Print the largest and the smallest numbers in a group.

```
class Sort
{
      static int arr[]=new int [5];
      static int i,j,temp;
      public static void main(String args[])
      {
            for(i=0;i<5;i++)
                  arr[i]=Integer.parseInt(args[i]);
                  for(i=0;i<5;i++)
                  for(j=i+1;j<5;j++)
                  if(arr[i]<arr[j])
            {
                  temp=arr[i];
                  arr[i]=arr[j];
                  arr[j]=temp;
            }
            System.out.println("The greatest and least numbers are
            "+arr[0]+"   "+arr[4]);
      }
}
```

Input: java sort 5 7 4 3 6

Output: The greatest and the least numbers are 7   3

Determine if first is a multiple of the second.

```
class multiple{
      public static void main(String args[])
      {
            int i, j;

            i = Integer.parseInt(args[0]);
            j = Integer.parseInt(args[1]);

            if(i % j == 0)
            {
                  System.out.println(i+" is a multiple of "+j);
            }
            else
                  System.out.println(i+" is not a multiple of "+j);
      }
}
```

Input: java multiple 10 5

Output: 10 is a multiple of 5

Concatenate the first and the last name with a space.

```
Class namecat{
      public static void main(String args[])
      {
            String lname, fname;

            lname = args[0];
            fname = args[1];

            fname = fname + " " + lname;

            System.out.println("The full name is..."+fname);
      }
}
```

Output: The full name is. . . vrishal kumar

Find out if three numbers form the sides of a rt. Triangle

```
class Pythogorean
{
      public static void main(String args[])
      {
            int a=Integer.parseInt(args[0]);
            int b=Integer.parseInt(args[1]);
            int c=Integer.parseInt(args[2]);

      if(((a*a)==(b*b)+(c*c))||((b*b)==(c*c)+(a*a))||((c*c)==(a*a)+(b*b)
))
      System.out.println("They can be sides of a rt angled triangle.");

      else
      System.out.println("They cannot be the sides of a rt angled
triangle.");
      }
}
```

Output: rttri 12 13 5

They can be sides of a rt angled triangle.

Implementing binary search

```
class Binary
{
      static int  Arr[]=new int[5];
      static      int E;
      static int binSrch(int l,int h,int E){
            while(l<h){
            int m=(l+h)/2;
                  if(E==Arr[m])
                  return m;
            else if(E<Arr[m])
                  h=m-1;
            else if(E>Arr[m])
                  l=m+1;
            }
            return 0;
            }
      static void sort(int l,int h){
            int temp=0,i,j=0;
            for(i=l;i<=h;i++)
                  for(j=i+1;j<=h;j++)
                   if(Arr[i]>Arr[j])
            {
                        temp=Arr[i];
                        Arr[i]=Arr[j];
                        Arr[j]=temp;
            }
      }

public static void main(String args[]){
      for(int i=0;i<5;i++)
            Arr[i]=Integer.parseInt(args[i]);
            E=Integer.parseInt(args[5]);
            sort(0,4);
            System.out.println("Elements in ascending order:");
            for(int i=0;i<5;i++)
                  System.out.print(Arr[i]+" ");
            System.out.print("\n");
            int b=binSrch(0,4,E);
            if (b==0)
                  System.out.println(E+" is not in the list.");
            else
                  System.out.println(E+" is in the "+(b+1)+"
position.");
      }
}
```

Output: binsea 5 6 4 3 7 4

      4 is in the list.

Perform arithmetic on complex numbers

```
class Complex
{
      int a1,a2,b1,b2,c1=0,c2=0;

      void compAdd(Complex cmp){
            c1=a1+a2;
            c2=b1+b2;
      }

      void compSub(Complex cmp){
            c1=a1-a2;
            c2=b1-b2;
      }

      void compMul(Complex cmp){
            c1=(a1*a2-b1*b2);
            c2=(a1*b2+a2*b1);
      }
}

class New
{
      public static void main(String args[]){
            Complex cmp=new Complex();
            cmp.a1=Integer.parseInt(args[0]);
            cmp.b1=Integer.parseInt(args[1]);
            cmp.a2=Integer.parseInt(args[2]);
            cmp.b2=Integer.parseInt(args[3]);
      System.out.println("First Number   : "+cmp.a1+"+"+cmp.b1+"i");
      System.out.println("Second Number  : "+cmp.a2+"+"+cmp.b2+"i");

      cmp.compAdd(cmp);
      System.out.println("Addition       : "+cmp.c1+"+"+cmp.c2+"i");

      cmp.compSub(cmp);
      System.out.println("Subtraction    : "+cmp.c1+"+"+cmp.c2+"i");

      cmp.compMul(cmp);
      System.out.println("Multiplication : "+cmp.c1+"+"+cmp.c2+"i");
      }
}
```

Output:

```
First Number     : 4 + 6i
Second Number    : 3 + 7i
Addition         : 7 + 13i
Subtraction      : 1 – 1i
```

Implementing various forms of inheritence

```
//Simple Inheritance

class SuperClass
{
   int a;
   SuperClass(int i)
   {
      a=i;
   }

   void display()
   {
      System.out.println("a= "+a);
   }
}

class SubClass extends SuperClass
{
   int b;
   SubClass(int a,int i)
   {
      super(a);
      b=i;
   }

   void display()
   {
      super.display();
      System.out.println("b= "+b);
   }
}

class  Simple
{
   public static void main(String args[])
   {
      SubClass ob=new SubClass(10,30);
      ob.display();
   }
}
```

Output:

```
            a = 10
            b = 30
```

```
      // Multilevel Inheritance

class SuperClass
{
   int a;
   SuperClass(int i)
   {
      a=i;
   }

   void display()
   {
      System.out.println("a= "+a);
   }
}
class SubClass1 extends SuperClass
{
   int b;
   SubClass1(int a,int i)
   {
      super(a);
      b=i;
   }

   void display()
   {
      super.display();
      System.out.println("b= "+b);
   }
}

class SubClass2 extends SubClass1
{
   int c;
   SubClass2(int a,int b,int i)
   {
      super(a,b);
      c=i;
   }
   void display()
   {
      super.display();
      System.out.println("c= "+c);
   }
}

class Multilevel
{
   public static void main(String args[])
   {
      SubClass2 ob=new SubClass2(10,20,30);
      ob.display();
   }
}
```

Output:

```
            a = 10
            b = 20
            c = 30
```

```
//    Hierarchial Inheritance

class A
{
   int a;
   A(int a)
   {
      this.a=a;
   }

   void show()
   {
      System.out.println("a= "+a);
   }
}

class B extends A
{
   int b;
   B(int a,int b)
   {
      super(a);
      this.b=b;
   }

   void show()
   {
      super.show();
      System.out.println("b= "+b);
   }
}
class C extends A
{
   int c;
   C(int a,int c)
   {
      super(a);
      this.c=c;
   }

   void show()
   {
      super.show();
      System.out.println("c= "+c);
   }
}
class Hierarchial
{
   public static void main(String args[])
   {
      B b=new B(10,20);
      C c=new C(30,40);
      b.show();
      c.show();
   }
}
```

Output:

```
            a = 10
            b = 20
            c = 30
            d = 40
```

Types of polymorphism

```
//Function overloading

class Areas
{
int area(int b,int h){
            return (b*h/2);
      }

float area(float l,float b){
            return (l*b);
      }

float area(float s){
            return (s*s);
      }

int area(int r){
            return (22*r*r/7);
      }
}

class Sample
{
      public static void main(String args[]){
            int intres=0;
            float fltres=0.0f;

            Areas area=new Areas();

            intres=area.area(8,9);
            System.out.println("Area of the triangle  : "+intres);

            intres=area.area(14);
            System.out.println("Area of the circle    : "+intres);

            fltres=area.area(6.4f,3.6f);
            System.out.println("Area of the rectangle : "+fltres);

            fltres=area.area(0.8f);
            System.out.println("Area of the square    : "+fltres);
      }
}
```

Output:

```
      Area of the triangle  : 36
      Area of the circle    : 616
      Area of the rectangle : 23.039999
      Area of the square    : 0.64000005
```

```
	// Function overriding

class A
{
	void callMe(){
		System.out.println("Inside A's callMe method.");
	}
}

class B extends A
{
	void callMe(){
		System.out.println("Inside B's callMe method.");
	}
}

class C extends A
{
	void callMe(){
		System.out.println("Inside C's callMe method.");
	}
}

class DynPoly
{
	public static void main(String args[]){
		A a=new A();
		B b=new B();
		C c=new C();

		A r;			//a reference of type A.

		r=a;
		r.callMe();
		r=b;
		r.callMe();
		r=c;
		r.callMe();
	}
}
```

Output:

```
	Inside A's callMe method.
	Inside B's callMe method.
	Inside C's callMe method.
```

Use of interfaces to implement multiple inheritence.

```
interface A
{
  static final int a=10;
  void display_a();
}

interface B
{
  static final int b=20;
  void display_b();
}

class C implements A,B
{
  public void display_a()
  {
    System.out.println("a= "+a);
  }

  public void display_b()
  {
    System.out.println("b= "+b);
  }
}

class DemoMultipleInheritance
{
  public static void main(String args[])
  {
     C ob=new C();
     ob.display_a();
     ob.display_b();
  }
}
```

Output:

```
a= 10
b= 20
```

Implementation of an abstract class.

```
abstract class Ac{
      abstract public int multiply(int x,int y);
      abstract public int divide(int x,int y);
}

class Ac1 extends Ac{
      public int multiply(int x,int y){
            return (x*y);
      }
      public int divide(int x,int y){
            return (x/y);
      }
}

 class Abstr{
       public static void main(String arg[]){
             Ac1 acob;
             acob=new Ac1();

             int a,b;
             a=acob.multiply(12,13);
             b=acob.divide(20,4);

             System.out.println("A="+a+" B="+b);
       }
 }
```

Output:

```
      A = 156
      B = 5
```

Creation and implementation of a package.

```
package mypack;

public class Volumes
{
	public float cubeVol(float s){
		return (s*s*s);
	}

	public float cuboidVol(float l,float b,float h){
		return (l*b*h);
	}

	public float sphereVol(float r){
		return (88*r*r*r/21);
	}
}


	// Using the above package.

import mypack.*;

class PackDemo
{
	public static void main(String args[]){
		Volumes vol=new Volumes();

		float  volume=vol.cubeVol(4.4f);
			System.out.println("Volume of a cube : "+volume);

		volume=vol.sphereVol(21.0f);
			System.out.println("Volume of a sphere : "+volume);
	}
}
```

Using all keywords of exception handling.

```
class UseException
{
	public static void main(String args[])
	{
		try
		{
			throwException();
		}

		catch(Exception e)
		{
			System.out.println("Exception handled in main");
		}
		doesNotThrowException();
	}

	public static void throwException() throws Exception
	{
		try
		{
			System.out.println("Method throw Exception");
			throw new Exception();
		}

		catch(Exception e)
		{
			System.out.println("Handled in method throw
exception");
			throw e;
		}

		finally
		{
			System.out.println("Finally is always executed");
		}
	}

	public static void doesNotThrowException()
	{
		try
		{
			System.out.println("Method does not throw exception");
		}
		catch(Exception e)
		{
			System.out.println(e.toString());
		}
		finally
		{
			System.out.println("Finally is always executed");
		}
	}
}
```

Output:

```
	Method throw Exception
	Handled in method throw exception
	Finally is always executed
	Exception handled in main
	Method does not throw exception
	Finally is always executed
```

Creating and running multiple threads.

```
class A extends Thread
 {
       public void run(){
             for(int i=0;i<=5;i++)
                   System.out.println("From Thread A : i = "+i);
             System.out.println("Exit from Thread A.");
       }
 }

 class B extends Thread
 {
       public void run(){
             for(int i=0;i<=3;i++)
                   System.out.println("From Thread B : i = "+i);
             System.out.println("Exit from Thread B.");
       }
 }

 class C extends Thread
 {
       public void run(){
             for(int i=0;i<=5;i++)
                   System.out.println("From Thread C : i = "+i);
             System.out.println("Exit from Thread C.");
       }
 }

 class Multi_thread
 {
       public static void main(String args[]){
       new A().start();
       new B().start();
       new C().start();
       }
 }
```

Output:

```
      From Thread A : i = 0
      From Thread B : i = 0
      From Thread C : i = 0
      From Thread A : i = 1
      From Thread B : i = 1
      From Thread C : i = 1
      From Thread A : i = 2
      From Thread B : i = 2
      From Thread C : i = 2
      From Thread A : i = 3
      From Thread B : i = 3
      From Thread C : i = 3
      From Thread A : i = 4
            Exit from Thread B.
      From Thread C : i = 4
      From Thread A : i = 5
      From Thread C : i = 5
            Exit from Thread A.
            Exit from Thread C.
```

Demonstrate high priority thread using sleep.

```
class A extends Thread
 {
      public void run(){
             System.out.println("Thread A started.");
             for(int i=0;i<=3;i++){
                   System.out.println("From Thread A : i = "+i);
                   try{
                         Thread.sleep(2000);
                   }catch(Exception e){};
             }
             System.out.println("Exit from Thread A.");
       }
 }

 class B extends Thread
 {
      public void run(){
             System.out.println("Thread B started.");
             for(int i=0;i<=3;i++)
                   System.out.println("From Thread B : i = "+i);
                   System.out.println("Exit from Thread B.");
       }
 }

 class C extends Thread
 {
        public void run(){
             System.out.println("Thread C started.");

             for(int i=0;i<=3;i++)
                   System.out.println("From Thread C : i = "+i);

             System.out.println("Exit from Thread C.");
       }
 }

 class thread_priori
 {
       public static void main(String args[]){
             A thA=new A();
             B thB=new B();
             C thC=new C();

             thA.setPriority(Thread.MAX_PRIORITY);
             thB.setPriority(thA.getPriority()-1);
             thC.setPriority(Thread.MIN_PRIORITY);

      System.out.println("Start Thread A.");
            thA.start();

      System.out.println("Start Thread B.");
            thB.start();

      System.out.println("Start Thread C.");
            thC.start();

      System.out.println("End of Main Thread.");

      }
 }
```

Output:

```
Start Thread A.
Thread A started.
From Thread A : i = 0
Start Thread B.
Thread B started.
From Thread B : i = 0
From Thread B : i = 1
From Thread B : i = 2
From Thread B : i = 3
Start Thread C.
Exit from Thread B.
End of Main Thread.
Thread C started.
From Thread C : i = 0
From Thread C : i = 1
From Thread C : i = 2
From Thread C : i = 3
Exit from Thread C.
```

Implementation of a stack.

```
import java.io.*;

interface St
{
  void push(int x);
  int pop();
  void dis();
}

class Stck implements St
{
  private int s[];
  private int top;
  Stck(int size)
  {
    s=new int[size];
    top=-1;
  }

  public void push(int item)
  {
    if(top==s.length-1)
      System.out.println("stack is full");
    else
      s[++top]=item;
  }

  public int pop()
  {
    if(top==-1)
    {
      System.out.println("stack is empty");
      return(-1);
    }
    else
       return(s[top--]);
  }

  public void dis()
  {
    for(int i=top;i>=0;i--)
       System.out.print(s[i]+"\t");
    System.out.println(" ");
  }
}

class Stack
{
  public static void main(String args[])
  {
    try
    {
      DataInputStream in=new DataInputStream(System.in);
      Stck ob=new Stck(5);
      int ch,i,t;
      System.out.println("1.push  2.pop  3.display  4.exit");
      ch=Integer.parseInt(in.readLine());
      while(ch<4)
      {
            switch(ch)
            {
              case 1:
```

```
                System.out.println("enter an element to push");
                i=Integer.parseInt(in.readLine());
                ob.push(i);
                break;
            case 2:
                t=ob.pop();
                if(t==-1)
                   break;
                else
                  System.out.println("poped element is "+t);
                break;
            case 3:
                ob.dis();
                break;
            }
          System.out.println("1.push  2.pop  3.display  4.exit");
          ch=Integer.parseInt(in.readLine());
       }
   }

catch(Exception e)
{
      System.out.println("Exception "+e);
}
}
}
```

Output:

```
1.push  2.pop  3.display  4.exit
1

enter an element to push
10

1.push  2.pop  3.display  4.exit
1

enter an element to push
20

1.push  2.pop  3.display  4.exit
2
poped element is 20

1.push  2.pop  3.display  4.exit
3
10

1.push  2.pop  3.display  4.exit
4
```

Implementation of a queue.

```
public class QueueApplication {
    Object elements[];
    int frontOfQueue,rareOfQueue;
    int size;
    public QueueApplication(int n) {
        frontOfQueue = 0;
        rareOfQueue = -1;
        size = n;
        elements = new Object[n];
    }

    public static void main (String args[]) {
        QueueApplication queue;
        queue = create();
        String string;
        queue.insert( new String("first" ));
        queue.insert( new String( "second"));
        string = (String)queue.getElement();
        System.out.println("front ele is "+ string);
        queue.delete();
        string = (String)queue.getElement();
        System.out.println("front ele is "+ string);
        queue.delete();
        queue.delete();
    }

    public static QueueApplication create()    {
        return (new QueueApplication(100) );
    }
    public void insert( Object obj )  {
        if( rareOfQueue > size-1)
            System.out.println("Queue overflow" );
        else
            elements[++rareOfQueue] = obj;
    }
    public Object delete( )    {
        if( rareOfQueue <= -1 || rareOfQueue < frontOfQueue)  {
            System.out.println( "Queue is empty" );
            return null;
        }
        else{
            System.out.println("deleteped.." +elements[frontOfQueue] );
            return elements[frontOfQueue++];
        }
    }

    public Object getElement()  {
        if( rareOfQueue <= -1 || rareOfQueue < frontOfQueue)  {
            System.out.println( "Queue is empty" );
            return null;
        }
        else
            return elements[frontOfQueue];
    }
}
```

Output:

```
front ele is first
deleteped..first
front ele is second
deleteped..second
Queue is empty
```

draw a border layout

```
import javax.swing.*;
import java.awt.*;
public class BorderLay extends JApplet
{
      Container c;
      BorderLayout bl;
      JButton b1,b2, b3, b4, b5;
      public void init()
      {
            c  =  getContentPane();
            bl  =  new BorderLayout();
            c.setLayout(bl);
            b1  =  new JButton("C");
            b2  =  new JButton("C++");
            b3  =  new JButton("Java");
            b4  =  new JButton("Pascal");
            b5  =  new JButton("SmallTalk");
            c.add(b1,BorderLayout.EAST);
            c.add(b2,BorderLayout.WEST);
            c.add(b3,BorderLayout.NORTH);
            c.add(b4,BorderLayout.SOUTH);
            c.add(b5,BorderLayout.CENTER);
      }
};
//<applet code = BorderLay width = 300 height = 300></applet>
```

Output:

```
                  Java
      C++                         C
                  SmallTalk
```

To demonstrate the box layout

```
import javax.swing.*;
import java.awt.*;
public class BoxLay extends JApplet
{
      Container c;
      BoxLayout b1;
      JCheckBox cb1,cb2;
      JRadioButton rb1,rb2;
      JPanel p1,p2,p3;
      public void init()
      {
            c  =  getContentPane();
            p1  =  new JPanel();
            p2  =  new JPanel();
            p3  =  new JPanel();
            cb1  =  new JCheckBox("Windows 2003");
            cb2  =  new JCheckBox("Windows XP");
            rb1  =  new JRadioButton("McLaren Mercedes");
            rb2  =  new JRadioButton("Ferrari FXT90");
            p1.add(cb1);p1.add(cb2);
            p2.add(rb1);p2.add(rb2);
            p3.add(p1);p3.add(p2);
            b1  =  new BoxLayout(p3,BoxLayout.Y_AXIS);
            p3.setLayout(b1);
            c.add(p3);
      }
};
//<applet code = BoxLay width = 400 height = 400></applet>
```

Output:

Windows 2003 Windows XP

McLaren Mercedes Ferrari FXT90

To show the card layout

```
import javax.swing.*;
import java.awt.*;
public class Card extends JApplet
{
      JPanel p1,p2,p3;
      Container c;
      JCheckBox cb1,cb2;
      JRadioButton rb1,rb2;
      CardLayout  cl;
      public void init()
      {
            p1 = new JPanel();
            cl = new CardLayout();
            p1.setLayout(cl);
            p2 = new JPanel();
            p3 = new JPanel();
            cb1 = new JCheckBox("Windows 98");
            cb2 = new JCheckBox("Windows NT");
            rb1 = new JRadioButton("RedHat Linux");
            rb2 = new JRadioButton("Mandrake Linux");
            p2.add(cb1);p2.add(cb2);
            p3.add(rb1);p3.add(rb2);
            p1.add(p3,"Linux");//Linux comes first
            p1.add(p2,"Windows");
            c = getContentPane();
            c.add(p1);cl.first(p1);
      }
};
//<applet code = Card width = 400 height = 400></applet>
```

To demonstrate Client authentication

```
import java.net.*;
import java.io.*;
public class Client
{
      public static void main(String args[]) throws IOException
      {
            Socket s = null;
            BufferedReader b = null;
            try
            {
                  s = new Socket(InetAddress.getLocalHost(),98);
                  b = new BufferedReader(new
                  InputStreamReader(s.getInputStream()));
            }
            catch(UnknownHostException u)
            {
                  System.out.println("I dont know the host");
                  System.exit(0);
            }
            String inp;
            while( (inp = b.readLine())! = null);
            {
                  System.out.println(inp);
            }
            b.close();
            s.close();
      }
};
```

To copy a file

```
import java.io.*;
class Copyfile
{
	public static void main(String args[]) throws IOException
	{
		int i;
		FileInputStream fin;
		FileOutputStream fout;
		try
		{
			//open file input
			try
			{
				fin=new FileInputStream(args[0]);
			}
			catch(FileNotFoundException e)
			{
				System.out.println("Input File Not Found");
				return;
			}
			//open output file
			try
			{
				fout=new FileOutputStream(args[1]);
			}
			catch(FileNotFoundException e)
			{
				System.out.println("Error Opening Output File");
				return;
			}
		}
		catch(ArrayIndexOutOfBoundsException e)
		{
			System.out.println("Usage:CopyFile From To");
			return;
		}
		//CopyFile
		try
		{
			do
			{
				i=fin.read();
				if(i!=-1)
					fout.write(i);
			}while(i!=-1);
		}
		catch(IOException e)
		{
			System.out.println("FileError");
		}
		fin.close();
		fout.close();
	}
};
```

```
		Java Copyfile 1.java 2.java
```

Output:

```
		Done
```

To demonstrate the flow layout

```
import javax.swing.*;
import java.awt.*;
public class Flow extends JApplet
{
      Container c;
      FlowLayout f1;
      JButton b1,b2,b3,b4;
      public void init()
      {
            c = getContentPane();
            f1 = new FlowLayout();
            c.setLayout(f1);
            b1 = new JButton("C");
            b2 = new JButton("C++");
            b3 = new JButton("Java");
            b4 = new JButton("Pascal");
            c.add(b1);c.add(b2);c.add(b3);c.add(b4);
      }
};
//<applet code = Flow width = 300 height = 300></applet>
```

Output:

```
            C     C++   Java  SmallTalk
```

To demonstrate the use of frames

```
import java.awt.*;
import javax.swing.*;
class Framex
{
      public static void main(String args[])
      {
            String branch [] = {"CSIT","CSE","ECE","MMT","MCT"};
            Frame fr = new Frame ();
            JPanel p1 = new JPanel();
            Label l = new Label ("Enter your branch");
            TextField t = new TextField(15);
            Button b = new Button ("pom");
            List l1 = new List (2,true);
            l1.add ("Nicole Kidman");
            l1.add ("Madonna");
            JComboBox cb = new JComboBox (branch);
            Checkbox cb1 = new Checkbox ("J Lo");
            JRadioButton r = new JRadioButton ("Ben");
            TextArea ta = new TextArea ("I am an MGITian");
            Scrollbar sb = new Scrollbar (Scrollbar.HORIZONTAL, 60,
            600,0,1200);
            Choice ch = new Choice ();
            ch.addItem("Saurav");
            ch.addItem("Sachin");
            p1.add(l);        p1.add(t);
            p1.add(b);        p1.add(l1);
            p1.add(cb1);      p1.add(ta);
            p1.add(sb);       p1.add(ch);
            p1.add(cb);       p1.add(r);
            fr.add(p1);
            fr.setSize(500,500);
            fr.setVisible(true);
      }
};
```

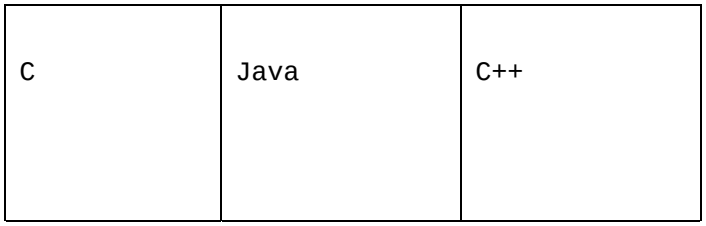
Output:

A frame is created...

To demonstrate a grid layout

```
import javax.swing.*;
import java.awt.*;
public class Grid extends JApplet
{
      Container c;
      GridLayout g1;
      JButton b1, b2, b3, b4;
      public void init()
      {
            c = getContentPane();
            g1 = new GridLayout();
            c.setLayout(g1);
            b1 = new JButton("C");
            b2 = new JButton("C++");
            b3 = new JButton("Java");
            b4 = new JButton("Pascal");
            c.add(b1);
            c.add(b2);
            c.add(b3);
            c.add(b4);
      }
};
//<applet code = Grid width = 300 height = 300></applet>
```

Output:



To demonstrate a GridBag layout

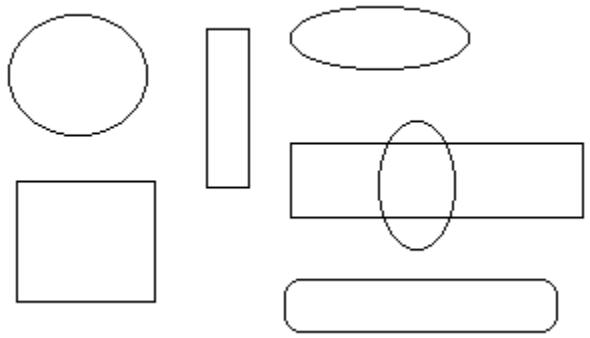
```
//<applet code = GridBag width = 400 height = 400></applet>
import javax.swing.*;
import java.awt.*;
public class GridBag extends JApplet
{
      Container c;
      JLabel l1,l2;
      JButton b1,b2;
      JTextField tf;
      JPasswordField pf;
      GridBagLayout gb1;
      GridBagConstraints gbc;
      public void init()
      {
            c = getContentPane();
            gb1 = new GridBagLayout();
            gbc = new GridBagConstraints();
            c.setLayout(gb1);
            l1 = new JLabel("User name");
            l2 = new JLabel("Password");
            b1 = new JButton("Login");
            b2 = new JButton("Reset");
            tf = new JTextField(10);
            pf = new JPasswordField(10);
            gbc.gridx = 1;
            gbc.gridy = 1;
            gb1.setConstraints(l1,gbc);
            c.add(l1);
            gbc.gridx = 2;
            gbc.gridy = 1;
            gb1.setConstraints(tf,gbc);
            c.add(tf);
            gbc.gridx = 1;
            gbc.gridy = 2;
            gb1.setConstraints(l2, gbc);
            c.add(l2);
            gbc.gridx = 2;
            gbc.gridy = 2;
            gb1.setConstraints(pf,gbc);
            c.add(pf);
            gbc.gridx = 1;
            gbc.gridy = 3;
            gb1.setConstraints(b1,gbc);
            c.add(b1);
            gbc.gridx = 2;
            gbc.gridy = 3;
            gb1.setConstraints(b2,gbc);
            c.add(b2);
      }
};
```

Output:

```
Login:    [       ]
Password: [       ]
Login       Reset
```

To draw various figures

```
import java.applet.*;
import java.awt.*;
public class Myfigure extends Applet
{
      public void paint(Graphics g)
      {
            int x[] = {250,225,240,300,280};
            int y[] = {50,70,120,120,80};
            g.drawOval(20,90,100,100);
            g.drawRect(35,105,70,70);
            g.drawRect(20,220,70,70);
            g.drawRect(20,350,70,70);
            g.drawRect(50,330,70,70);
            g.drawLine(20,350,50,330);
            g.drawLine(90,350,120,330);
            g.drawLine(20,420,50,400);
            g.drawLine(90,420,120,400);
            g.drawPolygon(x,y,5);
            g.drawLine(250,200,250,300);
            g.drawLine(350,200,350,300);
            g.drawOval(250,190,100,20);
            g.drawOval(250,290,100,20);
            g.drawRect(250,350,70,70);
            g.drawOval(251,351,69,69);
      }
};
//<applet code = Myfigure width = 500 height = 500></applet>
```



Output:

To show pushback

```
import java.io.*;
class Pushback
{
      public static void main(String args[]) throws IOException
      {
            String s = "if(a =  = 4) a = 0;\n";
            byte buf[] = s.getBytes();
            ByteArrayInputStream in = new ByteArrayInputStream(buf);
            PushbackInputStream f = new PushbackInputStream(in);
            int c;
            while((c = f.read())! = -1)
            {
                  switch(c)
                  {
                        case ' = ':
                              if((c = f.read()) =  =  ' = ')
                                    System.out.print(".eq.");
                        else
                        {
                              System.out.print("<-");
                              f.unread(c);
                        }
                        break;
                        default:
                              System.out.print((char) c);
                        break;
                  }
            }
      }
};
```

To demonstrate the use of the server function

```
import java.net.*;
import java.io.*;
public class Server
{
      public static void main(String args[])throws IOException
      {
            ServerSocket s1 = null;
            try
            {
                  s1 = new ServerSocket(98);
            }
            catch(IOException u)
            {
                  System.out.println("Could not find port 98");
                  System.exit(1);
            }
            Socket c = null;
            try
            {
                  c = s1.accept();
                  System.out.println("Connection from"+c);
            }
            catch(IOException e)
            {
                  System.out.println("Accept Failed");
                  System.exit(1);
            }
            PrintWriter out  = new
            PrintWriter(c.getOutputStream(),true);
            BufferedReader in = new BufferedReader(new
            InputStreamReader(c.getInputStream()));
            String str;
            BufferedReader br = new BufferedReader(new
            InputStreamReader(System.in));
            System.out.println("I am ready, type      new");
            while((str = br.readLine())! = null);
            {
                  out.println(str);
            }
            out.close();
            br.close();
            c.close();
            s1.close();
      }
};
```

To demonstrate the URL function

```
import java.net.*;
class Url
{
      public static void main(String args[]) throws
      MalformedURLException
      {
            URL url = new URL("http://www.rediff.com:40/downloads");
            System.out.println("Protocol:"+url.getProtocol());
            System.out.println("Port:"+url.getPort());
            System.out.println("Host:"+url.getHost());
            System.out.println("File:"+url.getFile());
            System.out.println("Ext:"+url.toExternalForm());
      }
};
```

Output:

```
      Protocol: http
      Port    : 40
      Host    : www.rediff.com
      File    : /downloads
      Ext     : http://www.rediff.com:40/downloads
```