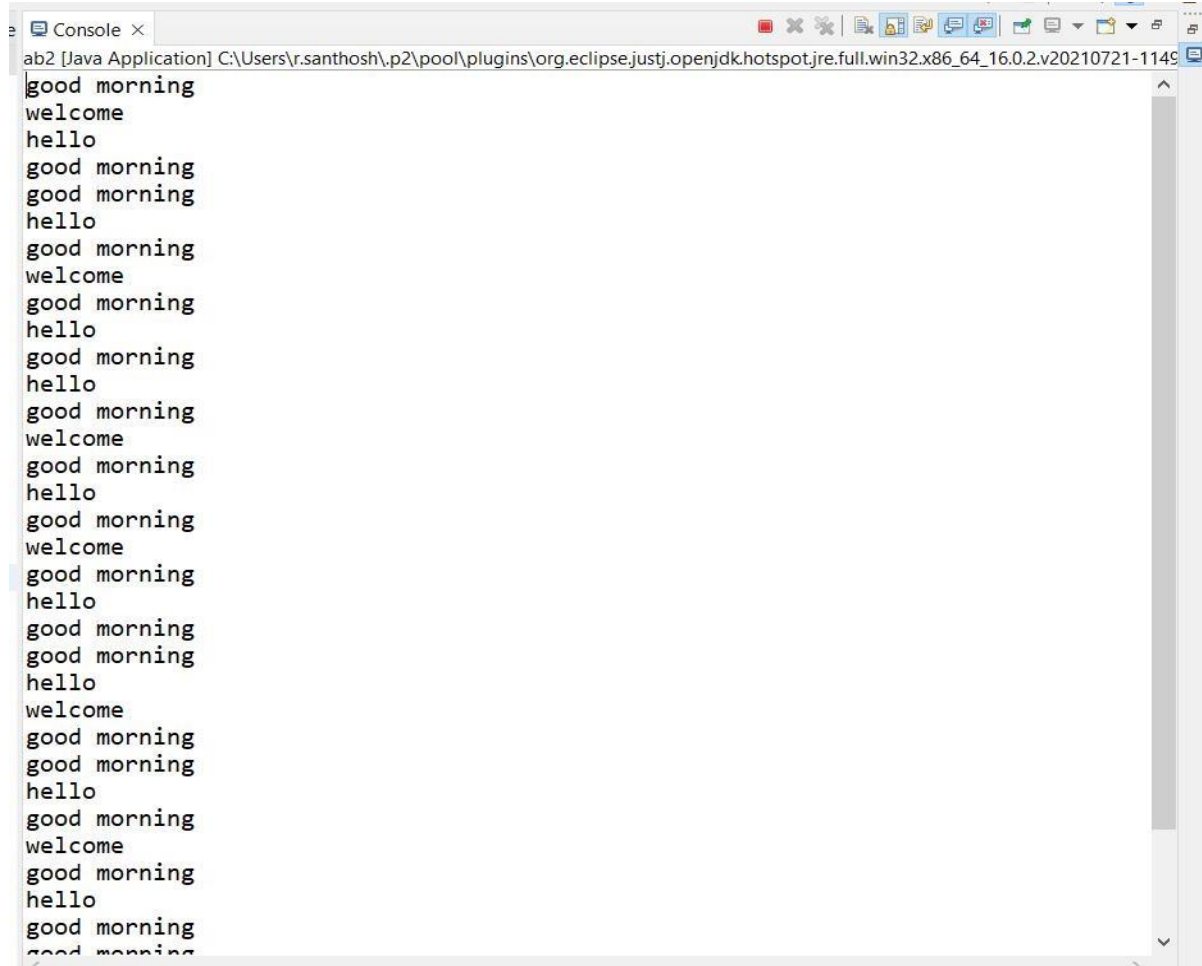
```
<terminated> multi [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.j
Roll number: 131
Name: sudha
Mark1: 98
Mark2: 99
Total: 197
Game: football
```

## PROGRAM:

### // Ex.No:11. creating Threads

```
class mt implements Runnable
{
Thread t; String s; int r; mt(String ss, int tt)
{ t=new Thread(this, ss);
  s=ss; r=tt;
  t.start();
} public void run()
{
  for(;;)
  {
    System.out.println(s); try
    {
      Thread.sleep(r);
    } catch(Exception e)
    { }
  } } class ab2
{ public static void main(String ar[])
  { mt t1=new mt("good morning",1000); mt t2=new mt("hello",2000); mt
    t3=new mt("welcome",3000);
  } }
```

## OUTPUT :



The screenshot shows the Eclipse IDE's console window. The title bar reads 'Console X' and the address bar shows the path 'ab2 [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86\_64\_16.0.2.v20210721-1149'. The console output consists of a repeating sequence of three lines: 'good morning', 'welcome', and 'hello'. This sequence is repeated 10 times, with the final line 'good morning' being partially cut off at the bottom of the visible area. The text is displayed in a monospaced font on a light gray background.

```
good morning
welcome
hello
good morning
good morning
hello
good morning
welcome
good morning
hello
good morning
hello
good morning
welcome
good morning
hello
good morning
welcome
good morning
hello
good morning
welcome
good morning
hello
good morning
good morning
hello
welcome
good morning
good morning
hello
good morning
welcome
good morning
hello
good morning
good morning
```


## **PROGRAM:**

### **//Ex.No:12. User Defined Error Handling**

```
class dividerdemo {
public static void main(String args[])
{
    try
    {
        int a=Integer.parseInt(args[0]); int b=Integer.parseInt(args[1]);
        System.out.println("Quotient: "+a/b);
    }
    catch(ArithmeticException e)
    {
        System.out.println("Error in denominator");
    }
    catch(ArrayIndexOutOfBoundsException e)
    {
        System.out.println("Error in index value");
    }
    catch(NumberFormatException n)
    {
        System.out.println("Data type error.");
    }
    finally
    {
        System.out.println("Finally block.");
    }
}
}
```



## OUTPUT :



The screenshot shows a console window titled "Console" with a close button. The text inside the console is as follows:

```
<terminated> cal1 [Java Application] C:\Users\r.santhosh\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_
Error in index value
Finally block.
```

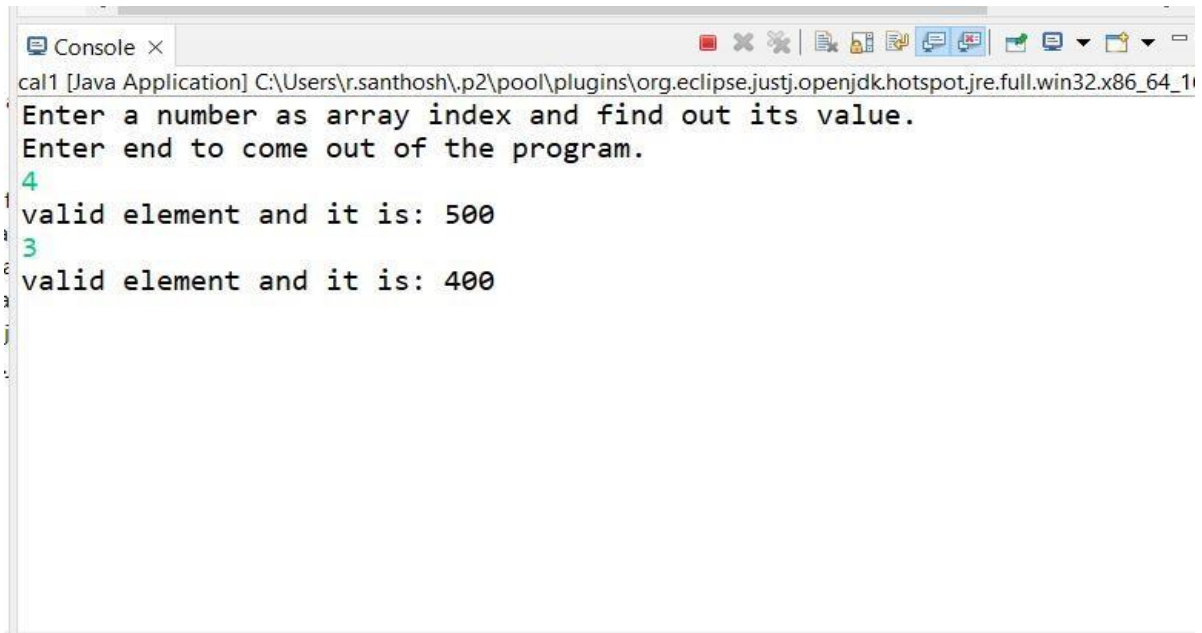
## PROGRAM:

**/\*Ex.No:13. ArrayIndexOutOfBoundsException  
andNumberFormatException\*/**

```
import java.io.*; class catchexcept
{ public static void main(String args[])
    {
        int arr[]= {100, 200, 300, 400, 500};
        System.out.println("Enter a number as array index and find out its value.");
        System.out.println("Enter end to come out of the program."); try
        {
            String    line;
            int x;
            BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
            while((line=br.readLine())!=null)
            {
                if(line.equals("end"))
                    break;
                else
                {
                    try
                    {
                        x=Integer.parseInt(line);
                        System.out.println("valid element and it is: "+arr[x]);
                    }
                    catch(ArrayIndexOutOfBoundsException e)
                    {
                        System.out.println("Invalid Element");
                        System.out.println("Exception Guarenteed: "+e);
                    }
                    catch(NumberFormatException n)
                    {
                        System.out.println("Sorry no characters.");
                        System.out.println("Generated Exception: "+n);
                    }
                }
            } //end of while
        } //end of else
    }
```

```
    }  
    catch(IOException i)  
    {} } //end of main}
```

## OUTPUT :



The screenshot shows a Java console window titled "Console x". The window contains the following text:

```
cal1 [Java Application] C:\Users\r.santhosh\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_11
Enter a number as array index and find out its value.
Enter end to come out of the program.
4
valid element and it is: 500
3
valid element and it is: 400
```

The console window has a standard Windows-style title bar with minimize, maximize, and close buttons. The text is displayed in a monospaced font, typical of a terminal or console application.

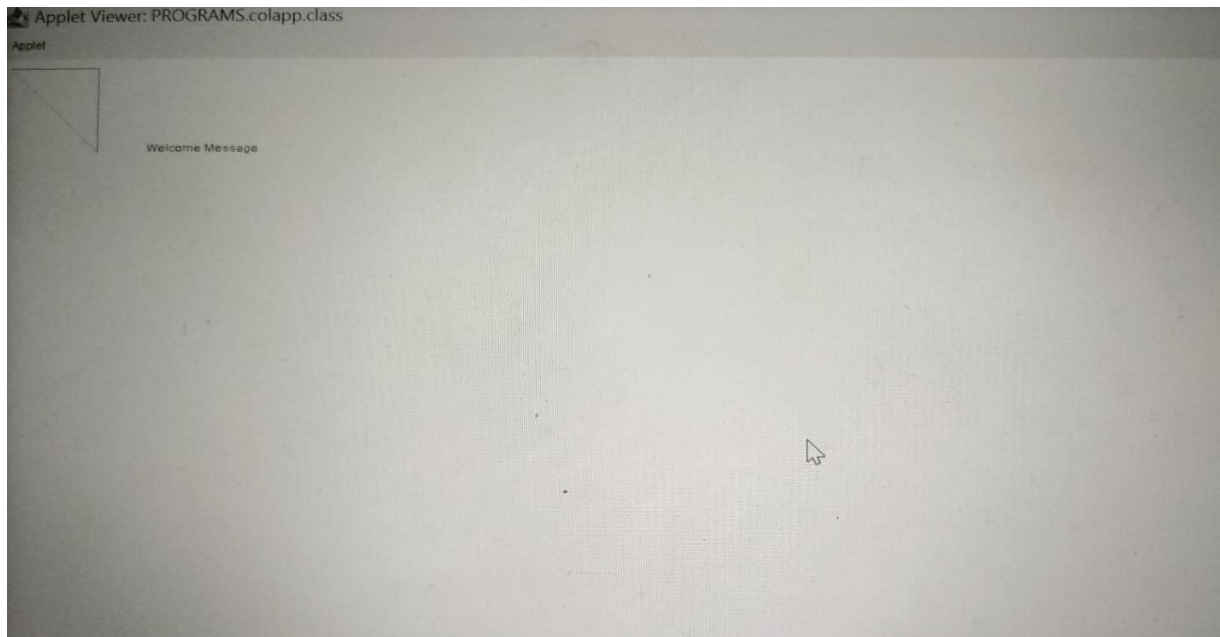
## **PROGRAM:-**

**// Ex.No:14. Applet that displays a simple message.**

```
import java.awt.*; import java.applet.*;

/*<applet code="colapp" width=200 height=200>
</applet>*/ public class colapp extends Applet
{ public void paint(Graphics g)
{
String s="Welcome Message"; g.drawString(s,150,100);
g.setColor(Color.red);
g.drawLine(10,10,100,10);
g.setColor(Color.blue);
g.drawLine(100,10,100,100);
g.setColor(Color.green);
g.drawLine(10,10,100,100);
}
}
```

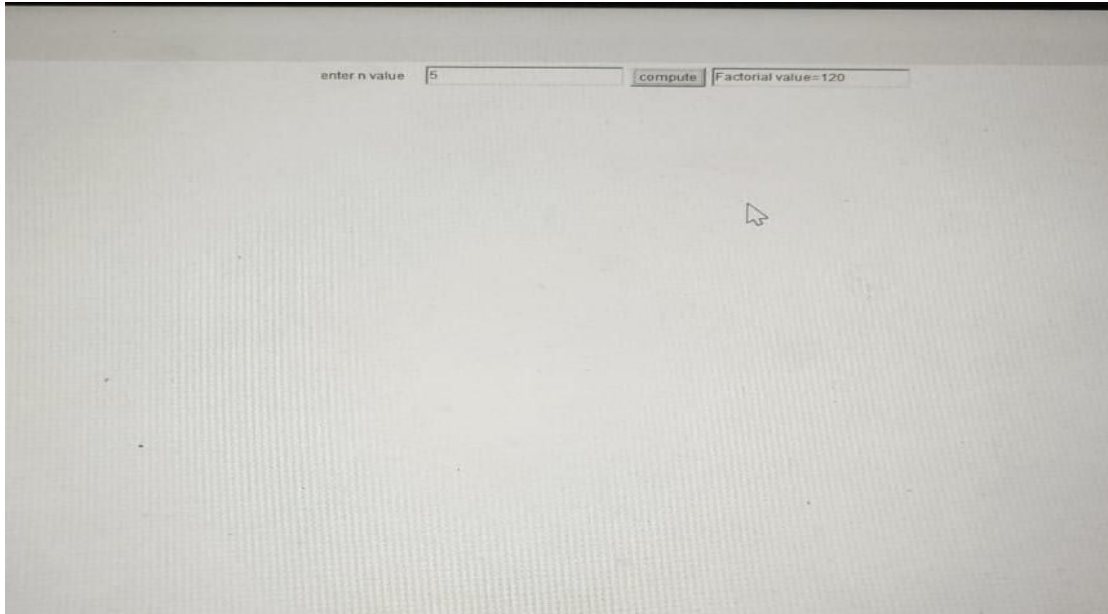
## OUTPUT :



## PROGRAM:

```
/* Ex.No:15. Factorial value using Applet */ import java.awt.*;
import java.awt.event.*; import java.applet.*; public class fact extends
Applet implements ActionListener
{ int n; TextField t1, t2;
Label l1; Button b; public
void init()
{ l1=new Label("enter n value",Label.LEFT); t1=new
    TextField(20); b=new Button("compute"); t2=new
    TextField(20); add(l1); add(t1); add(b); add(t2);
    b.addActionListener(this);
}
public void actionPerformed(ActionEvent ae)
{
    String s1=t1.getText(); int f=1; int
    n=Integer.parseInt(s1); for(int
    i=1;i<=n;i++) f=f*i;
    String s="Factorial value="+f; t2.setText(s);
}
}
/*<applet code="fact" width=300 height=300></applet>*/
```

## OUTPUT :





## **PROGRAM:**

**/\* Ex.No:16. Draw lines, rectangles and ovals \*/**

```
import java.awt.*; import
java.applet.Applet;
public class DrawShapes extends Applet
{ public void paint(Graphics g)
    {
        g.drawLine(40,30,200,30);
        g.drawRect(40,60,70,40);
        g.fillRect(140,60,70,40);
        g.drawOval(40,120,70,40);
        g.fillOval(140,120,70,40);
    }
}
/*<applet code="DrawShapes.class" height=500 width=500></applet>*/
```

## OUTPUT :

