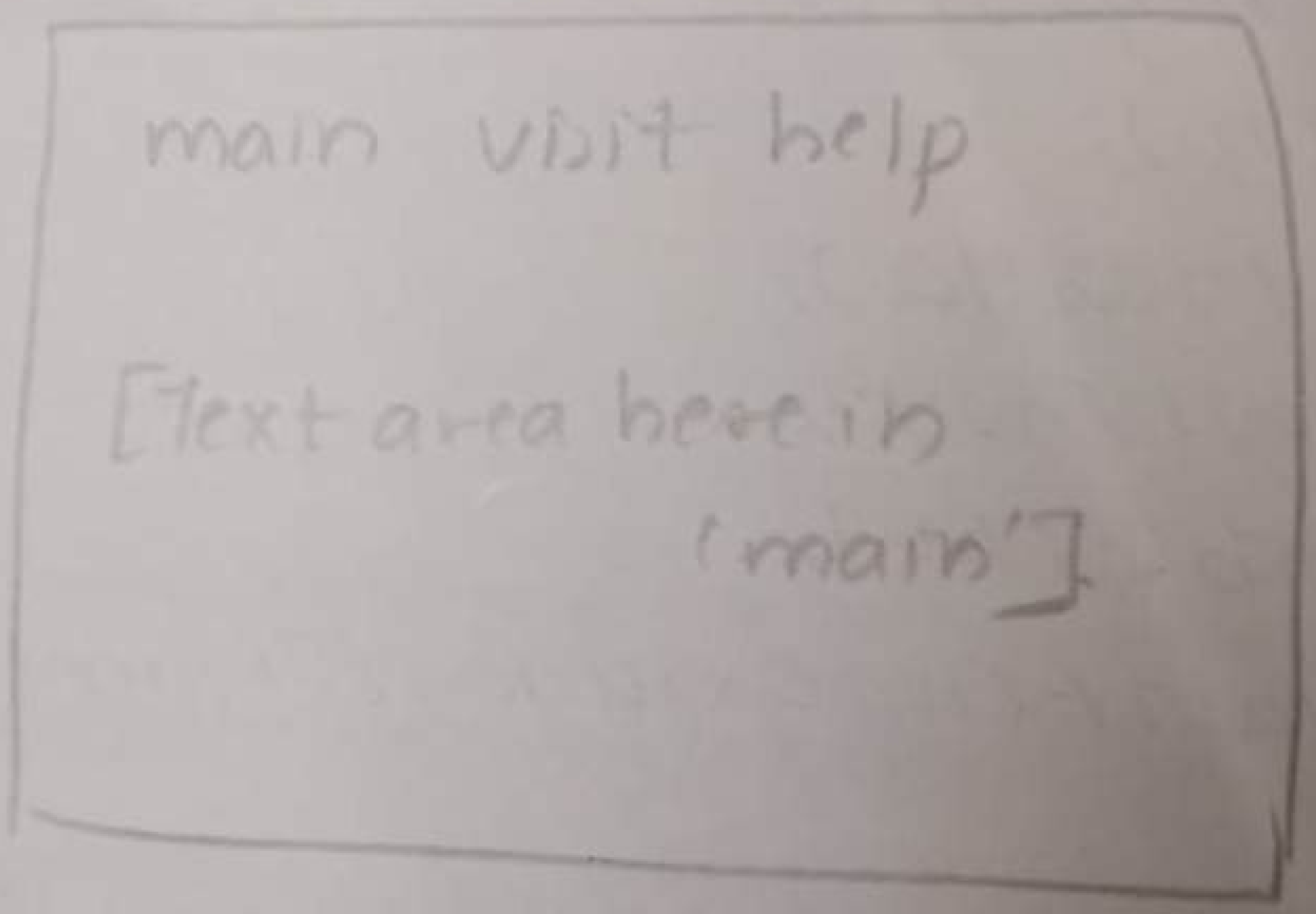
Step-6: Set the JFrame size and set setVisible(true) to make JTabbedPane visible.

Step-7: END

SOURCE CODE

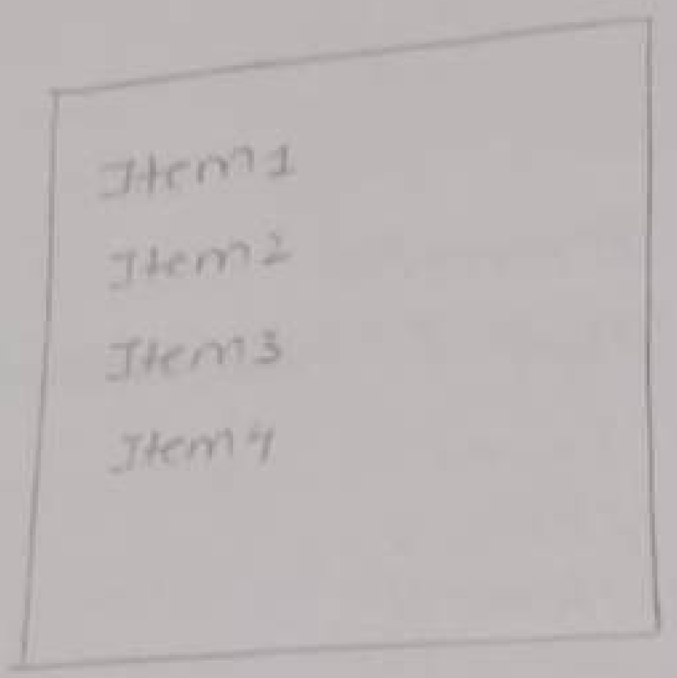
```
import javax.swing.*;
public class TabbedPaneExample{
JFrame f;
TabbedPane Example(){
    f=new JFrame();
    JTextArea -ta =new JTextArea(200,200);
    JPanel P1 =new JPanel()
     P1.add(ta);
    JPanel P2= new JPanel();
    JPanel P3=new JPanel();
    JTabbedPane tp =new JTapped Pane();
     tp.setBounds(50,50, 200, 200);
     tp.add("main",p1);
     tp.add ("visit",p2);
     tp.add ("help",p3);
       f.add(tp);
      f.setSize(400,400);
      f.setLayout(null);
      f.setVisible(true);
      f.set DefaultCloseOperation(JFrame.EXIT_ON_
                                                  CLOSE);
 }
public static void main(String[] args){
    new TabbedPaneExample();
 }}
```

OUTPUT



```
import javax.swing.*;
public class ListExmp
{
 ListExmp(){
   JFrame f=new JFrame();
   DefaultListModel<String> l1=new DefaultListModel<>();
   l1.addElement ("Item1");
   l1.addElement ("Item2");
   l1.addElement ("Item3"))
   l1.addElement ("Item4");
   JList<String>list =new  JList<>(l1);
   list.setBounds(100,100,75,75);
    f.add(list);
   f.setSize(400,400);
    fsetLayout(null);
    f.setVisible(true);
    f.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
  }
public static void main(String args[])
{
 new ListExmp();
 }
}
```

OUTPUT



```
Item1
Item2
Item3
Item4
```

2) JTree

ALGORITHM

Step-1: START

Step-2: Create a Panel and set panel Layout.

Step-3: Create JTree to the Panel.

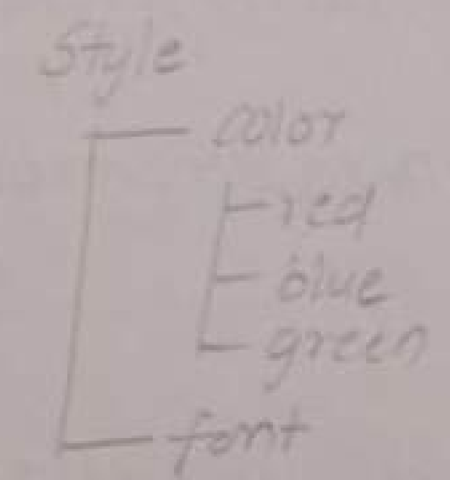
Step-4: set the properties of the JTree class

Step-5: END

SOURCE CODE

```
import javax.swing.*;
import javax.swing.tree.DefaultMutableTreeNode;
public class TreeEx
{
 JFrame f,
 TreeEx(){
    f=new JFrame();
   DefaultMutableTreeNode style=new DefaultMutableTreeNode
                                                ("style");
DefaultMutableTreeNode color=new DefaultMutableTreeNode
                                                ("color");
 DefaultMutableTreeNode font=new DefaultMutablTreeNode
                                                ("font");
 style.add(color);
 style.add(font);
DefaultMutableTreeNode red=new DefaultMutableTreeNode("red");
DefaultMutableTreeNode blue=new DefaultMutablTreeNode("blue");
DefaultMutableTreeNode green=new DefaultMutablTreeNode("green");
color.add(red);
color.add(blue);
color.add(green);
JTree jt=new JTree(style);
f.add(jt);
f.setSize(250,250);
f.setVisible(true);
f.setDefaultCloseOperation(JFrame.EXIT-ON-Close);
}
public static void main(String[] args){
      new TreeEx();
  }
}
```

OUTPUT



3) JTable

ALGORITHM

Step-1: START

Step-2: Create a Panel and set panel Layout.

Step-3: Add JTable to the Panel.

Step-4: Set the JTable rows and columns.

Step-5: Load the data into the JTable

Step-6: END

SOURCE CODE

```
import javax.swing.*;
public class TableEx
{
   JFrame f;
   Table Ex()
   {
    f=new JFrame();
    String data[][]={{"501","ravi","50000"},
                       {"502","raju","70000"},
                       {"503","ramu","90000"}};
  String column[]={"ID","NAME","SALARY"};
  JTable jt=new JTable(data,column);
  JScrollPane sp=new JScrollPane(jt);
  f.add(sp);
   f.setSize(300,400);
  f.setVisible(true);
  f.setDefaultCloseOperation(JFrame.EXIT_ON-CLOSE);
   }
public static void main(String[] args)
{ new TableEx();
  }
}
```

OUTPUT

ID	NAME	SALARY
501	ravi	50000
502	raju	70000
503	ramu	90000

VIVA QUESTIONS

1. What is the purpose of swings in java?

Ans. Swings is used to create graphical user interfaces (GUI's) in java with lightweight & flexible components.

2. What is the purpose of JList class?

Ans. The JList class is used to display a list of items from which the user can select one or more options.

3. What is the purpose of JTree class?

Ans. Used to display hierarchical data in a tree-like structure with expandable and collapsible nodes.

4. What is the purpose of JTable class?

Ans. Used to display and edit data in a table format with rows & columns in Java swing.

5. Differentiate between applet and swing.

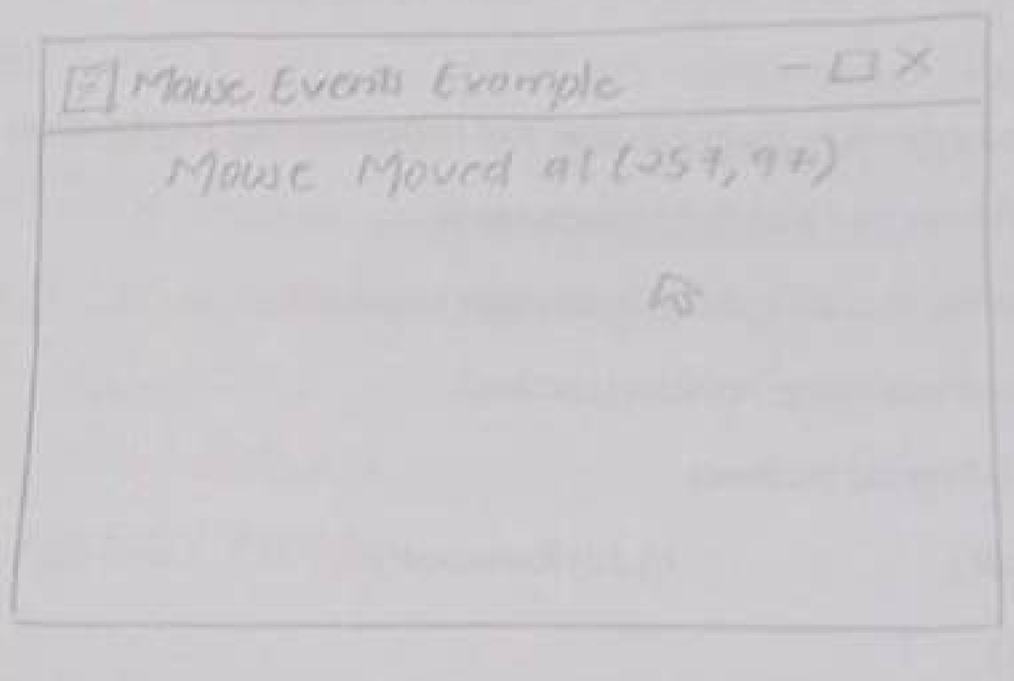
Ans. Applet used to create web based GUI programs.
Swing used to create desktop GUI applications.

Step-6: Using showStatus () method display the message. STEP 7: Display the necessary information on the screen. STEP 8: END

SOURCE CODE

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class MouseEventsExample extends JFrame
   implements MouseListener, MouseMotionListener
 {
   JLabel label;
   MouseEventsExample()
   {
    setTitle("Mouse Events Example");
    setSize( 400,300);
    setLayout(new FlowLayout());
    setDefault Close Operation(JFrame.EXIT_ON_CLOSE);
    label = new JLabel("Perform mouse actions
                          inside the window.");
    add(label);
    add MouseListener (this);
    addMouseMotionListener(this);
    set Visible(true);
   }
 public void mouseClicked(MouseEvent e){
   label.set Text("Mouse Clicked at ("+e.getX()+",
                          "+e.getY()+")");
   }
public void mouseEntered(MouseEvent e){
   labelsetText("Mouse Entered the window.");
}
public void mouseExited(MouseEvent e){
   label.setText("Mouse Exited the window.");
}
public void mousePressed(MouseEvent e){
   label.setText("Mouse Pressed at ("+e.getX()+",
                  "+e.getY()+")");
}
public void mouseReleased(MouseEvent e){
   label.setText("Mouse Released at ("+e.getX()+",
                  "+e.getY()+")");
}
public void mouseDragged(MouseEvent e){
   label.setText("Mouse Dragged at ("+e.getX()+",
              "+e.getY()+")");
}
public void mouseMoved(MouseEvent e){
   label.setText("Mouse Moved at ("+e.getX()+",
                 "+e.getY()+")");
}
public static void main(String[] args){
   new MouseEventsExample();
}
}
```

OUTPUT



```
Mouse Moved at (257,97)
```

2. KeyListener

ALGORITHM

Step-1: START

Step-2: Create a class extending from a Frame and implementing KeyListener.

Step-3: Create a TextArea and add KeyListener to it.

Step-4: Set bounds of the text area using setBounds() method.

Step-5: Set size to the frame using setSize() method.

Step-6: Override the following methods

a) keyPressed() b) keyReleased()

c) keyTyped()

Step-7: Using setText() method display the message in the text area.

Step-8: Display the necessary information on the screen.

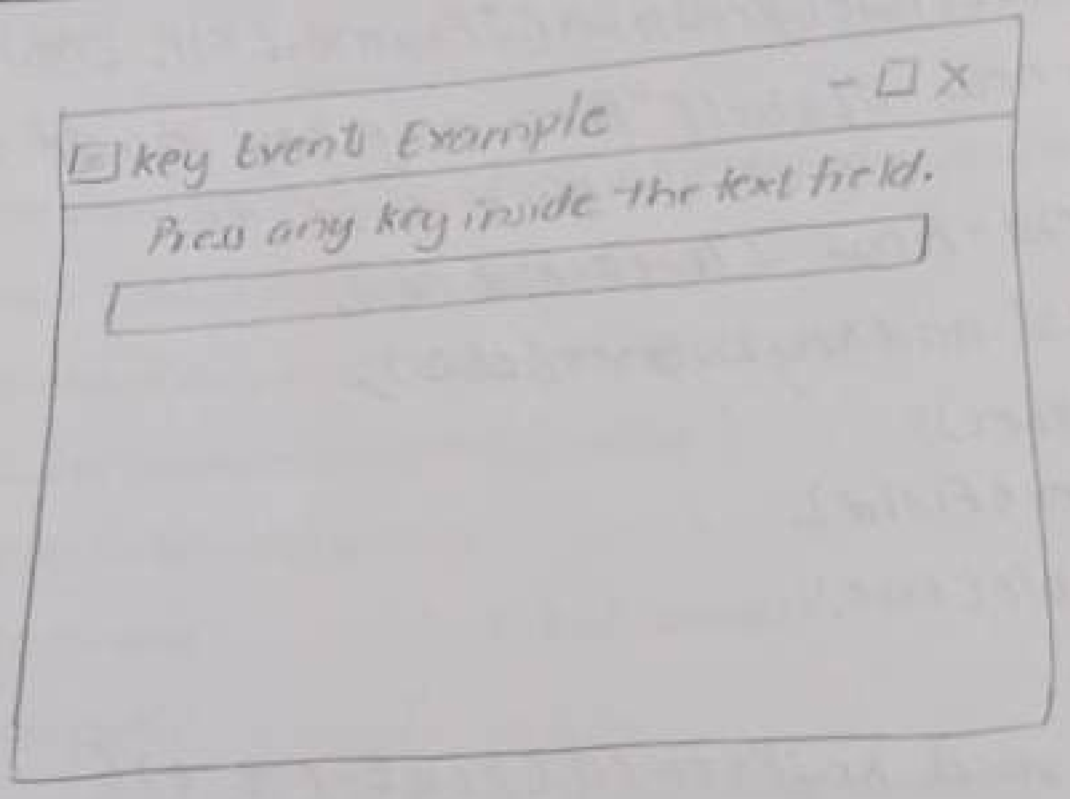
Step-9: END

SOURCE CODE

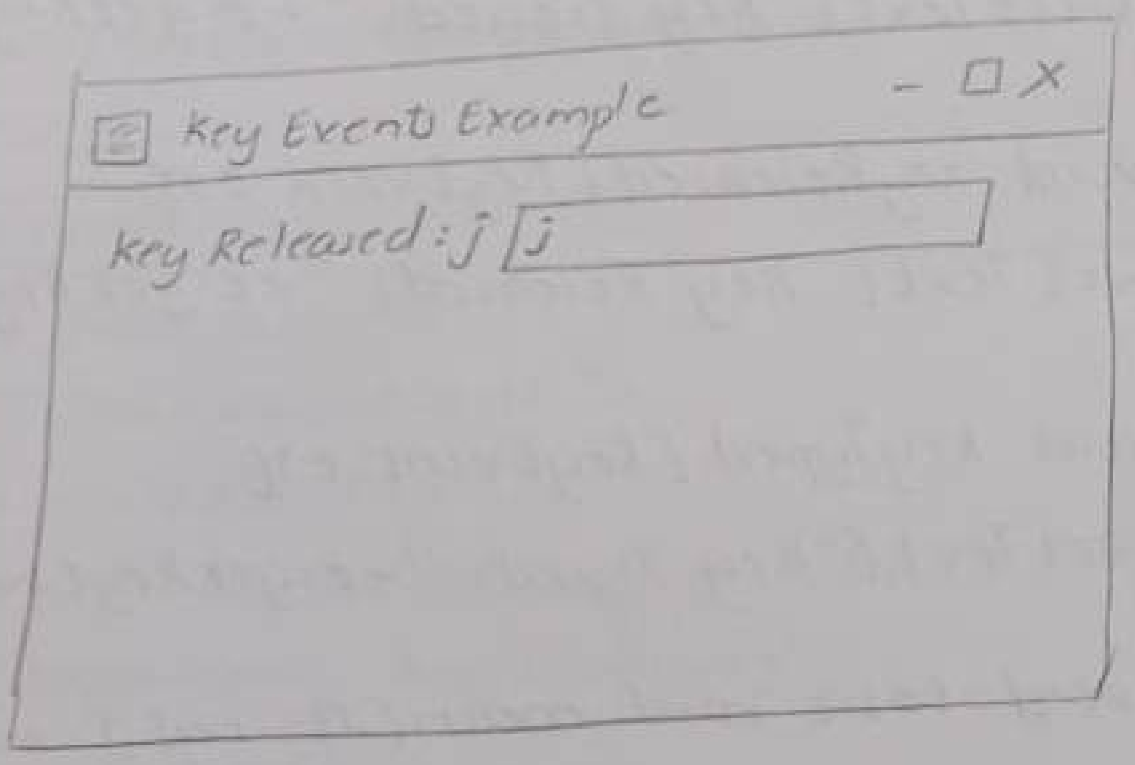
```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class keyEventsExample extends JFrame
        implements KeyListener{
  Jlabel label;
  JTextField textField;
  keyEventsExample()
  {
   setTitle("key Events Example");
  setSize(400,200);
```

```
    setLayout(new FlowLayout());
    setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
    label = new Jlabel("Press any key inside the
                              text field.");
    textField=new JTextField(20);
    textField.addKeyListener(this);
    add(label);
    add(textField);
    setVisible(true);
  }
  public void keyPressed(keyEvent e){
     label.setText("Key Pressed: "+e.getKeyChar());
  }
  public void keyReleased(KeyEvent e){
     label.setText("Key Released: "+e.getKeyChar());
  }
  public void keyTyped(KeyEvent e){
     label.setText("Key Typed: "+e.getkeyChar());
  }
  public ~~void~~ static void main(String[] args)
   {
     new KeyEventsExample();
   }
}
```

OUTPUT



Press any key inside the text field.



Key Released : j j

VIVA QUESTIONS

1. What is the purpose of adapter classes?

Ans. Adapter classes in java provide default(empty) implementations of listener interfaces so subclasses can override only the methods they need.

2. What is the purpose of mouselistener?

Ans. To handle mouse events like clicking, pressing, releasing, entering & exiting.

3. What is the purpose of mousemotionlistener?

Ans. To handle mouse movement events such as dragging and moving the mouse within a component.

4. What is the purpose of keylistener?

Ans. To handle keyboard events such as when a key is pressed, released or typed.

5. What is the purpose of showStatus() method and repaint() method?

Ans. showStatus() method in Java Applets used to display a short message in the status bar of the browser or applet viewer.

repaint() method is used to request the component to be redrawn, which internally calls the update() and paint() methods to refresh the display.

```
Step-6:  Calculate root2= -b – "D/2a
Step-7:  Print value of root1 and root2 these are Real and different
Step-8:  GOTO END
Step-9:  If (D==0) then continue else GOTO Step 13
Step-10: Calculate root1 = root2 = -b/2*a
Step-11: Print value of root1 and root2 the roots are real and equal.
Step-12: GOTO END
Step-13: Calculate RealPart = -b/2a
Step-14: Calculate imaginaryPart = "(-D)/2a
Step-15: Print value of root1= RealPart + i imaginaryPart
Step-16: Print value of root2= RealPart – i imaginaryPart the roots are complex and different
Step-17: END
```

SOURCE CODE

```
import java.util.Sconner;
class Roots {
Public Static void main(string[ ]args) {
int a,b,c;
double x,y;
Scanner sc = new Sconner(system.in)
System.out.print.ln ("enter a,b,c values:");
a= sc.nextInt();
b= sc.nextInt();
c =sc.nextInt();
double f= (b*b)-4*a*c;
System.out.println ("f value = "+f);
```

```
 if (f<0){
System.out.println("no of real roots");
}
else {
 double l=math.sqrt(f);)
 x = ((-b-l)/(2*a));
x = ((-b+l /(2*a));
System.out.println("Roots of given equation:"
                    +x+" \t"+y);
}
}
}
```

OUTPUT

```
Enter a,b,c volues:
 -2
  9
  8
f volue :195.0
roots of equotion: 2.0  2.0
```

```
import.java.util.sconner;
class prime {
Public static void main(string[] args) {
Sconner sc = new sconner(system.in);
System.out.println("enter n value");
int n= sc.next Int():
for (int i=2; i<=n; i++){
int p=0;
for (int j=2; j<=i/2; j++){
 if (i%j==0){
 p=1;
 break;
  }
 }
if (p==0){
 System.out.println(i);
}
}
}
}
```

OUTPUT

```
Enter n value
25
2
3
5
7
11
13
17
19
23
```

SOURCE CODE

```
 import java.util.Scanner;
class chkpolindrome {
Public static void main(string [ ] args){
 String rev=" ";
Scanner sc=new Scanner (system.in)
System.out.println ("enter a string");
String str =sc.nextLine();
 int len= str.length();
 for (int i=len-1 ;i>=0 ;i--)
 rev= rev+str.charAt (i);
if (str.equals(rev))
System.out.println(str+"is a polindrome");
else
system.out.println("not a polindrome");
}
}
```

OUTPUT

```
 enter a string
 modam
madam is a palindrome
 enter a string
 box
  not a palindrome
```

Step-20: for k: 0 to n-1 step 1

Step-21: print A[k]

Step-22: END

SOURCE CODE

```
import java.util.Scanner;
class Sortstrings {
Public Static void main(String [ ] args{
 String temp;
 Scanner sc=new Scanner(System.in);
 System.out.println("enter the value of n:");
 int n= sc.nextInt();
String names[ ]=new String[n];
System.out.println("enter names:");
for(int i=0;i<n;i++){
System.out.println("enter name ["+(i+1)+"]:");
names[i]=sc.nextLine();
 }
for(int i=0;i<n;i++){
for(int j=1;j<n;j++){
if (names[j-1].CompareTo(names[j])>0){
 temp=names[j-1];
names[j-1]=names[j];
names[j]=temp;
```

```
  }
 }
}
System.outprintln("\n sorted names are in
 Ascending order:");
for (int i=0 ; i<n ;i++)
{
system.Out.println(name[i]);
}
}
}
```

OUTPUT

```
Enter the value of n:5
enter names : enter name[1]:1
enter name[2]:4
enter name [3]:7
enter name[4]:8
enter name[5]:2
Sorted names are in Ascending order:
1
2
4
7
8
```

```
class Method@overload
{
 void calvolue()
 {
   int x=20;
   x = x*x;
System.out.println("sqrt of x is :"+x);
}
void Colvalue (int Y)
{
 Y = Y*Y*Y;
System.out.println("cube of Y is:"+Y);
}
void colvalue (int m,int n)
{
  int t=m*n;
system.out.println("product of m& n is:"+z);
}
}
Class MoL
{
 Public static void main (string args[ ])
{
Method overload m=new Methodoverload();
M.calvolue();
M.calvalue(10,20);
M.colvalue(10);
 }
 }
```

OUTPUT

```
Sqrt of x is 400
Product of m & n is 200
cube of y is 1000
```

```
class human{
Public void eat(){
System.out.println("human is eating");
}
}
class boy extends human{
Public void eat(){
System.out.println("boy is eating");
}
Public static void main(string[] args){
boy obj=new boy();
obj.eat();
}
}
```

```
interface Printable.
   {
   void Print();
   }
interface Showable{
   void show();
   }
   Class multiple-inheritance implements
            Printable, showable {
  Public void Print()
  {
  System.out.println("Hello");
  }
  Public void show(){
  System.out.println("welcome");
  }
 Public static void main(String args[ ])
 {
 multiple-inheritance obj = new
          multiple-inheritance ();
obj.print();
obj.show();
}
}
```

OUTPUT

```
Hello
Welcome
```

Step-4: Set the class path to send the class file of A.java source file into classes folder of c: drive.

Step-5: STOP

To Compile

```
E:\SOURCES> JAVAC -D C:\CLASSES SIMPLE.JAVA
```

To Run:

To run this program from e:\source directory, you need to set classpath of the directory where the class file resides.

```
E:\SOURCES> SET CLASSPATH=C:\CLASSES;.; E:\SOURCES> JAVA MYPACK.SIMPLE
```

SOURCE CODE

message.java

```
Package . mypack1;
Public class Message{
Public void show(){
System.out.println("user defined Package");
}}
```

Packagedemo.java

```
import. mypack1.message;
Public class Packagedemo{
Public static void main(String[] args){
Message m= new Message();
M.show();
}}
```

OUTPUT

User defined package.

(B) AIM

To write a Java program to create an abstract class

THEORY

We can require that some methods be overridden by sub classes by specifying the abstract type modifier. These methods are sometimes referred to as sub classer responsibility as they have no implementation specified in the super class. Thus a sub class must override them. Any class that contains one or more abstract methods must also be declared abstract. Such types of classes are known as abstract classes. Abstract classes can contain both abstract and non-abstract methods.

ALGORITHM

Step-1: START

Step-2: Create an abstract class named Shape.

Step-3: Declare integer variables for height, width and radius in the abstract class Shape.

Step-4: Declare abstract method printArea() in the abstract class Shape.

Step-5: Create sub classes Rectangle, Triangle and Circle that extends Shape.

Step-6: Implement the printArea() method in all three classes.

Step-7: Invoke the methods in the main class by the respective objects.

Step-8: END

SOURCE CODE

```
abstract class Shape
{ int a=2;
  int b=4;
  abstract void PrintArea();
}
 class Rectangle extends shape
{
 void PrintArea()
```

```
{ int r=a*b;
  System.Out.println("Area for rectangle="+r);
 }
}
Class Triangle extends shape
{
 void PrintArea()
 {int t: (a*b)/2 ;
System.Out.println("Area for triangle="+t);
}
}
class Circle extends Shape
{ void PrintArea()
{int area= (int)(3.14*a*a);
System.Out.println("area for circle with radius"
     +a "is = "+area);
}
}
Public class AbstractDemo {
Public static void main(Strings argsC])  {
 Rectangle r= new Rectangle();
 Triangle t= new Triangle( );
```

OUTPUT

```
Circle c=new circle();
shape S;
S= r
 S. printArea();
 S=t
 S. print Area( );
 S=c
 S.print Area();
 }
 }
```

O/p:

```
  Area for rectangle =8
  Area for triangle =4
  Area for circle with radius 2 is = 12.
```

Step-8: Write catch block to handle **ArrayIndexOutOfBoundsException**

Step-9: Write catch block to handle **NumberFormatException**

Step-10: Write Finally block

Step-11: Print message 'Exception Handling completed..'

Step-12: STOP

SOURCE CODE

```
Public class MultipleCatchBlock {
Public static void main(string args[]){
try{
int a[ ]=new int[5];
  a[5] =30/0;
}
catch( Arithmetic_Exception e)
{
System.Out.println("Arithmetic Exception occurs");
}
Catch (ArrayIndexOutofBounds Exception e)
{
System.out.println("Array Index out of Bound
                  Exception occurs");
}
Catch(Exception e)
{
System.out.println("Parent Exception occurs");
}
System.Out.println("reset of the Code");
}
}
```

OUTPUT

```
Arithmetic Exception occurs
reset of the code
     int a[] = new int[5]
      if a[9]=3;
O/P: Index Out of Bound Exception occurs.
```

(B) AIM

To write a java pro

THEORY

Creating our own E
exceptions are use
exception, we can
of the powerful n
can be maintaine

ALGORITHM

Step-1: START
Step-2: Create
Step-3: Define
Step-4: Call p
Step-5: Close
Step-6: Crea
Step-7: Defi
Step-8: Vali
Step-9: else
Step-10: D
Step-11: c
a
Step-12:
Step-13:

SOURC

```
Class InvalidAgeException extends Exception{
    InvalidAgeException (String s){
      Super(s);
      }
  }
  class TestCustomException1{
     Static void validate(int age)throws
InvalidAgeException{
        if(age<18)
           throw new InvalidAgeException ("not
        else                                    Valid");
           System.out.println("Welcome to vote");
    }
    Public Static Void main(String args[ ]){
       try{
          validate(13);
          }catch(Exception m) {
System.out.println("Exception occured:"+m);}
          System.out.println("Custom exception
                Demo Completed...");
```

OUTPUT

```
Exception occured:InvalidAgeException: not valid
Custom exception Demo Completed...

If age>18;
    21
=> Welcome to vote
Custom exception Demo Completed...
```