

INDEX

S.No.	Date	Name of the Experiment	Page No.	Marks Awarded	Remarks
1.	25-11-24	function point value.	1-6		
2.	28-11-24	constructive. Cost model (CoCoMo)	7-9.		
3.	2-12-24	Cyclomatic Complexity	10-13		
4.	5-12-24	Software maturity Index (SMI)	14-17		
5	9-12-24	D.I. Digital Differential Analyser	18-22.		
6.	12-12-24	Bresenham's line drawing Algorithm.	23-28		
7	16-12-24	midpoint Circle	29-33.		
8	19-12-24	midpoint Ellipse.	34-39.		
9.	23-12-24	2-Dimensional Basic. Transformations on	40-46		

Date :	GATE Educational Institutions	Page No : 1
	1. Function Point Value	Practical No :

Program Statement:

Design and develop a java program to calculate function point value.

Program - Description:

* Function point is an element of software development which helps to approximate the cost of development early in the process. It may measure functionality from user's point of view.

* Scale varies from 0 to 5 according to character of complexity adjustment factor.

$$F = 14 * \text{Scale}$$

* Calculate unadjusted complexity adjustment factor

$$CAF = 0.65 * (0.01 * F)$$

* Calculate Function point

$$FP = UFP * CAF$$

Algorithm:-

Step 1: Start the program

Step 2: Create class called FP value.

Date :	GATE Educational Institutions	Page No : 2
		Practical No :

Step 3 : Declare and initialize necessary variable and arrays within the FP value class.

Step 4 : Create a constructor method within FP-class to initialize the variables.

Step 5 : Create an input method within the FP-class to accept user input for the counts and average complexity adjustment.

Step 6 : Create a processor method within the FP value class to calculate the function point value.

Step 7 : Create FPVsiree class, invoke input(), process and output(), this is the main method.

Output :

Enter number of external inputs:

2

Enter number of external outputs:

3

Enter number of external inquiries:

1

Enter number of internal logical files:

4

Enter number of external interfaces:

Date :

GATE Educational Institutions

Page No. :
3

Practical No. :

Function Point Value:

Source code:

```
import java.io.*;
import java.lang.*;
class FPValue
{
    int count[] = new int[5];
    int avg[] = new int[5];
    int ct[] = new int[5];
    int i;
    int cttotal;
    int fi;
    float fp;
    BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
    FPValue()
    {
        System.out.println("\n\t Function Point(FP) Value : ");
        System.out.println("\n\t-----");
        cttotal=0;
        fi=0;
        fp=0;
    }
    void input() throws IOException
    {
        for(i=1;i<=5;i++)
        {
            if(i==1)
            {
                System.out.println("\nEnter numbr of external inputs : ");
                count[i] = Integer.parseInt(br.readLine());
            }
            else if(i==2)
```


Date :

GATE Educational Institutions

Page No. : 5

Practical No. :

```
        ct[j] = count[j] * avg[j];

        cttotal = cttotal + ct[j];

    }

    fp = cttotal * (0.65 + (0.01 * fi));

}

void output() throws IOException

{

    System.out.println("\n\n Count total is : " + cttotal);

    System.out.println("\n Sum of fi is : " + fi);

    System.out.println("\n\n\t FP = Count total * (0.65 + (0.01 * sum of fi)) ");

    System.out.println("\n Functional Point (FP) Value is : "+fp);

}

}

class FPVsiree

{

    public static void main(String args[]) throws IOException

    {

        FPValue fp;

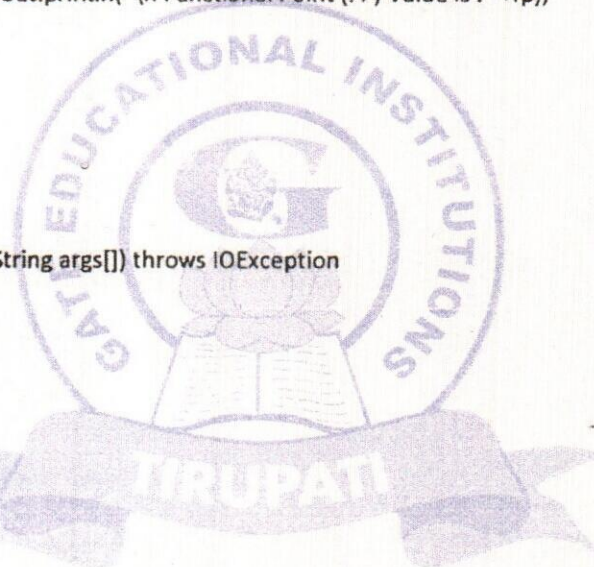
        fp.input();

        fp.process();

        fp.output();

    }

}
```



Date :

GATE Educational Institutions

Page No. : 4

Practical No. :

```
        System.out.println("\nEnter numbr of external outputs : ");
        count[i] = Integer.parseInt(br.readLine());
    }
    else if(i==3)
    {
        System.out.println("\nEnter numbr of external inquiries : ");
        count[i] = Integer.parseInt(br.readLine());
    }
    else if(i==4)
    {
        System.out.println("\nEnter numbr of internal logical files : ");
        count[i] = Integer.parseInt(br.readLine());
    }
    else if(i==5)
    {
        System.out.println("\nEnter numbr of external interfaces : ");
        count[j] = Integer.parseInt(br.readLine());
    }
}
System.out.println("\n\n Enter average complexity adjustments values : \n");
for(i=1;i<=5;i++)
{
    avg[i] = Integer.parseInt(br.readLine());
}
System.out.println("\n Enter Sum of value adjustments factors(fi) : ");
fi = Integer.parseInt(br.readLine());
}
void process() throws IOException
{
    for(int j=1;j<=5;j++)
    {
```


Date :	GATE Educational Institutions	Page No.: 4
	2. COCOMO	Practical No. :

Problem - Definition :

Design and develop a java program to calculate the estimated effort using cocomo model.

Description :

*The cocomo (constructive cost model) is well known software cost estimation model. It is based on the principle that the effort required to develop a software project is proportional to its size and complexity.

Algorithm :

Step 1 : Start the program

Step 2 : Create a class called cocomo.

Step 3 : declare necessary variables for screens reports, components, complexity weights and productivity factor.

Step 4 : Read the inputs from the user and store into the variables.

Step 5 : Calculate the Estimated Effort by the dividing the top by the productivity factor.

Step 6 : display the EE and NOP values.

Output:

Enter no. of Screens:

10

Enter complexity weights:

4

Enter no. of Reports:

5

Enter complexity weights:

3

Enter no. of components:

3

Enter complexity weight:

5

Enter value of prod:

150

$NOP = 104$

The Estimated effort for simple ATM
using cocomo II model is: 0.6933334.

Conclusion:-

The above is executed Successfully.

Date :	GATE Educational Institutions	Page No. :
COCOMO:		Practical No. :

Source code:

```

import java.io.*;

import java.util.*;
class COCOMO
{
    public static void main(String args[])
    {
        Scanner obj = new Scanner (System.in);
        int s,r,c,cw1,cw2,cw3,prod;
        float EE,NOP;
        System.out.println("Enter No of Screens:");
        s = obj.nextInt();
        System.out.println("Enter Complexity weight:");
        cw1 = obj.nextInt();
        System.out.println("Enter No of Reports:");
        r = obj.nextInt();
        System.out.println("Enter Complexity weight:");
        cw2 = obj.nextInt();
        System.out.println("Enter No of Components:");
        c = obj.nextInt();
        System.out.println("Enter Complexity weight:");
        cw3 = obj.nextInt();
        System.out.println("Enter vlaue of prod:");
        prod = obj.nextInt();
        NOP = (float) ((s*cw1)+(r*cw2)+(c*cw3));
        System.out.println("NOP = " + NOP);
        EE = (float) NOP/prod;
        System.out.println("The Estimated effort for simple ATM using Cocomo II model is : " +EE);
    }
}

```


Date :	GATE Educational Institutions	Page No.: 6
	3. CYCLOMATIC COMPLEXITY	Practical No. :

Problem - Definition :

Design and develop a java program to calculate the cyclomatic complexity of a control graph.

Description :

* Cyclomatic complexity is a way to measure the complexity of software modules. It is based on the control flow graph of the program, which represents, the flow the execution between various statements branches and loops.

$$M = E - N + 2$$

- 'M' represents the cyclomatic complexity
- 'E' represents the number of edges in the control flow graph
- 'N' is the number of nodes in the control flow graph.

Algorithm :

Step 1: Start the program

Step 2: Create a class called cyclomatic.

Date :	GATE Educational Institutions	Page No.: 7
		Practical No. :

Step 3: Declare variables for the number of edges, number of nodes (n), and cyclomatic complexity (v).

Step 4: Read the number of edges (e) from the user.

Step 5: Read the number of nodes (n) from the user.

Step 6: Calculate the cyclomatic complexity (v) using the formula $v = e - n + 2$.

Step 7: Display the calculated cyclomatic complexity (v).

Step 8: End the program.

Output:

-Enter the number of edges:

12

-Enter the number of nodes:

10

The cyclomatic complexity of a graph is: 4

Conclusion:

The above java program executed Successfully.

Date :

GATE Educational Institutions

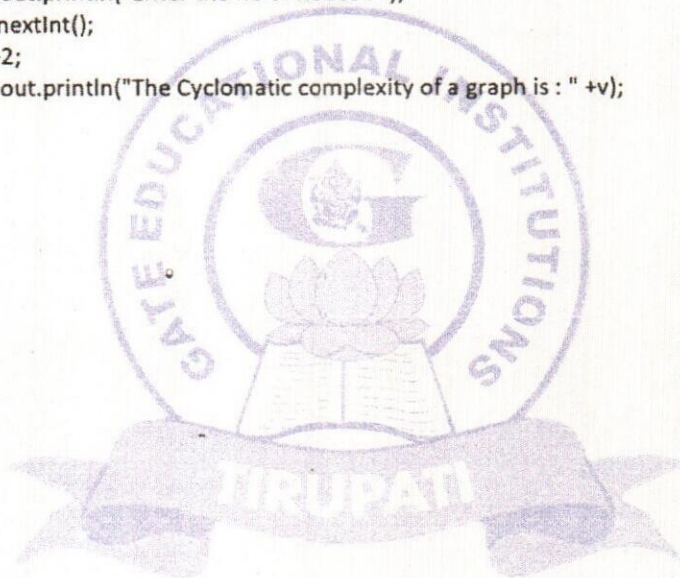
Page No. :
12

Practical No. :

CYCLOMATIC:

Source code:

```
import java.io.*;
import java.util.*;
class Cyclomatic
{
    public static void main(String args[])
    {
        int e,n,v;
        Scanner obj = new Scanner(System.in);
        System.out.println("Enter the no of Edges : ");
        e = obj.nextInt();
        System.out.println("Enter the no of nodes : ");
        n = obj.nextInt();
        v = e-n+2;
        System.out.println("The Cyclomatic complexity of a graph is : " +v);
    }
}
```



Date :	GATE Educational Institutions	Page No.: 8
	H. SMT	Practical No. :

Problem-Defination :-

Design and develop a java program to calculate SMT value.

Description :-

* The Software Maturity Index (SMT) is a metric used to assess the maturity level of a software product or system. It provides a quantitative measure of the software development process and the quality of the resulting software.

* The SMT value indicates the degree to stability, reliability and predictability of the software product

$$SMT = MT - (FA + FC + FD) / MT;$$

SMT - Software Maturity index value

MT - no. of software functions/modules in current.

FC - no. of software functions/modules that contain changes from the previous release.

FD - no. of functions/modules that are deleted from the previous release.

Date :	GATE Educational Institutions	Page No.: 9
		Practical No. :

Algorithm:

Step 1: Start the program

Step 2: Create class called SMI.

Step 3: Declare variables for Mean Time (MT), Failures avoided (FA), Failures corrected (FC), Failures deleted (FD), and Software maturity Index (SMI)

Step 4: Read the value of MT, FA, FC, FD from the user

Step 5: Calculate SMI using the formula.

$$SMI = (MT - (FA + FC + FD)) / MT$$

Step 6: display the calculated SMI.

Step 7: End the program.

Output :-

Enter MT:

100

Enter FA:

20

Enter FC:

10

Enter FD:

5

The resultant SMI is : 0.65

Date :

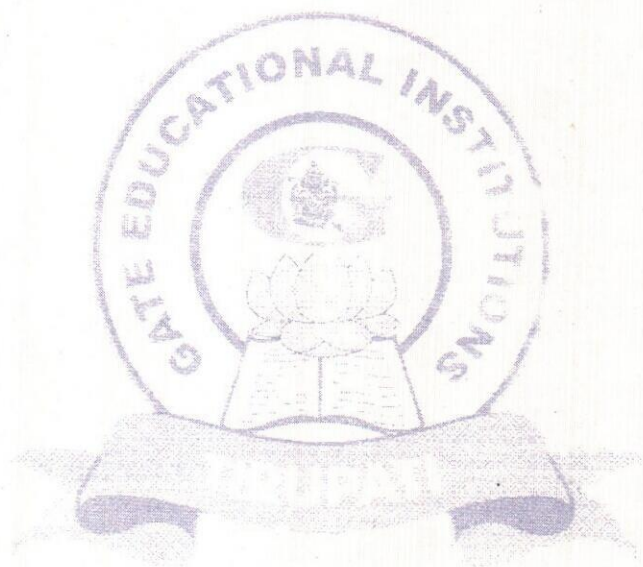
GATE Educational Institutions

Page No.: 10

Practical No. :

Conclusion:-

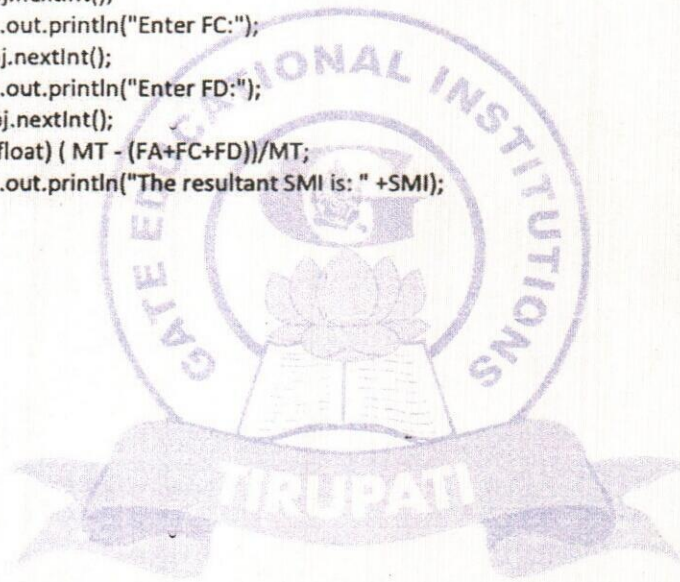
The above java programs executed Successfully



Date :	GATE Educational Institutions	Page No. :
SMI		Practical No. : 16

Source code:

```
import java.io.*;
import java.util.*;
class SMI
{
    public static void main(String args[])
    {
        Scanner obj = new Scanner(System.in);
        int MT,FA,FC,FD;
        float SMI;
        System.out.println("Enter MT:");
        MT = obj.nextInt();
        System.out.println("Enter FA:");
        FA = obj.nextInt();
        System.out.println("Enter FC:");
        FC = obj.nextInt();
        System.out.println("Enter FD:");
        FD = obj.nextInt();
        SMI = (float) ( MT - (FA+FC+FD))/MT;
        System.out.println("The resultant SMI is: " +SMI);
    }
}
```



Date :	GATE Educational Institutions	Page No.: 11
	5. DDA	Practical No. :

Problem - Definition :

Design and develop a java program to draw line by using DDA.

Description :

- * The Digital Differential Analyzer (DDA) is a line drawing algorithm.
- * It is a simple and efficient algorithm that calculates the co-ordinates of pixels along a line segment based on the slope of the line.
- * The DDA algorithm works by incrementally calculating the x and y co-ordinates of pixel on the line.

Algorithm :

- Step 1: Start the program
- Step 2: Create class called 'dda' that extends the 'applet' class and implements the 'ActionListener' interface.
- Step 3: Declaring variable for the starting and ending co-ordinates of the line 'x₁', 'x₂', 'y₁', 'y₂'.

Date :	GATE Educational Institutions	Page No.: 12
		Practical No. :

Step 4: Create tables (L_1, L_2, L_3, L_4) and text field (T_1, T_2, T_3, T_4) and button (b)

Step 5: Implement `init()` method to initialize the applet by adding labels.

Step 6: Implement '`paint()`' to perform the linedrawing

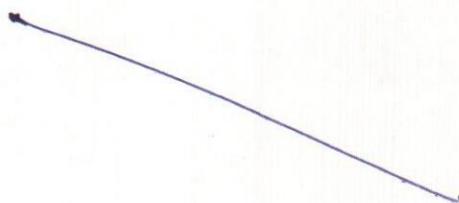
Step 7: `paint()` method can retrieve the co-ordinate values x, y .

Step 8: implement '`ActionPerformed()`' to perform or handle the button clicked event. To calls `Repaint()` method calls `paint()`.

Step 9: End the program.

Output:

$x_1 = 200$ $x_2 = 300$ $y_1 = 100$ $y_2 = 150$ draw



Conclusion:

The above java program executed Successfully

Date :

GATE Educational Institutions

Page No. :

Practical No. : 28

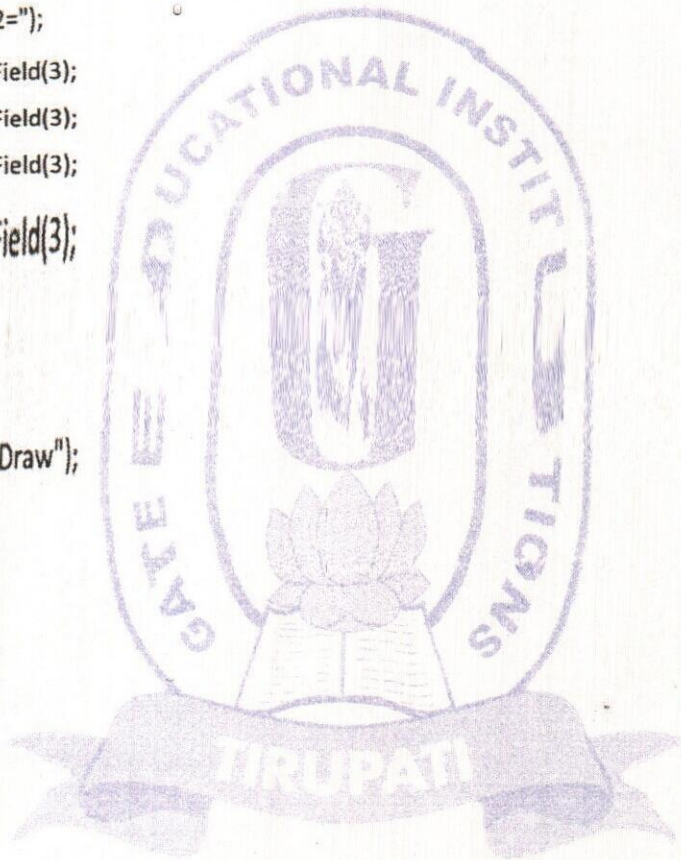
DDA:

Source code:

```
import java.awt.*;
import java.awt.event.*;
import java.applet.*;
public class dda extends Applet implements ActionListener
{
    int x1,x2,y1,y2;

    Label L1=new Label("x1=");
    Label L2=new Label("x2=");
    Label L3=new Label("y1=");
    Label L4=new Label("y2=");
    TextField t1=new TextField(3);
    TextField t2=new TextField(3);
    TextField t3=new TextField(3);
    TextField t4=new TextField(3);

    Button b=new Button("Draw");
    public void init(){
        add(L1);
        add(t1);
        add(L2);
        add(t2);
        add(L3);
        add(t3);
        add(L4);
        add(t4);
        add(b);
        b.addActionListener(this);
    }
    public void paint(Graphics g) {
        float xinc,yinc;
        int step;
        int x1=Integer.parseInt(t1.getText());
        int x2=Integer.parseInt(t2.getText());
        int y1=Integer.parseInt(t3.getText());
        int y2=Integer.parseInt(t4.getText());
        int dx=x2-x1;
```



Date :

GATE Educational Institutions

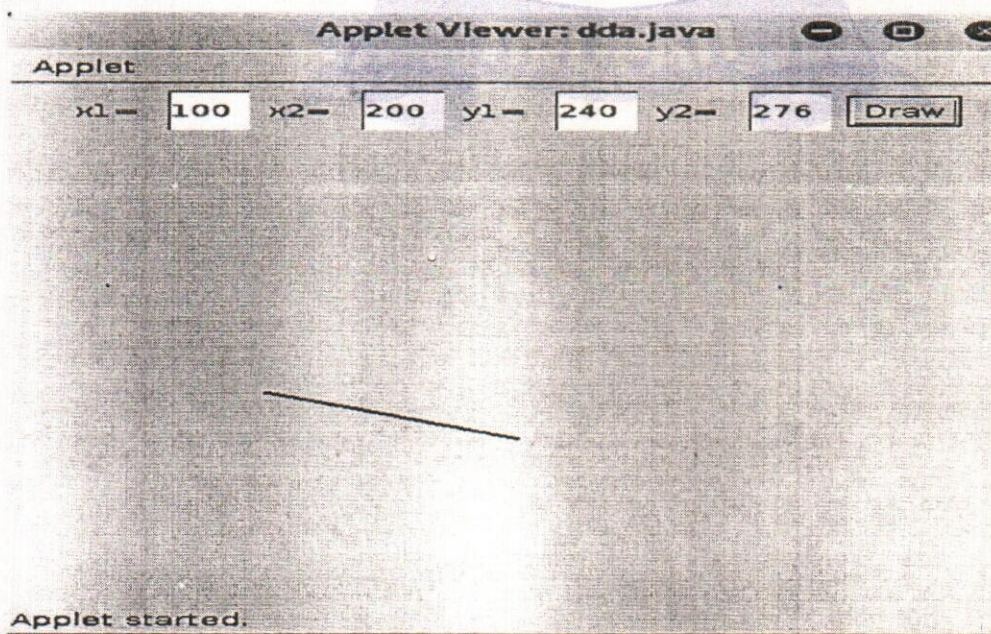
Page No. :

int dy=y2-y1;

Practical No. :

```
if(Math.abs(dx)>Math.abs(dy))
step=Math.abs(dx);
else step=Math.abs(dy);
xinc=(float)dx/step;
yinc=(float)dy/step;
float x=x1;
float y=y1;
g.drawString(".",(int)x,(int)y);
for(int i=1;i<step;i++) {
x=x+xinc;
y=y+yinc;
g.drawString(".",(int)x,(int)y);
}}
public void actionPerformed(ActionEvent a)
{
repaint();
}}
/*<applet code=dda.class width=500 height=500>
</applet>*/
```

Output:



Date :	GATE Educational Institutions	Page No.: 13
	6. Bresanham's Line Drawing Alg	Practical No. :

Problem - Destination :

Design and develop a java program to draw line by using Bresanham's Line Drawing Algorithm.

Description:

- * Bresanham's line Algorithm is an efficient method for rasterizing lines on a grid based display. Such as a computer screen. It determines which pixels to activate to represent a line between two given points.
- * Bresanham's algorithm uses the decision parameters when the increment or decrement the y-coordinate and adjust the error term accordingly. By making efficient choices based on the error term.

Algorithm :

Step-1 : Start the program

Step 2 : Create a class called 'bresan' that extends the 'applet' class and implements the 'ActionListener' interface.

Step 3 : Declare variables L_1, L_2, L_3, L_4 as Label and

text fields T_1, T_2, T_3, T_4 as Text fields and button 'b'.

Step 4: Implement `init()` method to setup the user interface by adding labels, textfield button.

Step 5: Parse the user input co-ordinates from the textfield to integers.

Step 6: Calculate the difference between x and y co-ordinates.

Step 7: Implement Bresenham's line algorithm using a loop to calculate and plot the co-ordinate of each pixel along the line.

Step 8: End the program.

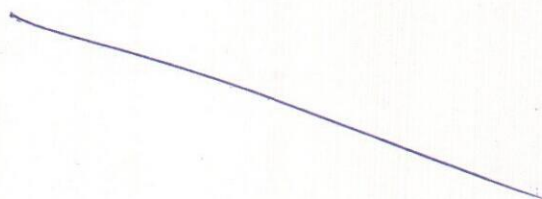
Output:

$x_1 = 20$

$x_2 = 230$

$y_1 = 320$

$y_2 = 140$



Conclusion:

The above java program executed Successfully.

Date :

GATE Educational Institutions

Page No. :

Practical No. : 25

Bresanhams line Drawing Algorithm:

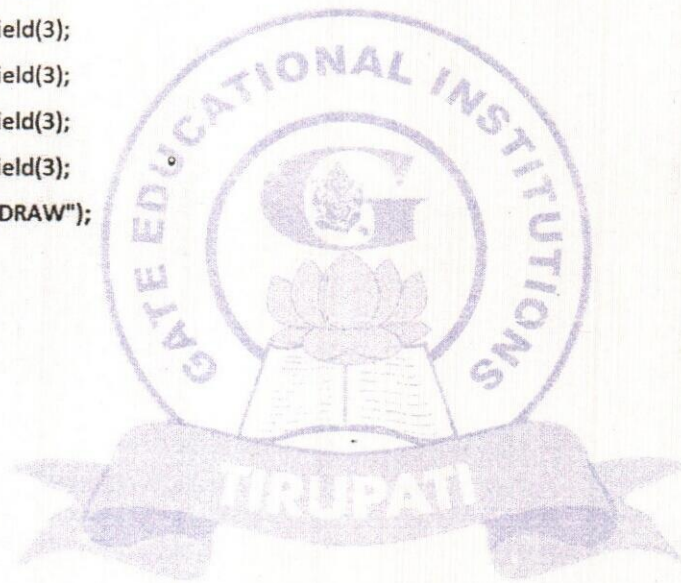
Source Code:

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

public class bresan extends Applet implements ActionListener {

    Label l1=new Label("x1=");
    Label l2=new Label("x2=");
    Label l3=new Label("y1=");
    Label l4=new Label("y2=");
    TextField t1=new TextField(3);
    TextField t2=new TextField(3);
    TextField t3=new TextField(3);
    TextField t4=new TextField(3);
    Button b=new Button("DRAW");
    public void init(){
        add(l1);
        add(t1);
        add(l2);
        add(t2);
        add(l3);
        add(t3);
        add(l4);
        add(t4);
        add(b);
        b.addActionListener(this);
    }

    public void paint(Graphics g) {
        int x1=Integer.parseInt(t1.getText());
        int x2=Integer.parseInt(t2.getText());
        int y1=Integer.parseInt(t3.getText());
        int y2=Integer.parseInt(t4.getText());
        int dx=x2-x1;
        int dy=y2-y1;
        int count=dx;
        int p=2*(dy-dx);
```



Date :	GATE Educational Institutions	Page No. :
int Xend,X,Y;		Practical No. :

```

if(x1>x2) {
X=x2;
Y=y2;
Xend=x1;
}
else {
X=x1;
Y=y1;
Xend=x2;
}
g.drawString("",X,Y);
while(count>0) {
X+=1;
if(p<0) {
p=p+2*dy;
}
else {
Y+=1;
p=p+2*(dy-dx);
}
g.drawString("",X,Y);
count--;
}}
public void actionPerformed(ActionEvent a)
{
repaint();
}}
/*<applet code= bresan.class width=400 height=400>
</applet>*/

```

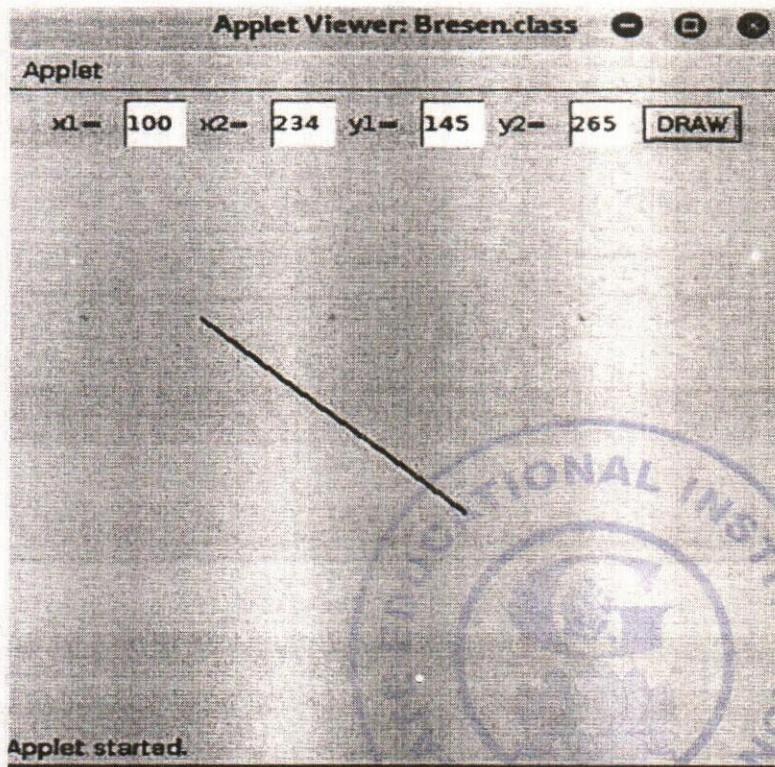

Date :

GATE Educational Institutions

Page No. :

Practical No. :
27

Output:



Date :	GATE Educational Institutions	Page No.: 15
	7. MID POINT CIRCLE	Practical No. :

Problem- Definition:

Design and develop a java program to draw a mid point circle.

Description:

* The mid point circle algorithm is an efficient method for drawing circles on raster display only using integer arithmetic.

* This algorithm works by considering a circles symmetry and utilizes the concept of the midpoint to determining which pixels to plot to approximate circle. It avoids the need for expensive trigonometric calculation floating point, making it computationally efficient.

Algorithm:-

Step 1: Start the program

Step 2: import the required SWT classes and packages.

Step 3: Define a class named "circle" and that extends the applet.

Step 4: Declare the necessary variables and components

Date :	GATE Educational Institutions	Page No.: 16
		Practical No. :

Such as labels, textfields and a button.

Step 5: Override the `paint()` method to perform the circle drawing.

Step 6: Retrieves the values entered in the text fields for the center coordinates (x_c, y_c) and the radius (`rad`).

Step 7: initialize variables `x` and `y` to 0 and `rad` respectively.

Step 8: Repeat the loop until `x` become greater than or equal to `y`.

Step 9: End the program.

Output:-

$x_0 = 100$ $y_0 = 150$ radius = 80 draw

Conclusion:-

The above java program executed Successfully.

Date :

GATE Educational Institutions

Page No. :

Mid-point Circle:

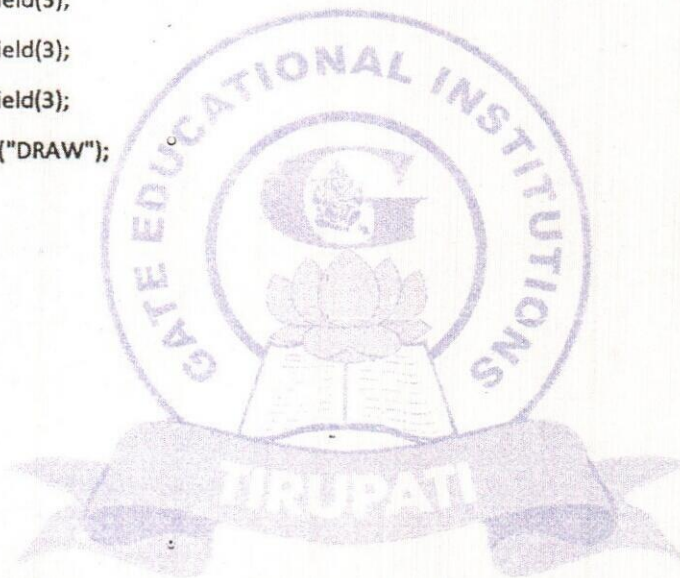
Practical No. :

31

Source Code:

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

public class circle extends Applet implements ActionListener
{
    Label l1=new Label("x0= ");
    Label l2=new Label("x0= ");
    Label l3=new Label("radius= ");
    TextField t1=new TextField(3);
    TextField t2=new TextField(3);
    TextField t3=new TextField(3);
    Button bw=new Button("DRAW");
    public void init() {
        add(l1);
        add(t1);
        add(l2);
        add(t2);
        add(l3);
        add(t3);
        add(bw);
        bw.addActionListener(this);
    }
    public void paint(Graphics g) {
        int xc=Integer.parseInt(t1.getText());
        int yc=Integer.parseInt(t2.getText());
        int rad=Integer.parseInt(t3.getText());
        int x=0;
        int y=rad;
        int p=5/4-rad;
        while(x<y){
            if(p<0){
                x++;
```



Date :

GATE Educational Institutions

Page No. :

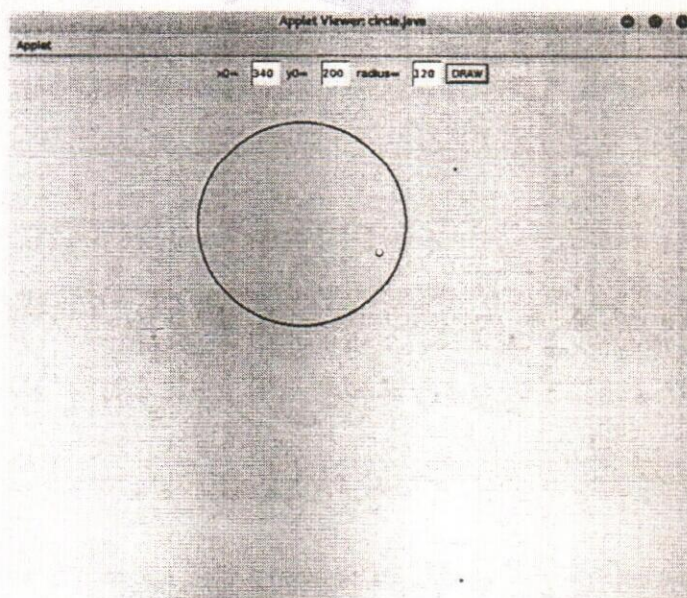
p=p+2*x+2+1;

Practical No. :

32

```
}  
else{  
x++;  
y--;  
p=p+2*x+2+1-2*y+2;  
}  
g.drawString(".",xc+x,yc+y);  
g.drawString(".",xc+y,yc-x);  
g.drawString(".",xc-y,yc+x);  
g.drawString(".",xc-x,yc+y);  
g.drawString(".",xc+x,yc-y);  
g.drawString(".",xc+y,yc+x);  
g.drawString(".",xc-y,yc-x);  
g.drawString(".",xc-x,yc-y);  
}}  
public void actionPerformed(ActionEvent aw){ repaint();  
}}  
/*<applet code=circle.class width=400 height=400>  
</applet>*/-
```

Output:



Date :	GATE Educational Institutions	Page No.: 17
	8. MID POINT ELLIPSE	Practical No. :

Problem- Definition:

Design and develop a java program Mid point Ellipse.

Description:

* Midpoint ellipse algorithm is a method used to draw an ellipse using raster graphics.

* Define a required variables to store the centre co-ordinates (x_c, y_c) of the ellipse, the major-axis length (a) and minor axis length (b) .

* Calculate the square of a and b $a\text{-square} = a * a$ and $b\text{-square} = b * b$.

* Calculate the initial decision parameter

$$P_1 = b\text{-square} - a\text{-square} * b + (0.25 * a\text{-square})$$

$$dx = 2 * b\text{-square} * x$$

$$dy = 2 * a\text{-square} * y$$

Algorithm:

Step 1: Start the program

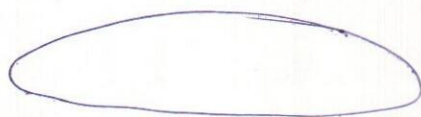
Step 2: Initialize the required variables x, y, x_c, y_c
 xx, yy, P_1 and P_2

Date :	GATE Educational Institutions	Page No.: 18
		Practical No. :

- Step 3 : Create labels and textfields for user input
- Step 4 : Create a button for drawing the ellipse
- Step 5 : Parse the user input for the Semi-major axis (r_x), Semi-minor-axis (r_y) and center co-ordinates (x_c, y_c)
- Step 6 : Set the initial values of x and y to 0 and $2y$.
- Step 7 : Calculate the initial value of P , using the equation $P_1 = (r_y)^2 - rx^2 + rx^2/4$.
- Step 8 : End the program.

Output:

Semi-major-axis - 200 Semi-minor-axis - 300
 x - 130 y - 160 draw



Conclusion:

The above java program executed Successfully.

Date :

GATE Educational Institutions

Page No. :

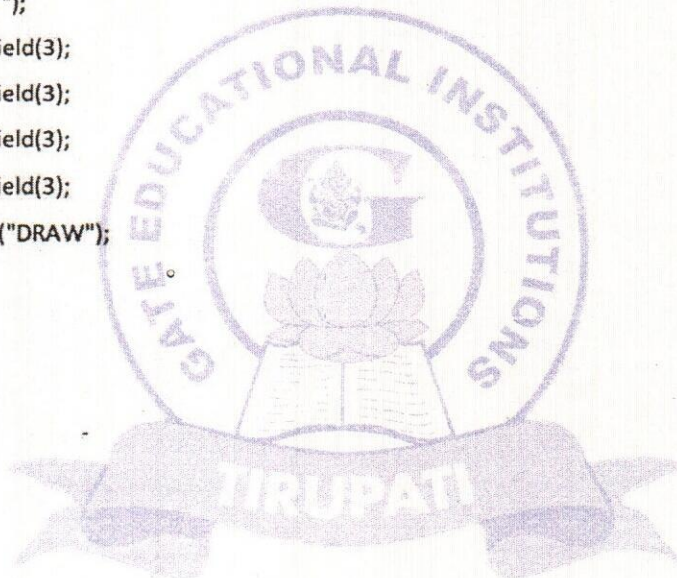
Mid-point Ellipse

Practical No. :
36

Source Code:

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

public class ellipse2 extends Applet implements ActionListener
{
    float x,y,xc,yc,rx,ry,p1,p2;
    Label l1=new Label("semi major axis= ");
    Label l2=new Label("semi minor axis=");
    Label l3=new Label("x=");
    Label l4=new Label("y= ");
    TextField t1=new TextField(3);
    TextField t2=new TextField(3);
    TextField t3=new TextField(3);
    TextField t4=new TextField(3);
    Button bw=new Button("DRAW");
    public void init() {
        add(l1);
        add(t1);
        add(l2);
        add(t2);
        add(l3);
        add(t3);
        add(l4);
        add(t4);
        add(bw);
        bw.addActionListener(this);
    }
    public void paint(Graphics g){
        rx=Integer.parseInt(t1.getText());
        ry=Integer.parseInt(t2.getText());
        xc=Integer.parseInt(t3.getText());
        yc=Integer.parseInt(t4.getText());
        x=0;
        y=ry;
        p1=(ry*ry)-(rx*rx*y)+(rx*rx/4);
```



Date :

GATE Educational Institutions

Page No. :

do{

Practical No. :

37

```

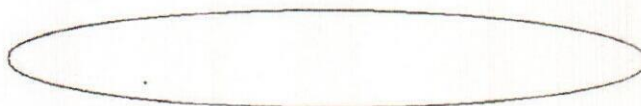
g.drawString(".",Math.round(xc-x),Math.round(yc+y));
g.drawString(".",Math.round(xc-x),Math.round(yc-y));
g.drawString(".",Math.round(xc+x),Math.round(yc+y));
g.drawString(".",Math.round(xc+x),Math.round(yc-y));
x++;
if(p1<0) p1=p1+(2*ry*ry*x)+(ry*ry);
else{
y-=1;
p1=p1+(2*ry*ry*x)-(2*rx*rx*y)+(ry*ry);
}
while((2*ry*ry*x)<(2*rx*rx*y));
p2=(ry*ry*(x+1/2)*(x+1/2))+(rx*rx*(y-1)*(y-1))-(ry*ry*rx*rx);
while(y>0) {
g.drawString(".",Math.round(xc-x),Math.round(yc+y));
g.drawString(".",Math.round(xc-x),Math.round(yc-y));
g.drawString(".",Math.round(xc+x),Math.round(yc+y));
g.drawString(".",Math.round(xc+x),Math.round(yc-y));
y--;
if(p2>0)
p2=p2-(2*rx*rx*y)+(ry*ry);
else{
x+=1;
p2=p2+(2*ry*ry*x)+(2*rx*rx*y)+(ry*ry);
}
public void actionPerformed(ActionEvent aw) { repaint();
}
}
/*<applet code=ellipse2.class width=400 height=400>
</applet>*/

```

Output:

Applet Viewer: ellipse.java
Applet

semi major axis= 320 semi minor axis= 50 x= 334 y= 250 DRAW



Date :	GATE Educational Institutions	Page No.: 19
	9. 2D-Basic Transformation.	Practical No. :

Problem- Destination :-

Design and develop a java program to implement 2D- basic Transformation.

Description :-

- * 2D - transformation refers to the manipulation of objects in a two-dimensional co-ordinate.
- * These transformations are used to modify the position, size, orientation, and shape of 2D objects.
- * These are several types of 2D transformations

1. Translation

2. Scaling

3. Rotation

4. Reflection

5. Shearing

Algorithm:

Step 1 : Start the program

Step 2 : Create a java applet class named "twoD" that extends the applet class and implements

the ActionListener interface.

Step 3: Declare and initialize the necessary variable and components, including, labels, textfield, and buttons.

Step 4: Implement the "init()" method to initialize the applet by adding labels, textfields and buttons to the applet GUI.

Step 5: Implement "paint()" method to handle drawing of the original line.

Step 6: If button b_1 clicked method, check the source of the event button b_2 perform the translation.

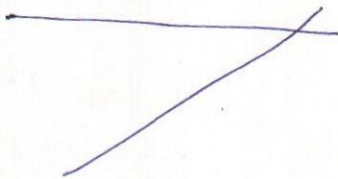
Step 7: End the program.

Output:

$x_1 = 100$ $x_2 = 10$ $y_1 = 20$ $y_2 = 200$ $x_3 = 100$, $y_3 = 200$

Show

Show



Conclusion:

The above java program executed Successfully.

Date :

GATE Educational Institutions

Page No. :

Practical No. :

2D-Basic Transformations:

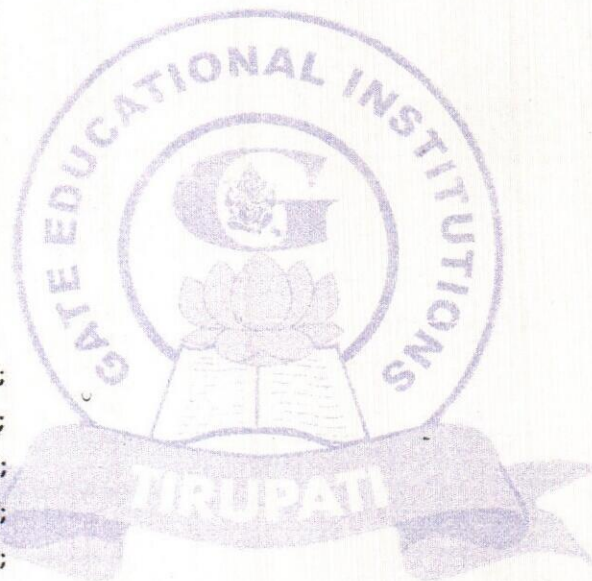
Source Code:

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

public class twoD extends Applet implements ActionListener{
    Label p1=new Label("enter the 2 end points of the line(x1,x2,y1,y2):");
    Label p2= new Label("enter translation factor.[tx,ty]:");
    Label p3= new Label("enter rotation angle and rotation factor[Q,rx,ry]:");
    Label p4=new Label("enter shearing factor[sx,sy]:");
    TextField t1= new TextField(3);
    TextField t2= new TextField(3);
    TextField t3= new TextField(3);
    TextField t4= new TextField(3);
    TextField t5= new TextField(3);
    TextField t6= new TextField(3);
    TextField t7= new TextField(3);
    TextField t8= new TextField(3);
    TextField t9= new TextField(3);
    TextField t10= new TextField(3);
    TextField t11= new TextField(3);
    Button b1=new Button("show");
    Button b2=new Button("show");
    Button b3=new Button("show");
    Button b4=new Button("show");

    int x1,x2,y1,y2,tx,ty,Q,rx,ry,sx,sy,nx1,nx2,ny1,ny2,t=0;

    public void init() {
        add(p1);
        add(t1);
        add(t2);
        add(t3);
        add(t4);
        add(b1);
        add(p2);
        add(t5);
        add(t6);
        add(b2);
```



Date :

GATE Educational Institutions

Page No. : 42

add(p3);

Practical No. :

```
add(t7);
add(t8);
add(t9);
add(b3);
add(p4);
add(t10);
add(t11);
add(b4);
b1.addActionListener(this);
b2.addActionListener(this);
b3.addActionListener(this);
b4.addActionListener(this);
}
public void paint(Graphics g) {
x1=Integer.parseInt(t1.getText());
x2=Integer.parseInt(t2.getText());
y1=Integer.parseInt(t3.getText());
y2=Integer.parseInt(t4.getText());
g.drawLine(x1,y1,x2,y2);
if(t==1)
g.drawLine(nx1,ny1,nx2,ny2);
}
public void actionPerformed(ActionEvent a) {
if(a.getSource()==b1) {
showStatus("original");
repaint();
}
if(a.getSource()==b2) {
tx=Integer.parseInt(t5.getText());
ty=Integer.parseInt(t6.getText());
showStatus("aftertranslation");
t=1;
nx1=x1+tx;
ny1=y1+ty;
nx2=x2+tx;
ny2=y2+ty;
repaint();
}
```


Date :

GATE Educational InstitutionsPage No. :
43

Practical No. :

```

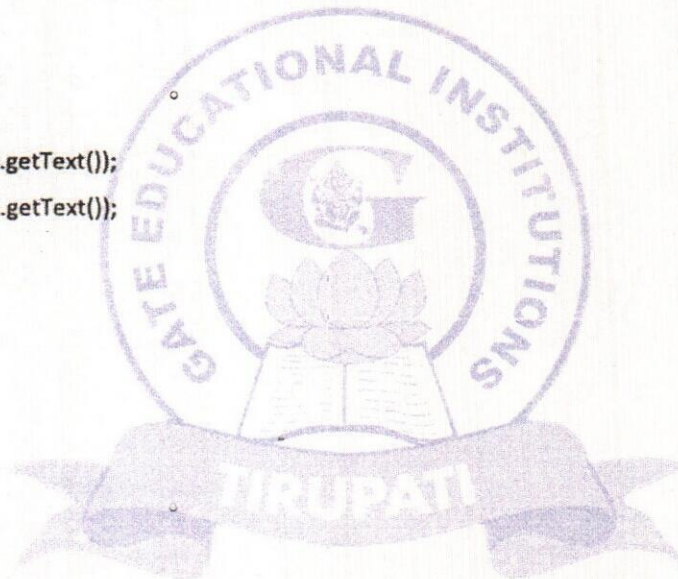
}

if(a.getSource()==b3) {
    Q=Integer.parseInt(t7.getText());
    rx=Integer.parseInt(t8.getText());
    ry=Integer.parseInt(t9.getText());
    showStatus("after rotation");
    t=1;
    double d=Math.toRadians(Q);
    nx1=rx+(int)((x1-rx)*Math.cos(d))-(int)((y1-ry)*Math.sin(d));
    nx2=rx+(int)((x2-rx)*Math.cos(d))-(int)((y2-ry)*Math.sin(d));
    ny1=ry+(int)((x1-rx)*Math.sin(d))-(int)((y1-ry)*Math.cos(d));
    ny2=ry+(int)((x1-rx)*Math.sin(d))-(int)((y1-ry)*Math.cos(d));
    repaint();
}

if(a.getSource()==b4) {
    sx=Integer.parseInt(t10.getText());
    sy=Integer.parseInt(t11.getText());
    showStatus("scaling");
    t=1;
    nx1=x1*sx;
    nx2=x2*sx;
    ny1=y1*sy;
    ny2=y2*sy;
    repaint();
}}

/*<applet code=twoD.class width=400 height=400>
</applet>*/

```



Date :

GATE Educational Institutions

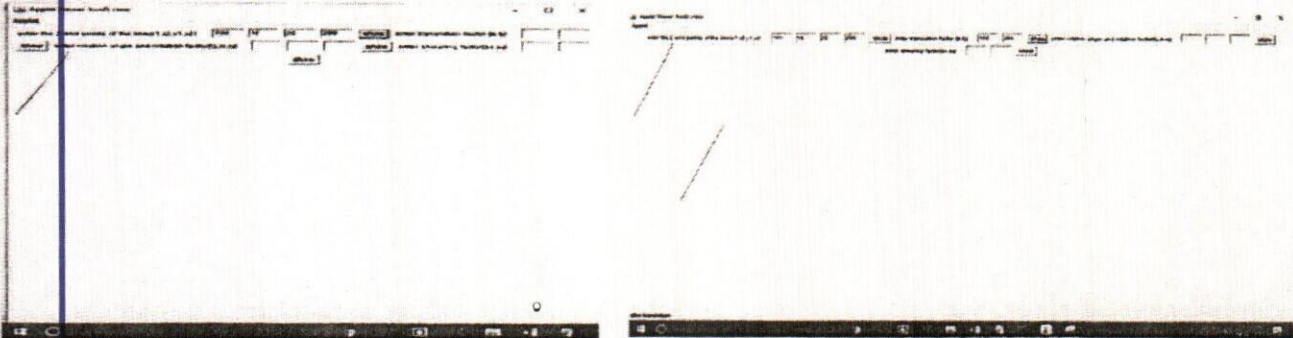
Page No. :

Practical No. :

Output:

Initialline

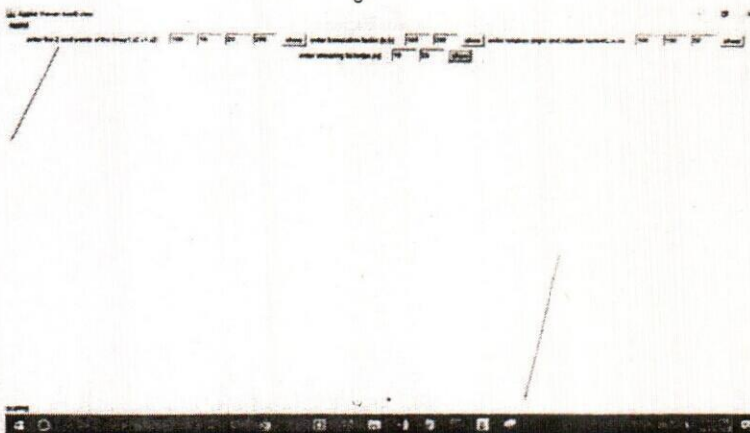
Translation



Scaling:



Shearing:



Date :	GATE Educational Institutions	Page No.: 21
	10. LINE CLIPPING	Practical No. :

Problem-Defination:

Design and develop a java program to implement Line clipping

Description:

- * Line clipping is a technique used to determine which portions of line segment are visible and should be drawn on the screen.
- * It is particularly useful when rendering lines that extends beyond the boundaries of a viewpoint or window.
- * Determine the intersection points of the line segment with the view point boundaries by applying clipping tests.

Algorithm:

Step 1: Start the program

Step 2: Initialize the applet by adding a button and the necessary event listeners.

Step 3: In the 'mousepressed' method, capture the starting coordinates of the line segment.

Step 4: In the 'mouse Released' method, capture the ending co-ordinates of the line segment.

Date :

GATE Educational Institutions

Page No.: 22

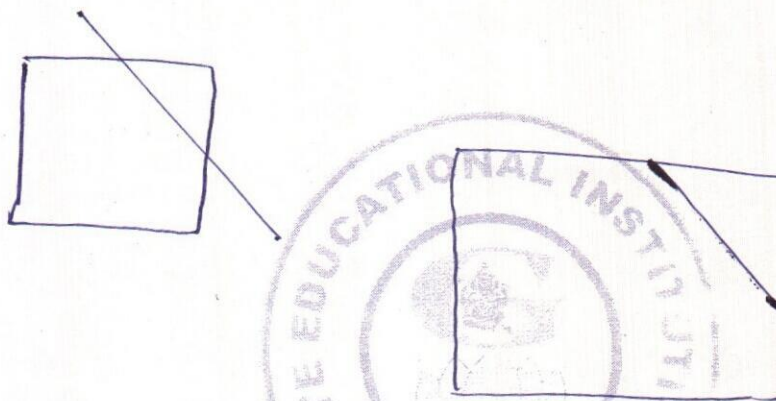
Practical No. :

Step 5: In the paint() method, draw the rectangle region and the original line segment.

Step 6: In the 'Actionperformed' method handle when the button clicked event

Step 7: End the program

Output:



Rectangular region

Conclusion:

The above java program executed Successfully.

Date :	GATE Educational Institutions	Page No. :
		Practical No. : 18

Line Clipping:

Source Code:

```

import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;

public class lineclip extends Applet implements MouseListener,
MouseMotionListener, ActionListener {

    Button b=new Button("clip");

    Graphics key;

    float x,y,xa,ya,xb,yb,dx,dy,m,p,xEnd;

    public void init() {
        add(b);
        b.addActionListener(this);
        addMouseListener(this);
        addMouseMotionListener(this);
    }

    public void mouseEntered(MouseEvent me){}
    public void mouseClicked(MouseEvent me){}
    public void mouseExited(MouseEvent me){}
    public void mouseMoved(MouseEvent me){}
    public void mouseDragged(MouseEvent me){}
    public void mousePressed(MouseEvent me) {
        xa=me.getX();
        ya=me.getY();
    }

    public void mouseReleased(MouseEvent me) {
        xb=me.getX();
        yb=me.getY();
        repaint();
    }

    public void paint(Graphics g) {
        g.drawRect(200,200,200,200);
        g.drawLine(Math.round(xa),Math.round(ya),Math.round(xb),Math.round(yb));
    }

    public void actionPerformed(ActionEvent a) {
        key=getGraphics();
        key.setColor(Color.white);

```


Date :

GATE Educational Institutions

Page No. :

49

Practical No. :

```
key.drawLine(Math.round(xa),Math.round(ya),Math.round(xb),Math.round(yb));
```

```
key.setColor(Color.black);
```

```
dx=Math.abs(xa-xb);
```

```
dy=Math.abs(ya-yb);
```

```
if(dx==0)
```

```
    m=2;
```

```
else
```

```
    m=(yb-ya)/(xb-xa);
```

```
    if(m>=0&&m<=1) {
```

```
        p=2*dy-dx;
```

```
        if(xa>xb){
```

```
            x=xb;
```

```
            y=yb;
```

```
            xEnd=xa;
```

```
        }
```

```
    else{
```

```
        x=xa;
```

```
        y=ya;
```

```
        xEnd=xb;
```

```
    }
```

```
    plot_clip(key); //comment
```

```
    while(x<xEnd){
```

```
        x=x+1;
```

```
        if(p<0)
```

```
            p=p+2*dy;
```

```
        else{
```

```
            y=y+1;
```

```
            p=p+2*(dy-dx);
```

```
        }
```

```
        plot_clip(key);
```

```
    }}
```

```
    if(m<0&&m>=-1) {
```

```
        p=2*dy-dx;
```

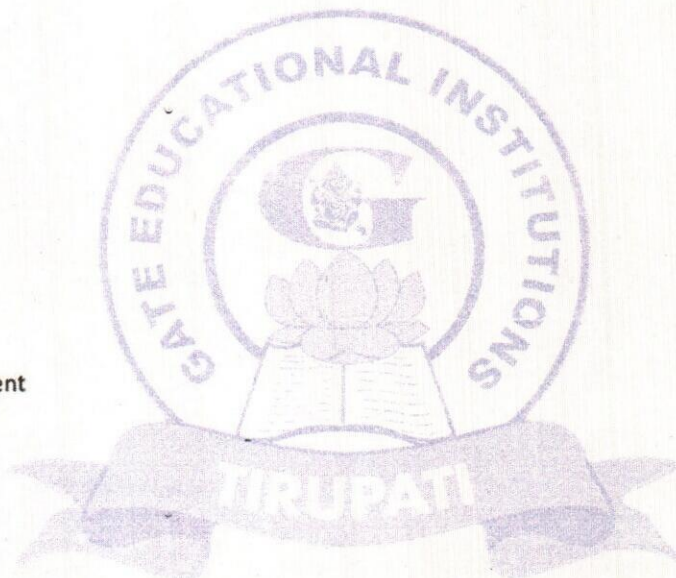
```
        if(xa<xb){
```

```
            x=xb;
```

```
            y=yb;
```

```
            xEnd=xa;
```

```
        }
```

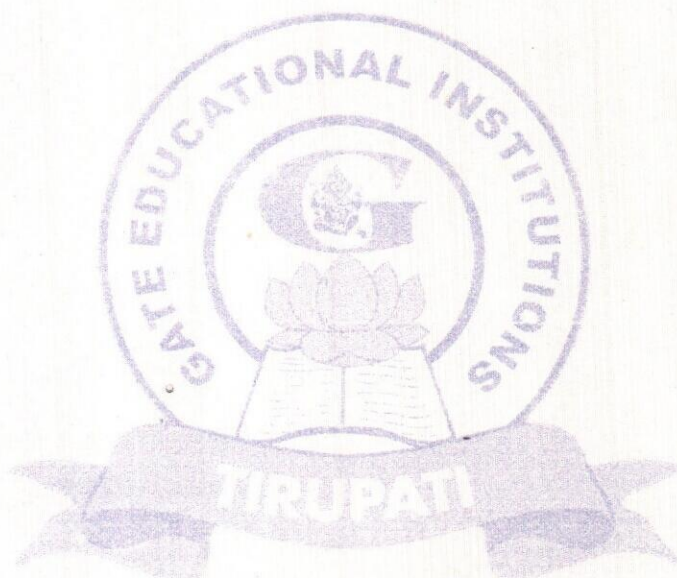


Date :	GATE Educational Institutions	Page No. : 50
else{		Practical No. :

```

x=xa;
y=ya;
xEnd=xb;
}
plot_clip(key);
while(x>xEnd) {
x=x-1;
if(p<0)
p=p+2*dy;
else{
y=y+1;
p=p+2*(dy-dx);
}
plot_clip(key);
}}
if(m<-1) {
p=2*dx-dy;
if(ya<yb) {
x=xb;
y=yb;
xEnd=ya;
}
else{
x=xa;
y=ya;
xEnd=yb;
}
plot_clip(key);
while(y>xEnd){
y=y-1;
if(p<0)
p=p+2*dx;
else{
x=x+1;
p=p+2*(dx-dy);
}
plot_clip(key);

```



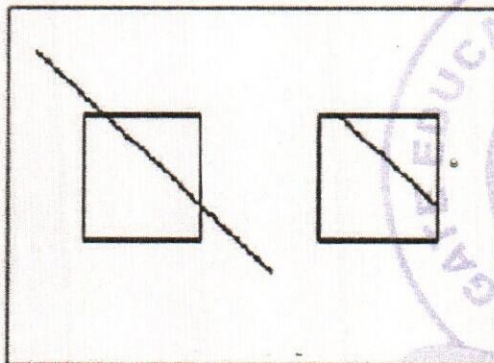
Date :	GATE Educational Institutions	Page No. : 51
}}		Practical No. :

```

public void plot_clip(Graphics key) {
    if(x<200 || x>400 || y<200 || y>400)
        key.setColor(Color.white);
    else
        key.setColor(Color.black);
    key.drawString(".",Math.round(x),Math.round(y));
}
}
/*<applet code=lineclip.class width=600 height=600>
</applet>*/

```

Output:



Date :	GATE Educational Institutions	Page No.: 23
	11. Cohen Sutherland Line Clipping	Practical No. :

Problem - Definition:

Design and develop a java program to implement cohen Sutherland Line clipping algorithm.

Description:

- * The most commonly used the clipping Algorithm.
- * is the cohen Sutherland (Line clipping) Algorithm.
- * Which divided the 2D Space into several regions based on the boundaries of the view point.
- * Each region is represented by a 4-bit code that indicates whether a point is inside or outside of the viewpoint.

Algorithm:

- Step 1: Start the program
- Step 2: Initialise the applet button and registering the necessary event listener.
- Step 3: Implement the mouse event listener methods to handle mouse actions.
- Step 4: In the mouse pressed method, capture the

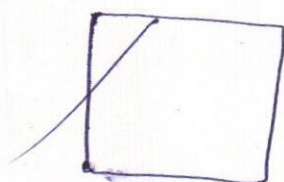
Starting co-ordinates of the line segment.

Step 5: In the paint method, draw the rectangular region and the original line segment.

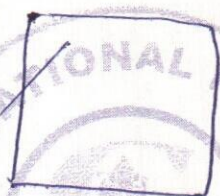
Step 6: Implement the 'plot-clip' method to draw a dot at each visible co-ordinate.

Step 7: End the program.

Output:



- 1.8



Second [1, 0, 0, 0] [0, 0, 0, 0]

Second [1, 0, 0, 0]

Conclusion:

The above java program executed Successfully.

Date :

GATE Educational Institutions

Page No. :

55
Practical No. :

Cohen Sutherland Line clipping Algorithm:

Source Code:

```
import java.util.*;
import java.lang.*;
import java.awt.*;
import java.awt.event.*;
import java.applet.*;

public class CSL extends Applet {
    int Xmax=90,Ymax=80,Xmin=40,Ymin=40;

    public int[] set(int x,int y) {
        int a[]=new int [4];

        if(x<Xmin)
            a[3]=1;
        else
            a[3]=0;

        if(x>Xmax)
            a[2]=1;
        else
            a[2]=0;

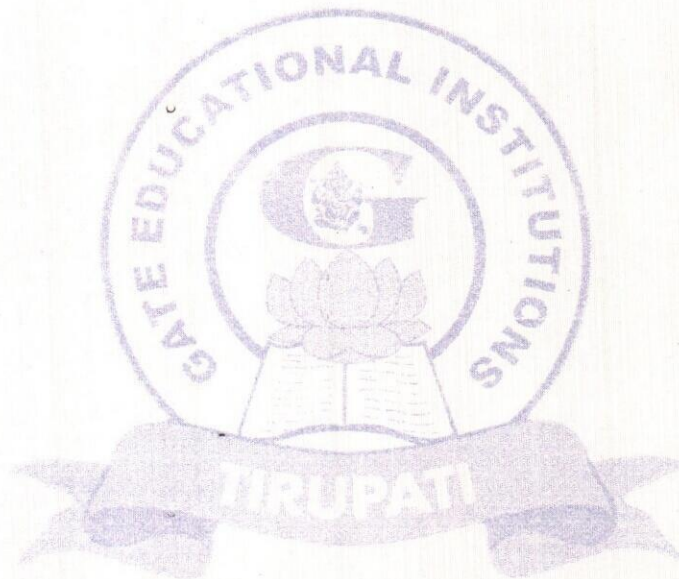
        if(y<Ymin)
            a[0]=1;
        else
            a[0]=0;

        if(y>Ymax)
            a[1]=1;
        else
            a[1]=0;

        return a;
    }

    boolean check(int a[]) {
        for(int i=0;i<a.length;i++) if(a[i]==1)
            return false;
        return true;
    }

    int [] productXY(int i,int X1,int Y1,float m){
        int a[]=new int[2];
        float x=0,y=0;
```



Date :

GATE Educational Institutions

Page No. :

56

Practical No. :

switch(1) {

case 0:

x=Xmin;

y=Y1+m*(x-X1);

break;

case 1:

x=Xmax;

y=Y1+m*(x-X1);

break;

case 2:

y=Ymax;

x=X1+(y-Y1)/m;

break;

case 3:

y=Ymin;

x=X1+(y-Y1)/m;

break;

}

a[0]=(int)x;

a[1]=(int)y;

return a;

}-

boolean doAnd(int a[],int b[]) {

for(int i=0; i<a.length;i++){

int k=a[i]&b[i];

if(k==1)

return false;

}

return true;

}

public void paint(Graphics g) {

g.drawRect(Xmin,Ymin,Xmax-Xmin,Ymax-Ymin);

g.drawRect(Xmin+100,Ymin,Xmax-Xmin,Ymax-Ymin);

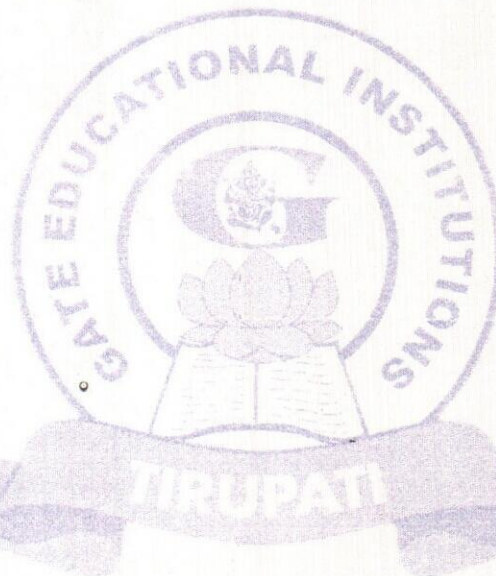
int a[][]=new int[2][4];

int b[][]=new int[2][4];

int c[]=new int [2];

int c1=20;

int X1=45,Y1=45,X2=20,Y2=90;



Date :

GATE Educational Institutions

Page No. :

57

Practical No. :

float m=(Y2-Y1)*1.0f/(X2-X1);

g.drawString(m+" ",100,100);

g.drawLine(X1,Y1,X2,Y2); a[0]=set(X1,Y1);

a[1]=set(X2,Y2);

g.drawString(Arrays.toString(a[0]),300,300);

g.drawString(Arrays.toString(a[1]),400,400);

if(check(a[0])&& check(a[1])){

g.drawLine(X1,Y1,X2,Y2);

}

else{

if(doAnd(a[0],b[0])){

for(int i=a[0].length-1;i>=0;i--){

if(a[0][i]==1){

c=productXY(a[0].length-1-i,X1,Y1,m);

b[0]=set(c[0],c[1]);

g.drawString("first"+Arrays.toString(b[0]),200,300+c1);

if(check(b[0])){

X1=c[0];

Y1=c[1];

break;

}

c1+=20;

}}

for(int i=a[0].length-1;i>=0;i--){

if(a[1][i]==1){

c=productXY(a[0].length-1-i,X1,Y1,m);

b[1]=set(c[0],c[1]);

g.drawString("second"+Arrays.toString(b[1]),200,300+c1);

if(check(b[1])){

X2=c[0];

Y2=c[1];

break;

}

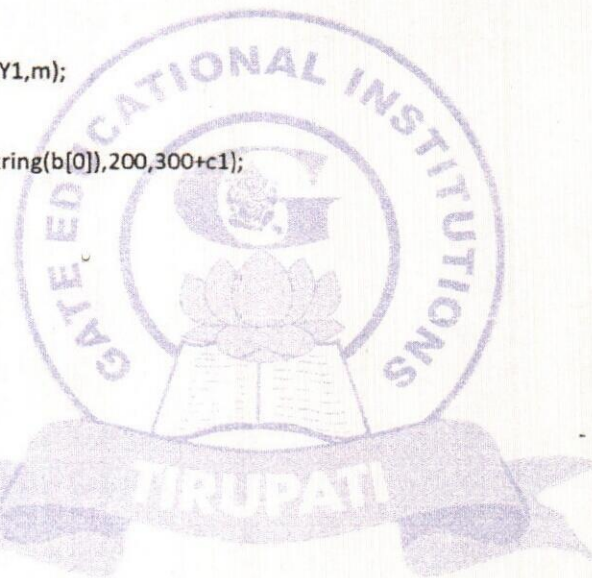
c1+=20;

}}

g.drawLine(X1+100,Y1,X2+100,Y2);

}}}

/*<applet code=CSL.class width=400 height=400>



Date :	GATE Educational Institutions	Page No. :
</applet>*/		58 Practical No. :

Output:



Date :	GATE Educational Institutions	Page No.: 25
	12. clock	Practical No. :

Problem-Defination:

Design and develop a java program to draw a clock.

Algorithm:

Step 1: Start the program

Step 2: Create a new java class that extends the 'Applet' class.

Step 3: Import the necessary Libraries for graphics & time handling such as java.util.*, java.awt.*

Step 4: Override the init() method of applet class this method is called when the applet is initialized.

Step 5: Override the paint method of Applet class called whenever the applet needs to be redrawn.

Step 6: Draw a clock face using drawoval(), draw the hour hand, minute drawline().

Step 7: End the program.

Date :

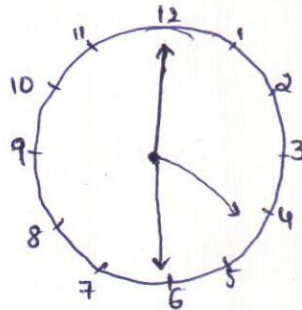
GATE Educational Institutions

Page No.: 26

Practical No. :

Output:

hour minute second



Conclusion:

The above java program executed Successfully.



Date :	GATE Educational Institutions	Page No. :
Clock:		Practical No. : 61

Source Code :

```
import java.awt.*;
import java.applet.Applet;
import java.awt.event.*;
import java.awt.image.*;
import java.lang.*;

public class clock extends Applet implements ActionListener
{
    int hour;
    int min;
    int second;
    float h,m,s;
    Label label1=new Label("hour ");
    Label label2=new Label("minute ");
    Label label3=new Label("second");
    TextField text1=new TextField(3);
    TextField text2=new TextField(3);
    TextField text3=new TextField(3);
    Button button=new Button("start");

    public void init() {
        add(label1);
        add(text1);
        add(label2);
        add(text2);
        add(label3);
        add(text3);
        add(button);
        button.addActionListener(this);

        second=-90;
        min=-90;
        hour=-90;
    }

    public void paint(Graphics g) {
        g.setColor(Color.black);

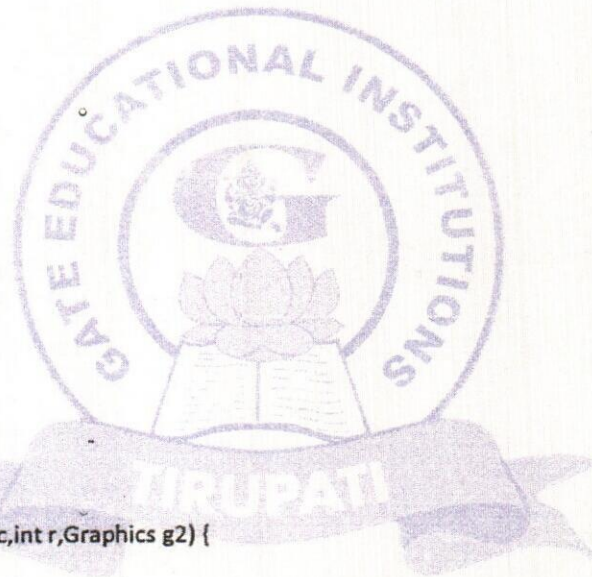
        int radius=91,j=0;
        String num[]={"3","4","5","6","7","8","9","10","11","12","1","2"};
```


Date :	GATE Educational Institutions	Page No. :
	<pre> midcircle(202,147,103,g); for(int i=0;i<360;i+=30) g.drawString(num[j++],(int)(202+radius*Math.cos(Math.PI*i/180)),(int)(147+radius*Math.sin(Math.PI*i/180))); g.setColor(Color.red); ddaline(202,147,(int)(202+(radius-20)*Math.cos(Math.PI*(int)hour/180)),(int)(147+(radius-20)*Math.sin(Math.PI*(int)hour/180)),g); g.setColor(Color.blue); ddaline(202,147,(int)(202+(radius-10)*Math.cos(Math.PI*min/180)),(int)(147+(radius-10)*Math.sin(Math.PI*min/180)),g); g.setColor(Color.green); ddaline(202,147,(int)(202+radius*Math.cos(Math.PI*second/180)),(int)(147+radius*Math.sin(Math.PI*second/180)),g); g.setColor(Color.red); second++; if(second>270) { second=-90; min+=6; hour+=(float)1/2; if(hour>270) hour=-90; if(min>270) min=-90; } for(double r=0;r<100000000;r++); for(double r=0;r<100000000;r++); for(double r=0;r<100000000;r++); for(double r=0;r<100000000;r++); for(double r=0;r<20000000;r++); for(double r=0;r<20000000;r++); for(double r=0;r<20000000;r++); for(double r=0;r<20000000;r++); for(double r=0;r<10000;r++); for(double r=0;r<10000;r++); for(double r=0;r<10000;r++); for(double r=0;r<10000;r++); repaint(); } public void actionPerformed(ActionEvent ae) { h=Integer.parseInt(text1.getText()); </pre>	Practical No. :

Date :

GATE Educational InstitutionsPage No. 3

Practical No. :

`m=Integer.parseInt(text2.getText());``s=Integer.parseInt(text3.getText());``min=(int)(6*m-90);``hour=(int)((30*((m/60)+h))-90);``second=(int)(6*s-90);``if(ae.getSource()==button) {``repaint();``}}``public void ddaline(int x1,int y1,int x2,int y2,Graphics g1) {``int dx=x2-x1,dy=y2-y1,step;``float xinc,yinc,x,y;``if(Math.abs(dx)>Math.abs(dy))``step=Math.abs(dx);``else step=Math.abs(dy);``xinc=(float)dx/step;``yinc=(float)dy/step;``x=x1;y=y1;``g1.drawString(".",(int)x,(int)y);``for(int i=1;i<step;i++) {``x=x+xinc;``y=y+yinc;``g1.drawString(".",(int)x,(int)y);``}}``public void midcircle(int xc,int yc,int r,Graphics g2) {``int x2=0;``int y2=r;``int p=5/4-r;``while(x2<y2){``if(p<0){``x2++;``p=p+2*x2+2+1;``}``else {``x2++; y2--;``p=p+2*x2+2+1-2*y2+2;``}``g2.drawString(".",xc+x2,yc+y2);``g2.drawString(".",xc+y2,yc+x2);`

Date :

GATE Educational Institutions

Page No. : 64

Practical No. :

```
g2.drawString(".",xc-y2,yc+x2);
```

```
g2.drawString(".",xc-x2,yc+y2);
```

```
g2.drawString(".",xc+x2,yc-y2);
```

```
g2.drawString(".",xc+y2,yc-x2);
```

```
g2.drawString(".",xc-y2,yc-x2);
```

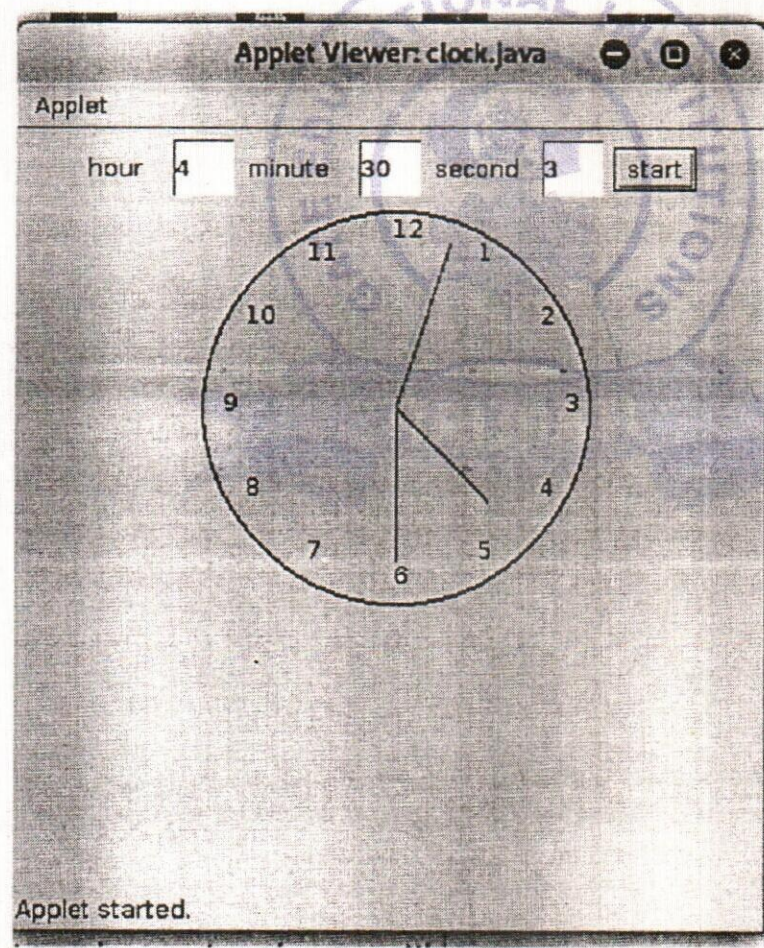
```
g2.drawString(".",xc-x2,yc-y2);
```

```
}}}
```

```
/*<applet code=clock.class width=400 height=400>
```

```
</applet>*/
```

Output:



Date :	GATE Educational Institutions	Page No.: 27
	13. Wheel Rotation	Practical No. :

Problem- Definition :

Design and develop a java program to draw wheel rotation.

Algorithm:

Step 1: Start the program

Step 2: Create class named wheelrot extends applet class.

Step 3: Declare instance variables to store the wheel image, rotation, rotation angle, and other necessary parameters.

Step 4: Override the init() method of the 'Applet' class. This method called when the applet is initialised.

Step 5: Override the paint() method of the Applet class. This method is called whenever the applet needs to be redrawn.

Step 6: Override the mouse dragged() method mouse motion listener interface.

Step 7: call the repaint() method to the trigger to redraw the applet.

Step 8: End the program.

Date :

GATE Educational Institutions

Page No. :

Wheel Rotation:

Practical No. :

68

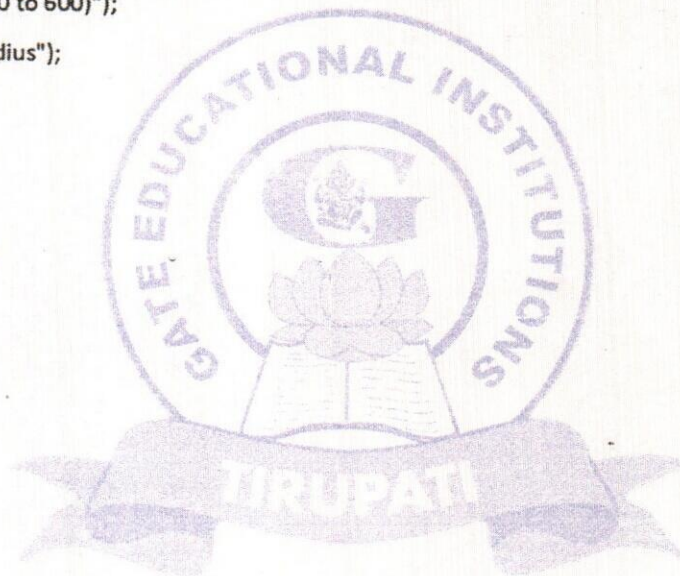
Source Code:

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

public class wheelrot extends Applet implements ActionListener {
    TextField t1,t2,t3;
    Button b;

    public void init() {
        Label l1=new Label("xc(0 to 1000)");
        Label l2=new Label("yc(0 to 600)");
        Label l3=new Label("radius");
        t1=new TextField(4);
        t2=new TextField(4);
        t3=new TextField(4);
        b=new Button("start");
        add(l1);
        add(t1);
        add(l2);
        add(t2);
        add(l3);
        add(t3);
        add(b);
        b.addActionListener(this);
    }

    public void paint(Graphics g) {
        Graphics g1=g;
        int xc,yc,i,r,k;
        double angle,pi,d;
        pi=3.14;
        angle=pi/180;
        xc=Integer.parseInt(t1.getText());
        yc=Integer.parseInt(t2.getText());
        r=Integer.parseInt(t3.getText());
```



Date :	GATE Educational Institutions	Page No. :
		Practical No. : <u>8</u>


```

while(xc!=0){
    for(double j=0;
    j<100000;j+=0.05);
    g.setColor(Color.black);
    g.fillRect(1,1,1360,1000);
    g.setColor(Color.green);
    drwLine(1,yc+r,1360,yc+r,g1);
    drwOval(xc,yc,r,g1);
    drwLine((int)(xc+r*Math.sin(angle)),
    (int)(yc-r*Math.cos(angle)),
    (int)(xc-r*Math.sin(angle)),
    (int)(yc+r*Math.cos(angle)),g1);
    drwLine((int)(xc-r*Math.cos(angle)),
    (int)(yc-r*Math.sin(angle)),
    (int)(xc+r*Math.cos(angle)),
    (int)(yc+r*Math.sin(angle)),g1);
    drwLine((int)(xc+r*Math.sin(angle+3.14/4)),
    (int)(yc-r*Math.cos(angle+3.14/4)),
    (int)(xc+r*Math.sin(angle+3.14/4)),
    (int)(yc+r*Math.cos(angle+3.14/4)),g1);
    drwLine((int)(xc-r*Math.cos(angle-3.14/4)),
    (int)(yc-r*Math.sin(angle-3.14/4)),
    (int)(xc+r*Math.cos(angle-3.14/4)),
    (int)(yc+r*Math.sin(angle-3.14/4)),g1);
    xc+=10*pi*r/180;
    angle=angle+10*pi/180;
    if(xc>(1000-r))
    xc=20;
    }}
    public void actionPerformed(ActionEvent aa) {
    repaint();
    }
    public void drwLine(int c,int d,int e,int f, Graphics g2) {
    int x1=c;

```


Date :

GATE Educational Institutions

Page No. :

int y1=d;

Practical No. : 70

int x2=e;

int y2=f;

int step;

float xinc,yinc,x,y;

int dy=y2-y1;

int dx=x2-x1;

if(dx>dy)

step=dx;

else

step=dy;

xinc=(float)dx/step;

yinc=(float)dy/step;

x=x1;

y=y1;

g2.drawString(".",(int)x,(int)y);

for(int k=1;k<=step;k++) {

x=x+xinc;

y=y+yinc;

g2.drawString(".",(int)x,(int)y);

}}

public void drwOval(int a,int b,int r,Graphics g1){

int xc=a;

int yc=b;

int radius=r;

int x=0;

int y=radius;

float p=5/4-r;

do {

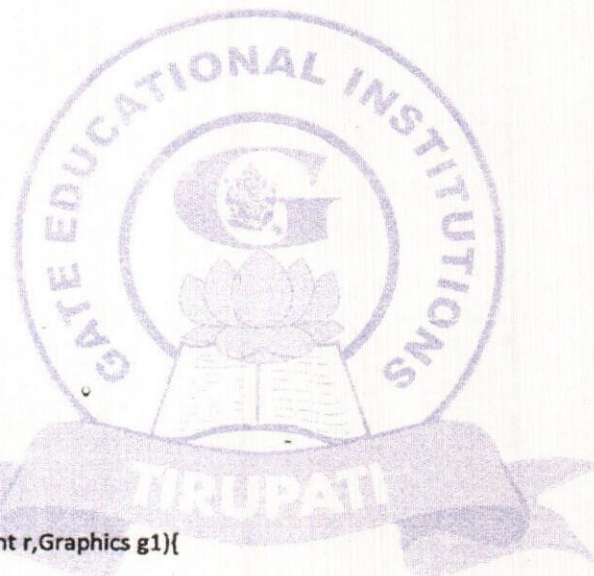
g1.drawString(".",Math.round(xc+x),Math.round(yc-y));

g1.drawString(".",Math.round(xc+y),Math.round(yc-x));

g1.drawString(".",Math.round(xc+y),Math.round(yc+x));

g1.drawString(".",Math.round(xc+x),Math.round(yc+y));

g1.drawString(".",Math.round(xc-y),Math.round(yc+x));



Date :

GATE Educational Institutions

Page No. :

Practical No. :

```
g1.drawString(".",Math.round(xc-y),Math.round(yc-x));
```

```
g1.drawString(".",Math.round(xc-x),Math.round(yc-y));
```

```
if(p<0) {
```

```
x=x+1;
```

```
p=p+(2*x)+3;
```

```
}
```

```
else {
```

```
x=x+1;
```

```
y=y-1;
```

```
p=p-2*y+2*x+5;
```

```
}}
```

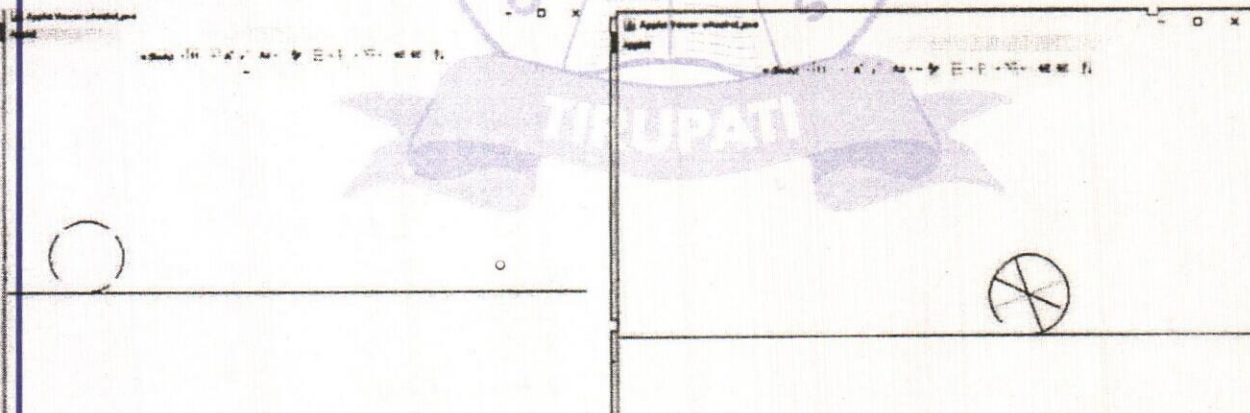
```
while(x<=y);
```

```
}}
```

```
/*<applet code=wheelrot.class width=600 height=400>
```

```
</applet>*/
```

Output:



Date :	GATE Educational Institutions	Page No.: 28
	14. Bezeir Curve	Practical No. :

Problem - Defination:

Design and Develop a java program to implement Bezier curve

Description:

- * Bezier curve is a type of curve commonly used for in computer graphics and geometric modeling.
- * It is set of control points that influence its shape.
- * The curve starts at the first control point and ends at the last control point.
- * While the intermediate curve points determine the curve path.
- * To draw bezier curve various algorithms can be used.

Algorithm :-

Step 1 : Start the program

Step 2 : Input the co-ordinates of the three control points $P_0, P_1, P_2, \dots$

Step 3 : Set the parameter t to a value between 0 and 1 (inclusive) to define the position along the curve.

Step 4 : Compute the blending functions

$$B_0(t) = (1-t)^2$$

$$B_1(t) = 2 * t * (1-t)$$

$$B_2(t) = t^2$$

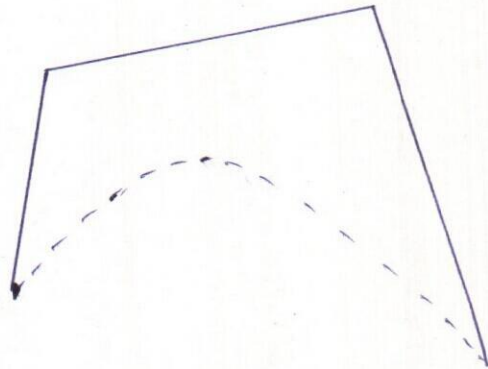
Step 5 : Repeat the Steps 2-4 for different values of t to generate points along the curve

Step 6 : Connect the generated points to form the bezier curve.

Step 7 : End the program.

Date :	GATE Educational Institutions	Page No.: 30
		Practical No. :

Output :-



Conclusion :

The above java program executed Successfully

Ventura
Lochitha

