Date _____

Expt. No. 1 Page No. _____

Write a program to generate a line using Bresenham's line drawing technique. Consider slopes greater than one and slopes less than one. User must be able to draw as many lines and specify inputs through keyboard/mouse.

```
#define BLACK 0
#include <stdio.h>
#include <GL/glut.h>
 int x1, x2;
 int y1, y2;

void draw_pixel (int x, int y, int value)
{
   glBegin (GL_POINTS);
   glVertex2i (x, y);
    glEnd();
}

void bres (int x1, int x2, int y1, int y2)
{
 int dx, dy, i, e;
 int incx, incy, inc1, inc2;
 int x, y;
 dx = x2 - x1;
  dy = y2 - y1;
 if (dx<0)
    dx = -dx;
 if (dy<0)
      dy = -dy;
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
incx = 1;
  if (x2 < x1)
   incx = -1;

incy = 1
   if ( y2 < y1)
    incy = -1;

x = x1;
y = y1;
 if (dx > dy)
  {
   draw_pixel (x, y, BLACK);
    e = 2 * dy - dx;
   inc 1 = 2 * (dy - dx);
   inc 2 = 2 * dy;
  for ( i = 0; i < dx; i++)
     {
        if ( e >= 0)
        {
           y += incy;
           e += inc 1;
         }
       else  e += inc 2;
        x += inc x;
       draw_pixel (x, y, BLACK);
     }
  }
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
else
{
draw_pixel(x, y, BLACK);
e = 2*dx - dy;
inc1 = 2*(dx - dy);
inc2 = 2*dx;
for(i=0; i<dy; i++)
{
  if(e>=0)
  {
    x += incx;
    e += inc1;
  }
  else e += inc2;
  y += incy;

draw_pixel(x, y, BLACK);
}
}
}

void display(){
  glClear(GL_COLOR_BUFFER_BIT);
  bres(x1, y1, x2, y2);
  glFlush();
}
```

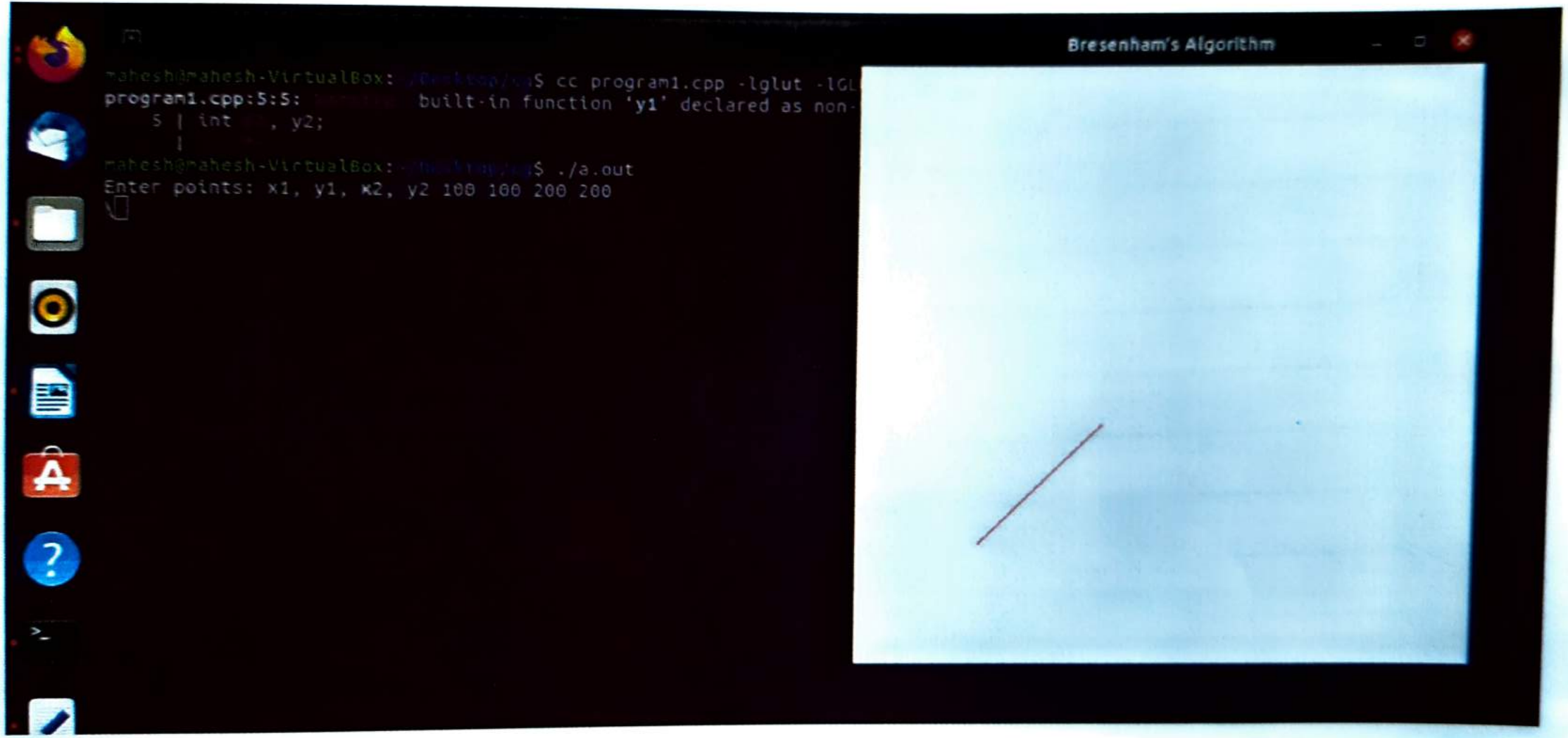
Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
void myinit ()
{
 glClearColor ( 1.0, 1.0, 1.0, 1.0);
 glColor3f ( 1.0, 0.0, 0.0);
 glPointSize (1.0);
 glMatrixMode ( GL-PROTECTION);
 glLoadIdentity ();
  gluOrtho2D ( 0.0, 499.0, 0.0, 499.0);
}

int main ( int argc, char** argv)
{
 printf (" Enter points : x1, y1, x2, y2 ");
 scanf (" %d %d %d %d ", &x1, &y1, &x2, &y2);

 glutInit (&argc, argv);
 glutInitDisplayMode (GLUT_SINGLE | GLUT_RGB);
 glutInitWindowSize (500, 500);
 glutInitWindowPosition (0,0);
 glutCreateWindow (" Bresenham's Algorithm ");
 glutDisplayFunc ( display );
 myinit ();
  glutMainloop ();
}
```

Teacher's Signature _____


mahesh@mahesh-VirtualBox: ~/Desktop/cg$ cc program1.cpp -lglut -lGL
program1.cpp:5:5: built-in function 'y1' declared as non-
5 | int , y2;
mahesh@mahesh-VirtualBox: ~/Desktop/cg$ ./a.out
Enter points: x1, y1, x2, y2 100 100 200 200
Bresenham's Algorithm

Date _____

Expt. No. 2 Page No. _____

Write a program to generate a circle and ellipse using Bresenham's circle drawing and ellipse drawing techniques. Use two windows to draw circle in one window and ellipse in the other window. User can specify inputs through keyboard / mouse.

```
#include <GL/glut.h>
#include <stdio.h>
#include <math.h>
int xc, yc, r;
int rx, ry, xce, yce;

void draw_circle (int xc, int yc, int x, int y)
{
    glBegin (GL_POINTS);
    glVertex2i (xc + x, yc + y);
    glVertex2i (xc - x, yc + y);
    glVertex2i (xc + x, yc - y);
    glVertex2i (xc - x, yc - y);
    glVertex2i (xc + y, yc + x);
    glVertex2i (xc - y, yc + x);
    glVertex2i (xc + y, yc - x);
    glVertex2i (xc - y, yc - x);
    glEnd;
}

void circlebres ()
{
    glClear (GL_COLOR_BUFFER_BIT);
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
int x = 0, y = r;
int d = 3 - 2*r;
while ( x <= y)
  {
     draw_circle (xc, yc, x, y);
     x++;
     if (d<0)
          d= d + 4*x + 6;
     else
      {
          y--;
          d = d + 4*(x-y) + 10;
      }
     draw_circle ( xc, yc, x, y);
  }
  glFlush ();
}
int p1_x, p2_x, p1_y, p2_y;
int point1_done = 0;

void myMouseFuncircle (int button, int state, int x, int y)
{
   if ( button == GLUT_LEFT_BUTTON && state == GLUT_DOWN
     && point1_done == 0)
    {
          p1_x = x - 250;
          p1_y = 250 - y;
          point1_done = 1;
    }
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
else if (button == GLUT_LEFT_BUTTON && state = GLUT_DOWN)
{
    x2 - x = x - 250;
    x2 - y = 250 - y;
      xc = x1 - x;
      yc = x1 - y;
  float exp = x2x x2 - x - x1 - y
float exp = (x2 - x - x1 - x)*(x2 - x - x1 - x)
          + (x2 - y - x1 - y)*(x2 - y - x1 - y);
    r = (int)(sqrt(exp));
  circlebres();
   point1 - done = 0;
  }
}

//ELLIPSE//

void draw_ellipse (int xce, int yce, int x, int y)
{
   glBegin(GL_POINTS);
   glVertex2i(x + xce, y + yce);
    glVertex2i(-x + xce, y + yce);
    glVertex2i(x + xce, -y + yce);
    glVertex2i(-x + xce, -y + yce);
    glEnd();
}
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
void midptellipse()
{
    glClear(GL_COLOR_BUFFER_BIT);
    float dx, dy, d1, d2, x, y;
    x=0;
    y= ry;
    d1 = (ry * ry) - (rx * rx * ry) + (0.25 * rx * rx);
    dx = 2 * ry * ry * x;
    dy = 2 * rx * rx * y;

while ( dx < dy)
{
    draw_ellipse( xce, yce, x, y);
    if (d1 < 0)
      {
          x++;
          dx = dx + (2* ry * ry);
          d1 = d1 + dx + (ry * ry);
      }
    else
      {   x++;
          y--;
          dx = dx + (2* ry * ry);
          dy = dy - (2* rx * rx);
          d1 = d1 + dx - dy + (ry * ry);
      }
}
d2 = ((ry * ry) * ((x+0.5) * (x+0.5))) + ((rx * rx) * ((y-1) * (y-1)))
    - (rx * rx * ry * ry);
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
while (y>=0)
{
  draw_ellipse (xce, yce, x, y);
  if (dz>0)
   {   y--;
       dy = dy - (2*rx*rx);
       dz = dz + (rx*rx) - dy;
   }
  else
   {   y--;
       x++;
       dx = dx + (2*ry*ry);
       dy = dy + (2*rx*rx);
       dz = dz + dx - dy + (rx*rx);
   }
  }
  glFlush();
}
int p1e_x, p2e_x, p1e_y, p2e_y, p3e_x, p3e_y;
int point1e_done = 0;
void myMouseFunc(int button, int state, int x, int y)
{
  if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN
      && point1e_done == 0)
  {  p1e_x = x - 250;
     p1e_y = 250 - y;
      xce = p1e_x;
      yce = p1e_y;
      point1e_done = 1;
  }
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
else if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN
        && point1e_done == 1)
{
    p2e_x = x - 250;
    p2e_y = 250 - y;
float exp = (p2e_x - p1e_x) * (p2e_x - p1e_x)
            + (p2e_y - p1e_y) * (p2e_y - p1e_y);
    rx = (int)(sqrt(exp));
    point1e_done = 2;
}
else if (button == GLUT_LEFT_DOWN && BUTTON &&
        state == GLUT_DOWN && point1e_done == 2)
{
    p3e_x = x - 250;
    p3e_y = 250 - y;
float exp = (p3e_x - p1e_x) * (p3e_x - p1e_x)
            + (p3e_y - p1e_y) * (p3e_y - p1e_y);
    ry = (int)(sqrt(exp));
    midptellipse();
    point1e_done = 0;
}
}
void myDrawing(){}
void myDrawingc(){}
void minit(){
        glClearColor(1,1,1,1);
        glColor3f(1.0, 0.0, 0.0);
        glPointSize(3.0);
        gluOrtho2D(-250, 250, -250, 250);
}
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

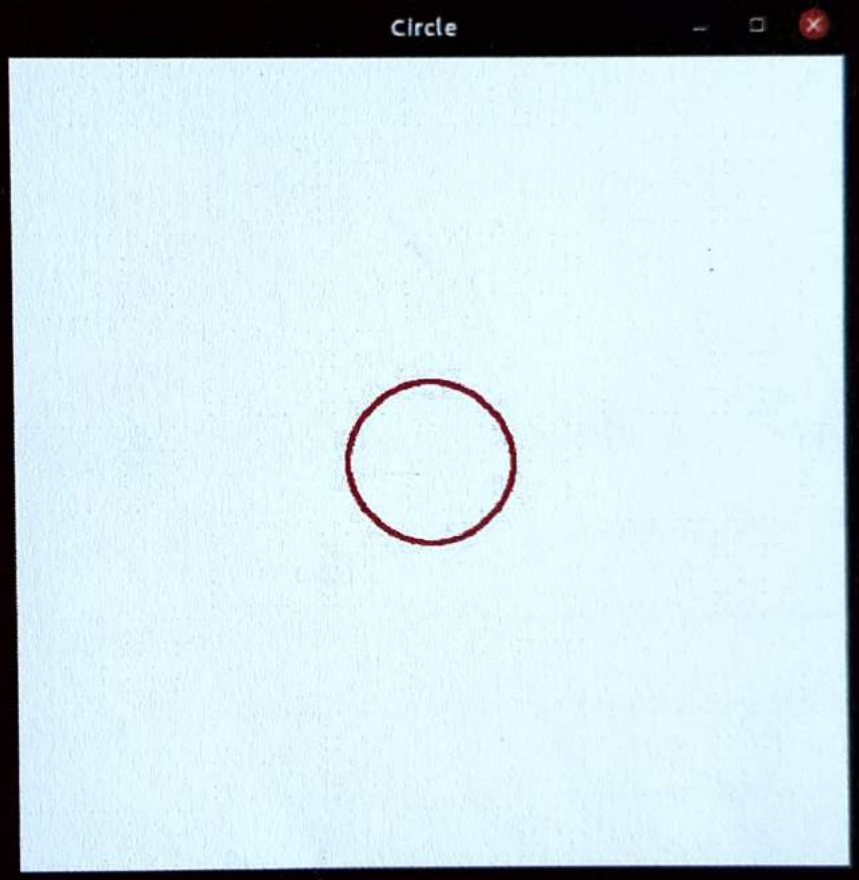
```
int main ( int argc, char * argv [ ]) {

  glut Init ( & argc, argv);
  glut Init Display Mode ( GLUT_SINGLE | GLUT_RGB);
  glut Init Window Size ( 500, 500);
  glut Init Window Position (0, 0);
  printf (" Enter 1 to draw circle , 2 to draw ellipse \n");
  int ch;
   scanf (" %. d", & ch);
switch (ch) {
case 1 :
printf (" Enter coordinates of centre of circle and radius \n");
scanf (" %d %. d %. d ", & xc, & yc, & r);
glut Create Window (" Circle ");
 glut Display Func ( circlebres);
  break;


case 2 :
printf (" Enter coordinates of centre of ellipse and major and
        minor radius \n");
scanf (" %. d %. d %. d %. d", & xce, & yce, & rx, & ry);
glut Create Window (" Ellipse ");
 glut Display Func ( midptellipse);
  break;
}
  minit ( );
  glut Main Loop ( );
}
```

Teacher's Signature _____

Activities Unknown Dec 2 12:05

mahesh@mahesh-VirtualBox: ~/Desktop/cg/git

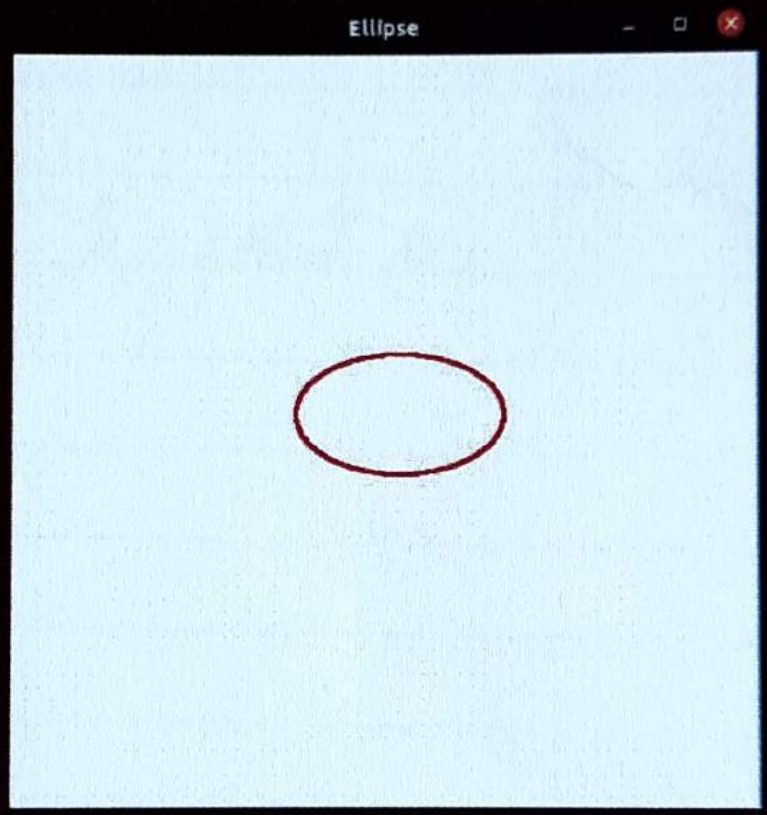
```
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program2.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
Enter 1 to draw circle , 2 to draw ellipse
1
Enter coordinates of centre of circle and radius
0
0
50
```



Activities Unknown Dec 2 12:05

mahesh@mahesh-VirtualBox: ~/Desktop/cg/git

```
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program2.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
Enter 1 to draw circle , 2 to draw ellipse
2
Enter coordinates of centre of ellipse and major and minor radius
10
10
70
40
```



Date _____

Expt. No. 3 Page No. _____

Write a program to recursively subdivide a tetrahedron to form 3D Sierpinski gasket. The number of recursive steps is to be specified at execution time.

```
#include <GL/glut.h>
typedef GLfloat point[3];
point v[] = {{30.0, 50.0, 100.0}, {0.0, 450.0, -150.0},
             {-350.0, -400.0, -150.0}, {350., -400., -150.0}};
int n;

void triangle (point a, point b, point c)
{
 glBegin (GL_TRIANGES);
 glVertex3fv(a);
 glVertex3fv(b);
 glVertex3fv(c);
 glEnd();
}

void divide_triangle (point a, point b, point c)
{
 point v0, v1, v2;
 int j;
 if (m>0)
 { for(j=0; j<3; j++) v0[j]= (a[j]+b[j])/2;
   for(j=0; j<3; j++) v1[j]= (a[j]+c[j])/2;
   for(j=0; j<3; j++) v2[j]= (b[j]+c[j])/2;
 divide_triangle (a, v0, v1, m-1);
 divide_triangle (c, v1, v2, m-1);
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
    divide_triangle (b, v2, v0, m-1);
  }
  else (triangle (a, b, c));
}

void tetra (int m)
{
  glColor3f (1.0, 0.0, 0.0);
  divide_triangle (v[0], v[1], v[2], m);
  glColor3f (0.0, 1.0, 0.0);
  divide_triangle (v[3], v[2], v[1], m);
  glColor3f (0.0, 0.0, 1.0);
  divide_triangle (v[0], v[3], v[1], m);
  glColor3f (0.0, 0.0, 0.0);
  divide_triangle (v[0], v[2], v[3], m);
}

void display()
{
  glClearColor (1.0, 1.0, 1.0, 1.0)
  glClear (GL_COLOR_BUFFER_BIT);
  tetra (n);
  glFlush ();
}
void myinit() {
      glMatrixMode (GL_PROJECTION);
      glLoadIdentity ();
      glOrtho (-499.0, 499.0, -499.0, -499.0, 499.0);
      glMatrixMode (GL_MODELVIEW);
}
```

Teacher's Signature _____

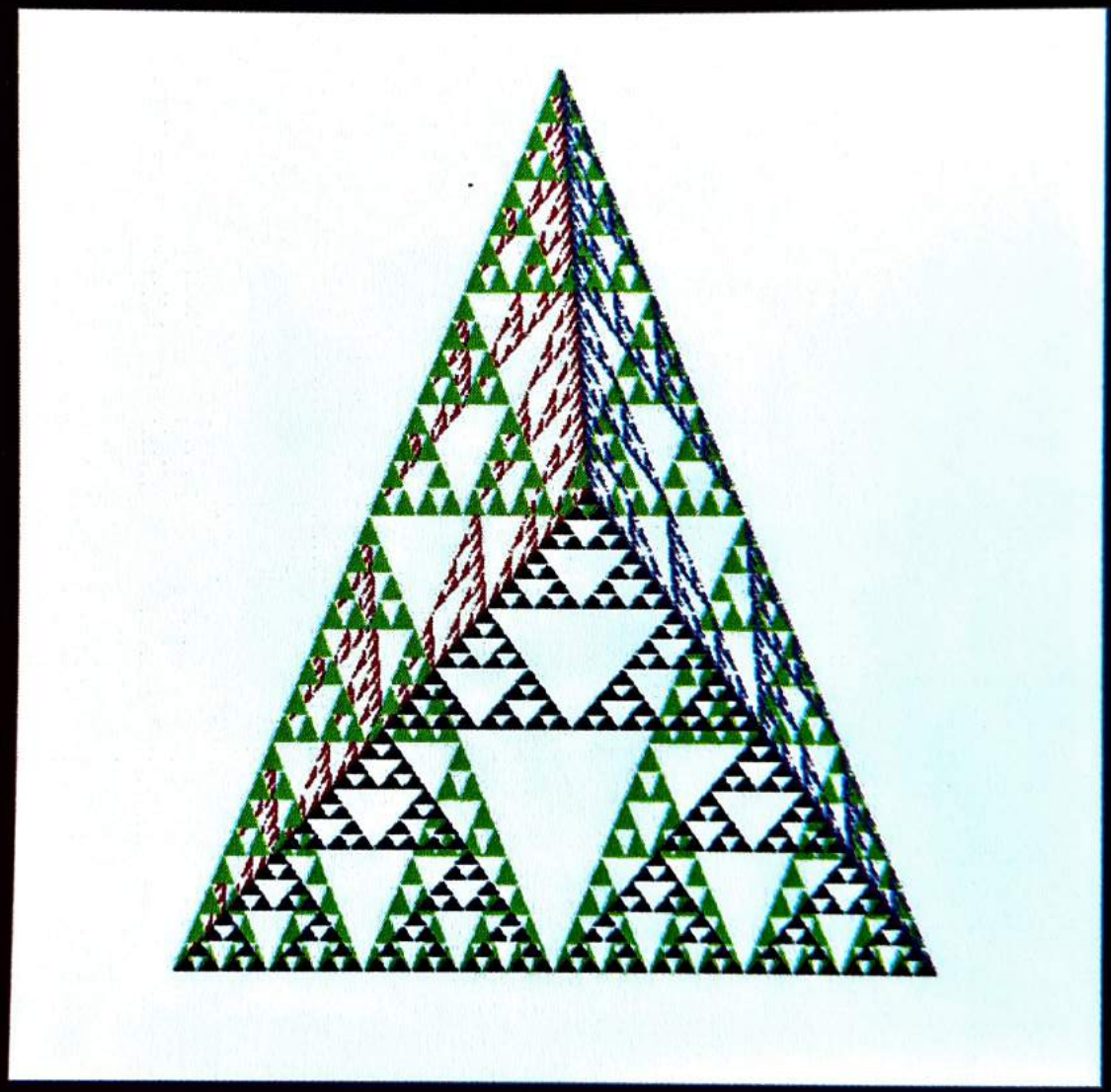
Date _____

Expt. No. _____ Page No. _____

```
int main( int argc, char **argv)
{
  n=5;
  glutInit ( &argc, argv);
  glutInitDisplayMode (GLUT_SINGE | GLUT_RGB);
  glutInitWindowSize ( 500,500);
  glutCreateWindow ("3D Gasket");
  glutDisplayFunc (display);
  myinit();
  glutMainLoop();
}
```

Teacher's Signature _____

```
mahesh@mahesh-VirtualBox:~/Desktop/cg$ cc program4.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg$ ./a.out
```



Date _____

Expt. No. 4 Page No. _____

Write a program to fill any given polygon using scan-line area filling algorithm.

```
# include <stdlib.h>
# include <GL/glut.h>
# include <algorithm>
# include <iostream>
# include <unistd.h>
using namespace std;
float x[100], y[100];
int n, m;
int wx = 500, wy = 500;
static float intx[10] = {0};

void draw_line(float x1, float y1, float x2, float y2) {
  sleep(100);
    glColor3f(1, 0, 0);
    glBegin(GL_LINES);
    glVertex2f(x1, y1);
    glVertex2f(x2, y2);
    glEnd();
    glFlush();
}
void edgeDetect(float x1, float y1, float x2, float y2, int scanline){
  float temp;
  if(y2 < y1){
      temp = x1; x1 = x2; x2 = temp;
      temp = y1; y1 = y2; y2 = temp;
  }
```

Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
    if( scanline > y1 && scanline < y2)
        intx[m++] = x1 + (scanline - y1)*(x2 - x1)/(y2 - y1);
}

void scanfill (float x[], float y[]){
    for(int s1 = 0, s1 <= wy; s1++){
        m = 0;
        for( int i=0; i<n; i++){
            edgeDetect (x[i], y[i], x[(i+1)%n], y[(i+1)%n], s1);
        }
        sort( intx, (intx + m));
        if( m >= 2)
            for( int i=0; i<m; i=i+2)
                draw_line (intx[i], s1, intx[i+1], s1);
    }
}

void display_filled_polygon(){
        glClear (GL_COLOR_BUFFET_BIT);
        glLineWidth (2);
        glBegin (GL_LINE_LOOP);
        for( int i=0; i<n; i++)
            glVertex2f (x[i], y[i]);
    glEnd();
    scanfill (x, y);
}

void myInit(){
        glClearColor( 1, 1, 1, 1);
```

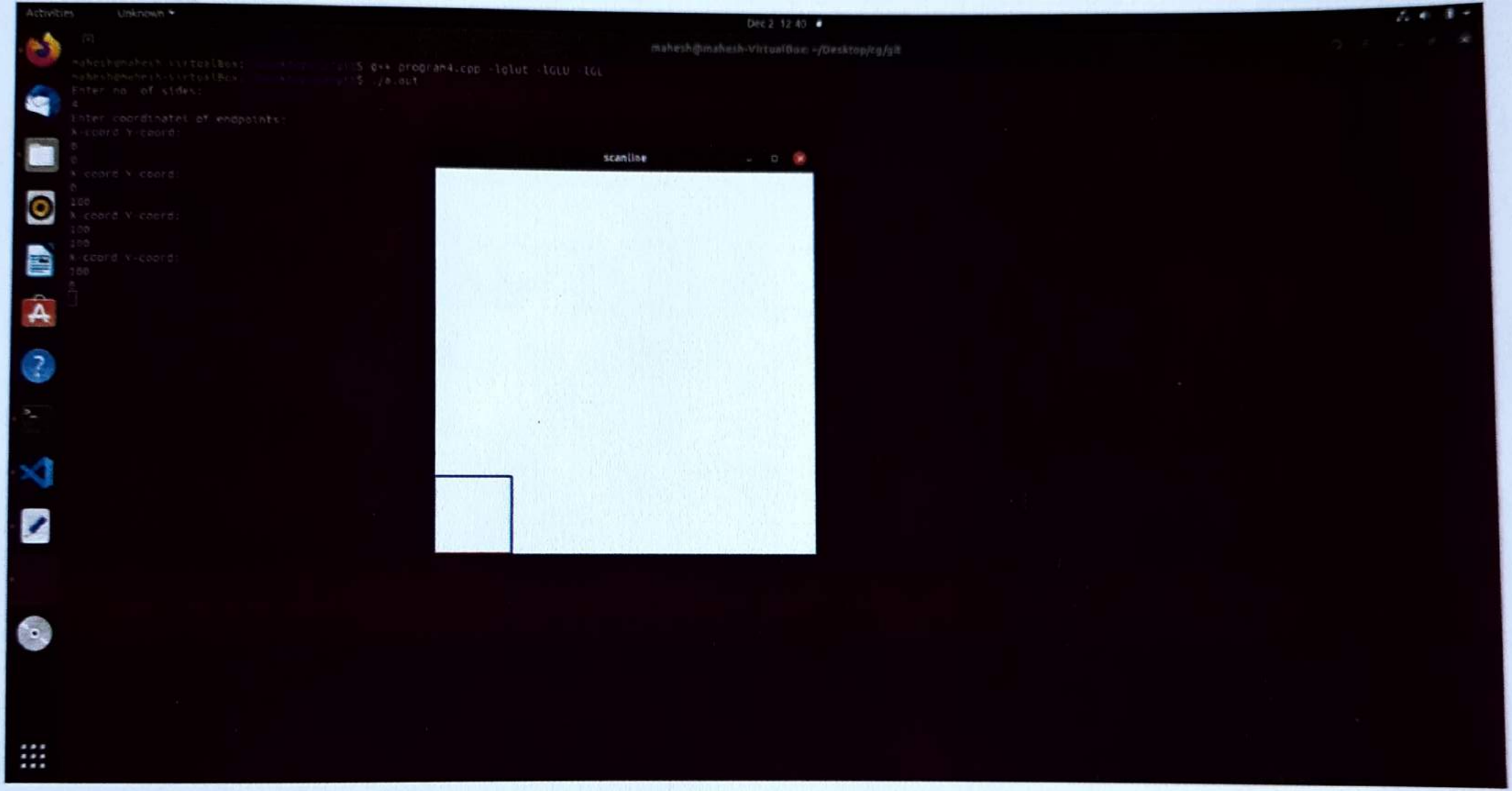
Teacher's Signature _____

Date

Expt. No. Page No.

```
    glColor3f (0,0,1);
    glPointSize (1);
    gluOrtho2D (0, wx, 0, wy);
}

int main ( int ac, char* av[] ) {
    glutInit ( &ac, av);
    printf ("Enter no. of sides : \n");
     scanf ("%d", &n);
    printf ("Eater coordinates of endpoints : \n");
for ( int i=0; i<n; i++)
  {
     printf ("x-coord  Y coord: \n");
     scanf ("%f %f", &x[i], &y[i] );
  }
 glutInitDisplayMode (GLUT_SINGLE |GLUT_RGB);
 glutInitWindowSize (500, 500);
 glutInitWindowPosition (0,0);
  glutCreateWindow ("scanline");
  glutDisplayFunc (display_filled_polygon);
  myInit ();
   glutMainLoop ();
}
```

Teacher's Signature


Activities
Unknown
Dec 2 12:40
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-virtualBox: $ g++ program4.cpp -lglut -lGLU -lGL
mahesh@mahesh-virtualBox: $ ./a.out
Enter no. of sides:
4
Enter coordinates of endpoints:
X-coord Y-coord:
0
0
X-coord Y-coord:
0
100
X-coord Y-coord:
100
100
X-coord Y-coord:
100
0
scanline

Date _____

Expt. No. 5 Page No. _____

Write a program to create a house like figure and perform the following operations

i) Rotate it about a given fixed point using OpenGL transformation functions.

ii) Reflect it about an axis $y = mx + c$ using OpenGL transformation functions

```
#include <GL/glut.h>
#include <math.h>
#include <stdlib.h>
#include <stdio.h>

float house[11][2] = {{100,200},{200,250},{300,200},{100,200},
                      {100,100},{175,100},{175,150},{225,150},
                      {225,100},{300,100},{300,200}};
int angle;
float m, c, theta;
void display() {
    glClearColor(1,1,1,0);
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_
            BIT);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(-450,450,-450,450);
    glMatrixMode(GL_MODELVIEW);
    glLoadIdentity();
    glColor3f(1,0,0);
    glBegin(GL_LINE_LOOP);
```

Teacher's Signature _____

Date _____

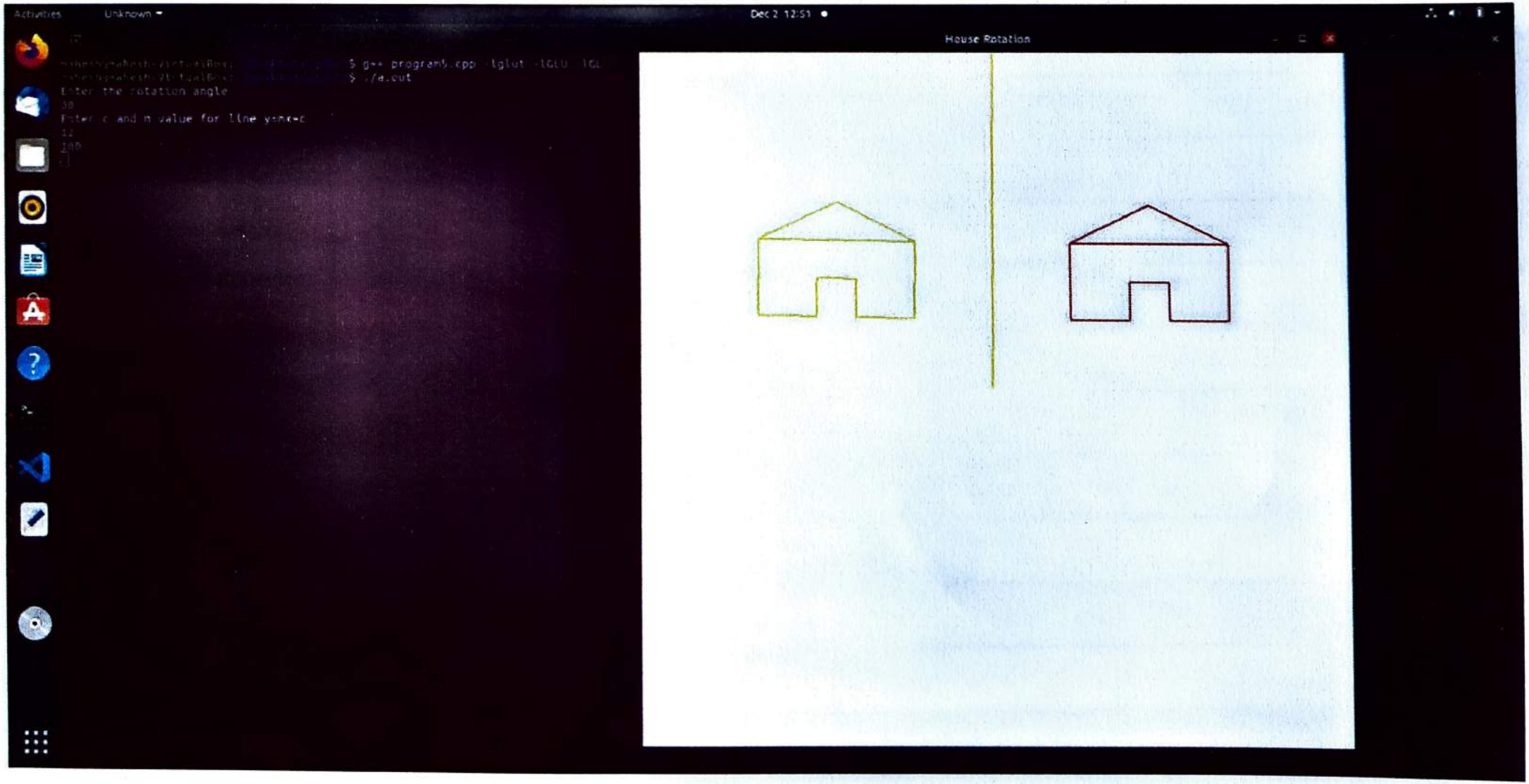
Expt. No. _____ Page No. _____

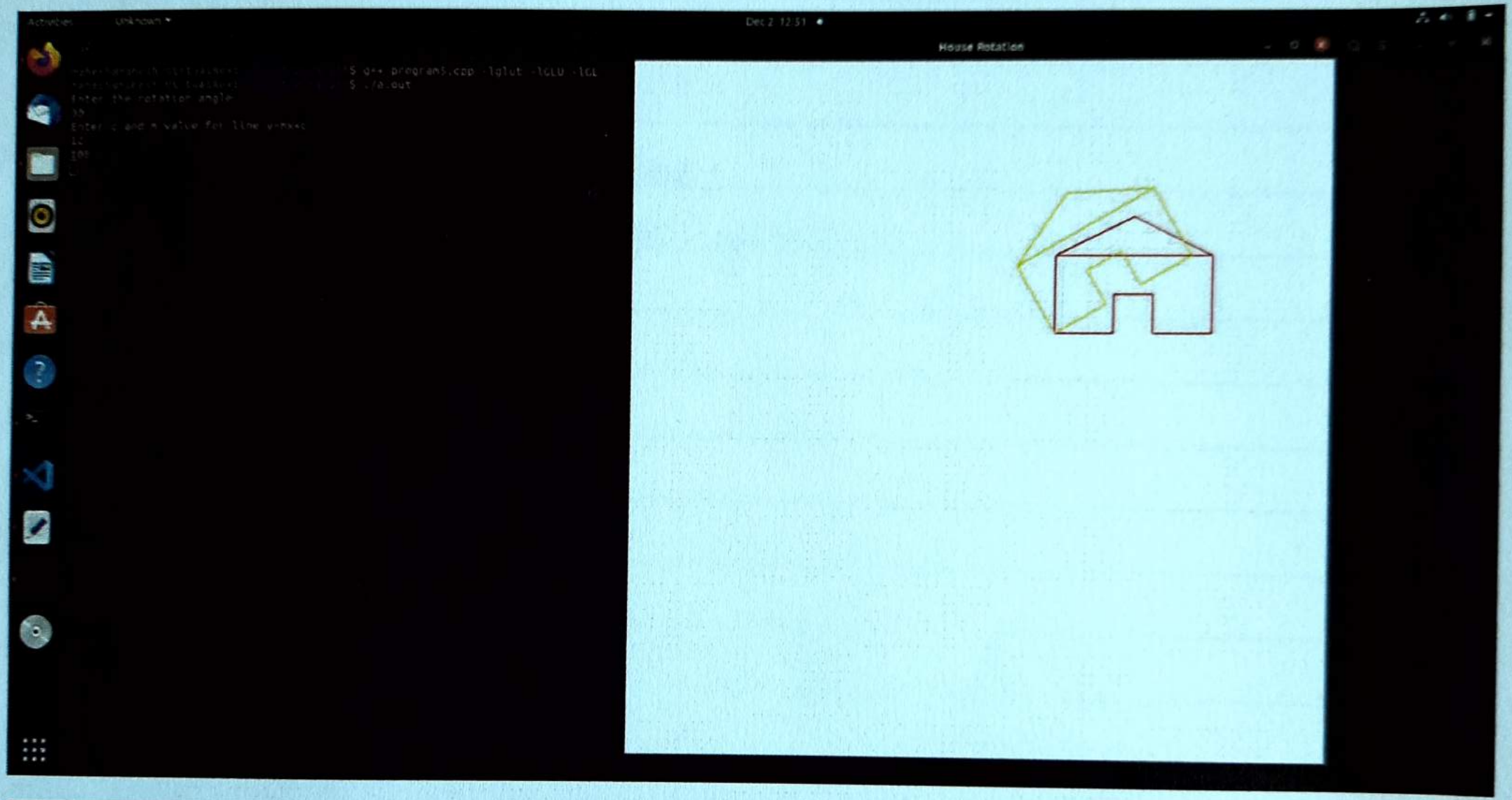
```
        for(int i=0; i<11; i++)
            glVertex2fv(house[i]);
        glEnd();
        glFlush();
        glPushMatrix();
        glTranslatef(100,100,0);
        glRotatef(angle,0,0,1);
        glTranslatef(-100,-100,0);
        glColor3f(1,1,0);
        glBegin(GL_LINE_LOOP)
        for(int i=0; i<11; i++)
            glVertex2fv(house[i]);
        glEnd();
        glPopMatrix();
        glFlush();
}
void display2(){
        glClearColor(1,1,1,0);
        glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_BUFFER_BIT);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(-450,450,-450,450);
        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
        glColor3f(1,0,0);
        glBegin(GL_LINE_LOOP);
        for(int i=0; i<11; i++)
            glVertex2fv(house[i]);
        glEnd();
```

Teacher's Signature _____

```
        glFlush();
        glPushMatrix();
        glTranslatef(0, c, 0);
        theta = atan(m);
        theta = theta * 180 / 3.14;
        glRotatef(theta, 0, 0, 1);
        glScalef(1, -1, 1);
        glRotatef(-theta, 0, 0, 1);
        glTranslate(0, -c, 0);
        glBegin(GL_LINE_LOOP);
        for(int i=0; i<11; i++)
            glVertex2fv(house[i]);
        glEnd();
        glPopMatrix();
        glFlush();
}
void myInit(){
        glClearColor(1.0, 1.0, 1.0, 1.0);
        glColor3f(1.0, 0.0, 0.0);
        glLineWidth(2.0);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(-450, 450, -450, 450);
}
void mouse(int btn, int state, int x, int y){
    if(btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN){
        display();
    }
    else if(btn == GLUT_RIGHT_BUTTON && state == GLUT_DOWN){
```

```
        display2();
    }
}
int main(int argc, char** argv)
{
    printf("Enter the rotation angle: \n");
    scanf("%d", &angle);
    printf("Enter c and m value for line y=mx+c \n");
    scanf("%f %f", &c, &m);
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(300, 300);
    glutInitWindowPosition(100,100);
    glutCreateWindow("House Rotation");
    glutDisplayFunc(display);
    glutMouseFunc(mouse);
    myInit();
    glutMainLoop();
}
```


Activities
Unknown
Dec 2 12:51
House Rotation
$ g++ program5.cpp -lglut -lGLU -lGL
$ ./a.out
Enter the rotation angle
30
Enter c and m value for line y=mx+c
12
100


Activities
Unknown
Dec 2 12:51
House Rotation
$ g++ programs.cpp -lglut -lGLU -lGL
$ ./a.out
Enter the rotation angle
30
Enter c and m value for line y=mx+c
12
100

Date _____

Expt. No. 6 Page No. _____

Write a program to implement the Cohen - Sutherland line clipping algorithm. Make provision to specify the input for multiple lines, window for clipping and viewport for displaying the clipped image.

```
# include < stdio.h >
# include < stdlib.h >
# include < GL/glut.h >
# define outcode int
# define true 1
# define false 0
double xmin, ymin, xmax, ymax;
double xvmin, yvmin, xvmax, yvmax;
constant int RIGHT = 4;
const int LEFT = 8;
const int TOP = 1;
const int BOTTOM = 2;
int n;
struct line_segment {
        int x1;
        int y1;
        int x2;
        int y2;
};
struct line_segment ls[10];

outcode computeoutcode (double x, double y)
{
   outcode code = 0;
    if ( y > ymax)
```

Teacher's Signature _____

Date _____

Expt. No. 8 Page No. _____

Write a program to implement the Cohen-Sutherland line clipping algorithm. Make provision to specify the input for multiple lines, window for clipping and viewport for displaying the clipped image.

```
            code |= TOP;
        else if ( y < ymin)
            code |= BOTTOM;
        if ( x > xmax)
            code |= RIGHT;
        else if ( x < xmin)
            code |= LEFT;

    return code;
}

void cohensuther(double x0, double y0, double x1, double y1)
{
 outcode outcode0, outcode1, outcodeout;
 bool accept = false, done = false;
outcode0 = computecode ( x0, y0);
outcode1 = computecode ( x1, y1);
do
 {
    if( !( outcode0 | outcode1))
     {
        accept = true;
        done = true;
     }
```

Teacher's Signature _____

```
else if ( outcode0 & outcode1)
        done = true;
else
{
    double x, y;
    outcodeout = outcode0 ? outcode0 : outcode1 ;
    if ( outcodeout & TOP)
    {
        x = x0 + (x1 - x0)* (ymax - y0) / (y1 - y0) ;
        y = ymax;
    }
    else if ( outcodeout & BOTTOM)
    {
        x = x0 + (x1 - x0)* (ymin - y0) / (y1 - y0);
        y = ymin;
    }
    else if ( outcodeout & RIGHT)
    {
        y = y0 + (y1 - y0)*(xmax - x0) /(x1 - x0);
        x = xmax;
    }
    else
    {
        y = y0 + (y1 - y0)* (xmin - x0) / (x1 - x0);
        x = xmin;
    }
    if ( outcodeout == outcode0)
    {       x0 = x;
            y0 = y;
```

Date _____

Expt. No. _____ Page No. _____

```
            outcode0 = computeoutcode (x0, y0);
        }
        else {
                x1 = x ;
                y1 = y;
                outcode1 = computeoutcode ( x1, y1);
            }
        }
} while (!done);

if(accept)
{
   double sx = (xvmax - xvmin) / (xmax - xmin);
   double sy = (yvmax - yvmin) / (ymax - ymin);
   double vx0 = xvmin + (x0 - xmin) * sx;
   double vy0 = yvmin + (y0 - ymin) * sy;
   double vx1 = xvmin + (x1 - xmin) * sx;
   double vy1 = yvmin + (y1 - ymin) * sy;
  glColor3f (1,0,0);
   glBegin (GL_LINE_LOOP);
   glVertex2f ( xvmin, yvmin );
   glVertex2f ( xvmax, yvmin);
   glVertex2f ( xvmax, yvmax);
   glVertex2f ( xvmin, yvmax);
   glEnd();
    glColor3f (0,0,1) ;
    glBegin (GL_LINES);
    glVertex2d ( vx0, vy0);
    glVertex2d ( vx1, vy1);
```

Teacher's Signature _____

```
    glEnd();
    }
}
void display()
{
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(0,0,1);
    glBegin(GL_LINE_LOOP);
    glVertex2f(xmin, ymin);
    glVertex2f(xmax, ymin);
    glVertex2f(xmax, ymax);
    glVertex2f(xmin, ymax);
    glEnd();
    for(int i=0; i<n; i++)
    {
        glBegin(GL_LINES);
        glVertex2d(ls[i].x1, ls[i].y1);
        glVertex2d(ls[i].x2, ls[i].y2);
        glEnd();
    }
for(int i=0; i<n; i++)
    cohensuther(ls[i].x1, ls[i].y1, ls[i].x2, ls[i].y2);

    glFlush();
}
void myinit(){
    glClearColor(1,1,1,1);
    glColor3f(1,0,0);
    glPointSize(1.0);
```

```
    glMatrixMode (GL_PROJECTION);
    glLoadIdentity ();
    gluOrtho2D (0, 500, 0, 500);
}
int main (int argc, char** argv)
{
    printf ("Enter window coordinates (xmin ymin xmax ymax): \n");
    scanf ("%lf %lf %lf %lf", &xmin, &ymin, &xmax, &ymax);
    printf ("Enter viewport coordinates (vxmin vymin vxmax vymax): \n");
    scanf ("%lf %lf %lf %lf", &vxmin, &vymin, &vxmax, &vymax);
    printf ("Enter no. of lines : \n");
    scanf ("%d", &n);
    for (int i=0; i<n; i++) {
        printf ("Enter line endpoints (x1 y1 x2 y2): \n");
        scanf ("%d %d %d %d", &ls[i].x1, &ls[i].y1, &ls[i].x2,
                              &ls[i].y2);
    }
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_SINGE | GLUT_RGB);
    glutInitWindowSize (500, 500);
    glutInitWindowPosition (0,0);
    glutCreateWindow ("clip");
    myInit ();
    glutDisplayFunc (display);
    glutMainLoop ();
}
```

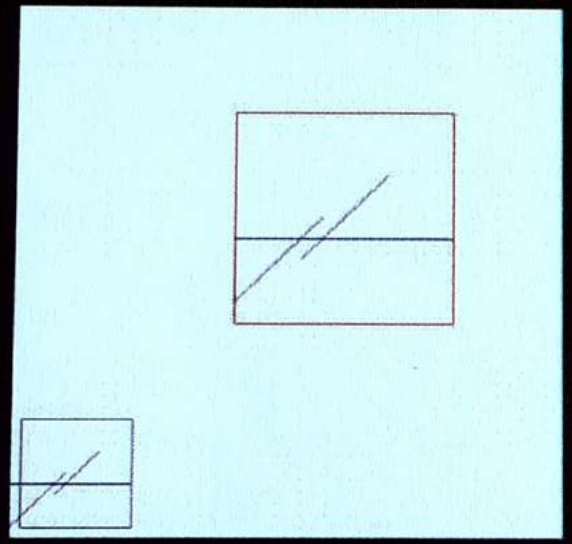
Activities Unknown

Dec 2 12:50

mahesh@mahesh-VirtualBox: ~/Desktop/cg/git

```
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program6.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
Enter window coordinates (xmin ymin xmax ymax):
10 10 110 110
Enter viewport coordinates (xvmin yvmin xvmax yvmax) :
200 200 400 400
Enter no. of lines:
3
Enter line endpoints (x1 y1 x2 y2):
0 10 50 60
Enter line endpoints (x1 y1 x2 y2):
0 50 110 50
Enter line endpoints (x1 y1 x2 y2):
40 40 80 80
```

clip



Date _____

Expt. No. 7 Page No. _____

Write a program to implement the Liang-Barsky line clipping algorithm. Make provision to specify the input for multiple lines, window for clipping and viewport for displaying the clipped image.

```
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
double xmin, ymin, xmax, ymax;
double vxmin, vymin, vxmax, vymax;
int n;
struct line_segment {
        int x1;
        int y1;
        int x2;
        int y2;
};
struct line_segment ls[10];
int clipTest (double p, double q, double* u1, double* u2)
{
  double r;
  if (p)
      r = q/p;
  if (p < 0.0) {
          if (r > *u1) *u1 = r;
          if (r > *u2) return (false);
        }
  else
        if (p > 0.0) {
                if (r < *u2) *u2 = r;
```

Teacher's Signature _____

```
            if(r < *u1) return (false);
        }
        else
            if( r == 0.0)
            {
                if(q < 0.0) return (false);
            }
    return (true);
}

void LiangBarskyLineClipAndDraw(double x0, double y0, double x1, double y1)
{
 double dx = x1 - x0, dy = y1 - y0 , u1 = 0.0 , u2 = 1.0;
 glColor3f (1.0, 0.0, 0.0);
 glBegin ( GL_LINE_LOOP);
 glVertex2f ( xwmin, ywmin);
 glVertex2f ( xwmax, ywmin);
 glVertex2f ( xwmax, ywmax);
 glVertex2f ( xwmin, ywmax );
 glEnd();
 if( cliptest (-dx, x0 - xwmin , &u1 , &u2))
   if (cliptest ( dx, xwmax - x0, &u1, &u2))
      if (cliptest (-dy, y0 - ywmin , &u1, &u2))
        if( cliptest( dy, ywmax - y0, &u1, &u2))
          {
            if (u2 < 1.0)
              {
                x1 = x0 + u2 * dx;
                y1 = y0 + u2 * dy;
              }
```

Teacher's Signature _____

```
        if (u1 > 0.0){
                x0 = x0 + u1 * dx;
                y0 = y0 + u1 * dy;
            }
    double sx = (xvmax - xvmin) / (xmax - xmin);
    double sy = (yvmax - yvmin) / (ymax - ymin);
    double vx0 = xvmin + (x0 - xmin) * sx;
    double vy0 = yvmin + (y0 - ymin) * sy;
    double vx1 = xvmin + (x1 - xmin) * sx;
    double vy1 = yvmin + (y1 - ymin) * sy;
     glColor3f(0.0, 0.0, 1.0);
     glBegin(GL_LINES);
     glVertex2d(vx0, vy0);
     glVertex2d(vx1, vy1);
     glEnd();
    }
}
void display(){
        glClear(GL_COLOR_BUFFER_BIT);
        glColor3f(1.0, 0.0, 0.0);
        for(int i=0; i<n; i++)
        {
            glBegin(GL_LINES);
            glVertex2d(ls[i].x1, ls[i].y1);
            glVertex2d(ls[i].x2, ls[i].y2);
            glEnd();
        }
        glColor3f(0.0, 0.0, 1.0);
        glBegin(GL_LINE_LOOP);
```

Date _____

Expt. No. _____ Page No. _____

```
        glVertex2f (xmin, ymin);
        glVertex2f (xmax, ymin);
        glVertex2f (xmax, ymax);
        glVertex2f (xmin, ymax);
        glEnd();
      for (int i=0; i<n; i++)
        LiangBarskyLineClipAndDraw (ls[i].x1, ls[i].y1,
                                    ls[i].x2, ls[i].y2);
      glFlush();
  }

void myinit()
{
    glClearColor(1.0,1.0,1.0,1.0);
    glColor3f (1.0, 0.0, 0.0);
    glLineWidth (2.0);
    glMatrixMode (GL_PROJECTION);
    glLoadIdentity ();
    gluOrtho2D (0.0, 499.0, 0.0, 499.0);
}
void main (int argc, char** argv)
{
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_SINGE | GLUT_RGB);
    glutInitWindowSize (500, 500);
    glutInitWindowPosition (0,0);
  printf ("Enter window coordinates : (xmin ymin xmax ymax) \n");
scanf ("%lf %lf %lf %lf", &xmin, &ymin, &xmax, &ymax);
printf ("Enter viewport coordinates : (xmin ymin xmax ymax) \n");
```

Teacher's Signature _____

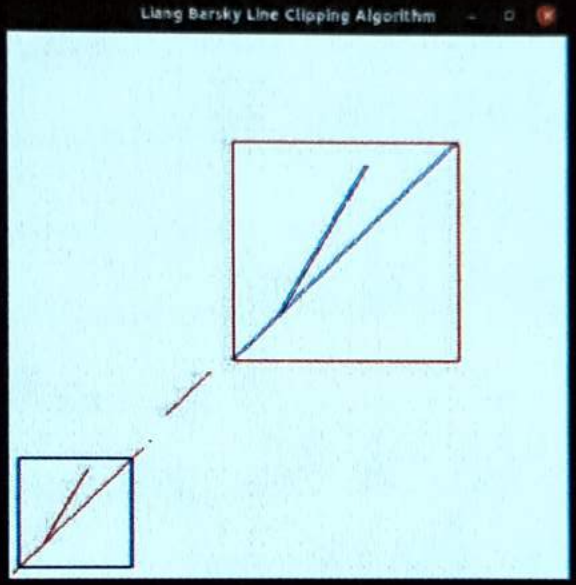
Date _____

Expt. No. _____ Page No. _____

```
scanf("%lf %lf %lf %lf", &xvmin, &yvmin, &xvmax, &yvmax);
printf("Enter no. of lines : \n");
scanf("%d", &n);
for(int i=0; i<n; i++)
    {
    printf("Enter coordinates : (x1, y1 x2 y2) \n");
    scanf("%d %d %d %d", &ls[i].x1, &ls[i].y1, &ls[i].x2, &ls[i].y2);
    }
glutCreateWindow("Liang Barsky Line Clipping Algorithm");
glutDisplayFunc(display);
myinit();
glutMainLoop();
}
```

Teacher's Signature _____

```
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program7.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
Enter window coordinates: (xmin ymin xmax ymax)
30 30 110 110
Enter viewport coordinates: (xvmin yvmin xvmax yvmax)
200 200 400 400
Enter no. of lines:
3
Enter coordinates: (x1 y1 x2 y2)
30 30 70 100
Enter coordinates: (x1 y1 x2 y2)
140 150 180 190
Enter coordinates: (x1 y1 x2 y2)
5 5 120 120
```



Date _____

Expt. No. 8 Page No. _____

Write a program to implement the Cohen - Hodgman polygon clipping algorithm. Make provision to specify the input polygon and window for clipping.

```
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
using namespace std;
int poly_size, poly_points[20][2], org_poly_
  size, org_poly_points[20][2],
  clipper_size, clipper_points[20][2];
const int MAX_POINTS = 20;
void drawPoly (int p[][2], int n) {
  glBegin (GL_POLYGON);
 for(int i=0; i<n; i++)
   glVertex2f (p[i][0], p[i][1]);
   glEnd();
}

int x_intersect (int x1, int y1, int x2,
      int y2, int x3, int y3, int x4,
      int y4) {
int num = (x1*y2 - y1*x2)*(x3 - x4)
    -(x1 - x2)*(x3*y4 - y3*x4);

int den = (x1 - x2)*(y3 - y4) -
      (y1 - y2)*(x3 - x4);
return num/den;
}

int y_intersect (int x1, int y1, int x2, int y2,
      int x3, int y3, int x4, int y4)
{
int num = (x1*y2 - y1*x2)*(y3 - y4) -
    (y1 - y2)*(x3*y4 - y3*x4);
int den = (x1 - x2)*(y3 - y4) -
      (y1 - y2)*(x3 - x4);
return num/den;
}

void clip(int poly_points[][2], int &poly_size,
  int x1, int y1, int x2, int y2 )
{
int new_points[MAX_POINTS][2],
    new_poly_size = 0;
for(int i=0; i<poly_size ; i++)
{
 int k = (i+1) % poly_size ;
int ix = poly_points[i][0], iy = poly_points[i][1];
int kx = poly_points[k][0], ky = poly_points[k][1];
int i_pos = (x2 - x1)*(iy - y1) -
      (y2 - y1)*(ix - x1);
int k_pos = (x2 - x1)*(ky - y1) -
      (y2 - y1)*(kx - x1);
```

Teacher's Signature _____

```
if(i_pos>0=0 && k_pos >=0)
{
   new_points [new_poly_size][0]= kx;
   new_points [new_poly_size][1]= ky;
   new_poly_size ++;
}
else if( i_pos<0 && k_pos >=0)
      {
            new_points[new_poly_size][0]= x_intersect (x1, y1, x2, y2, ix, iy, kx,
                                                       ky);

            new_points [new_poly_size][1]=y_intersect (x1, y1, x2, y2, ix, iy, kx, ky);
            new_poly_size ++;
            new_points [new_poly_size][0]= ksc;
            new_points [new_poly_size][1]= ky;
            new_poly_size ++;
      }
    else if ( i_pos>=0 && k_pos<0)
    {
       new_points [new_poly_size][0]= x_intersect (x1, y1, x2, y2, ix, iy, kx, ky);
       new_points [new_poly_size][1]= y_intersect (x1, y1, x2, y2, ix, iy, kx, ky);
       new_poly_size ++;
    }
    else { // No points are added }
}
    poly_size = new_poly_size;
       for( int i=0; i< poly_size ; i++)
            {
               poly_points [i][0]= new_points [i][0];
               poly_points [i][1]= new_points [i][1];
            }
}
```

Date _____

Expt. No. _____ Page No. _____

```
void init() {
    glClearColor(0.0f, 0.0f, 0.0f, 0.0f);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    glOrtho(0.0, 500.0, 0.0, 500.0, 0.0, 500.0);
    glClear(GL_COLOR_BUFFER_BIT);
}
void display() {
    init();
    glColor3f(1.0f, 0.0f, 0.0f);
    drawPoly(clipper_points, clipper_size);
    glColor3f(0.0f, 1.0f, 0.0f);
    drawPoly(org_poly_points, org_poly_size);
  for(int i = 0; i < clipper_size; i++)
   {
      int k = (i+1) % clipper_size;
 clip(poly_points, poly_size, clipper_points[i][0], clipper_points[i][1],
    clipper_points[k][0], clipper_points[k][1]);
   }
  glColor3f(0.0f, 0.0f, 1.0f);
   drawPoly(poly_points, poly_size);
   glFlush();
}

int main(int argc, char *argv[])
{
  printf("Enter no. of vertices : \n");
   scanf("%d", &poly_size);
    org_poly_size = poly_size;
```

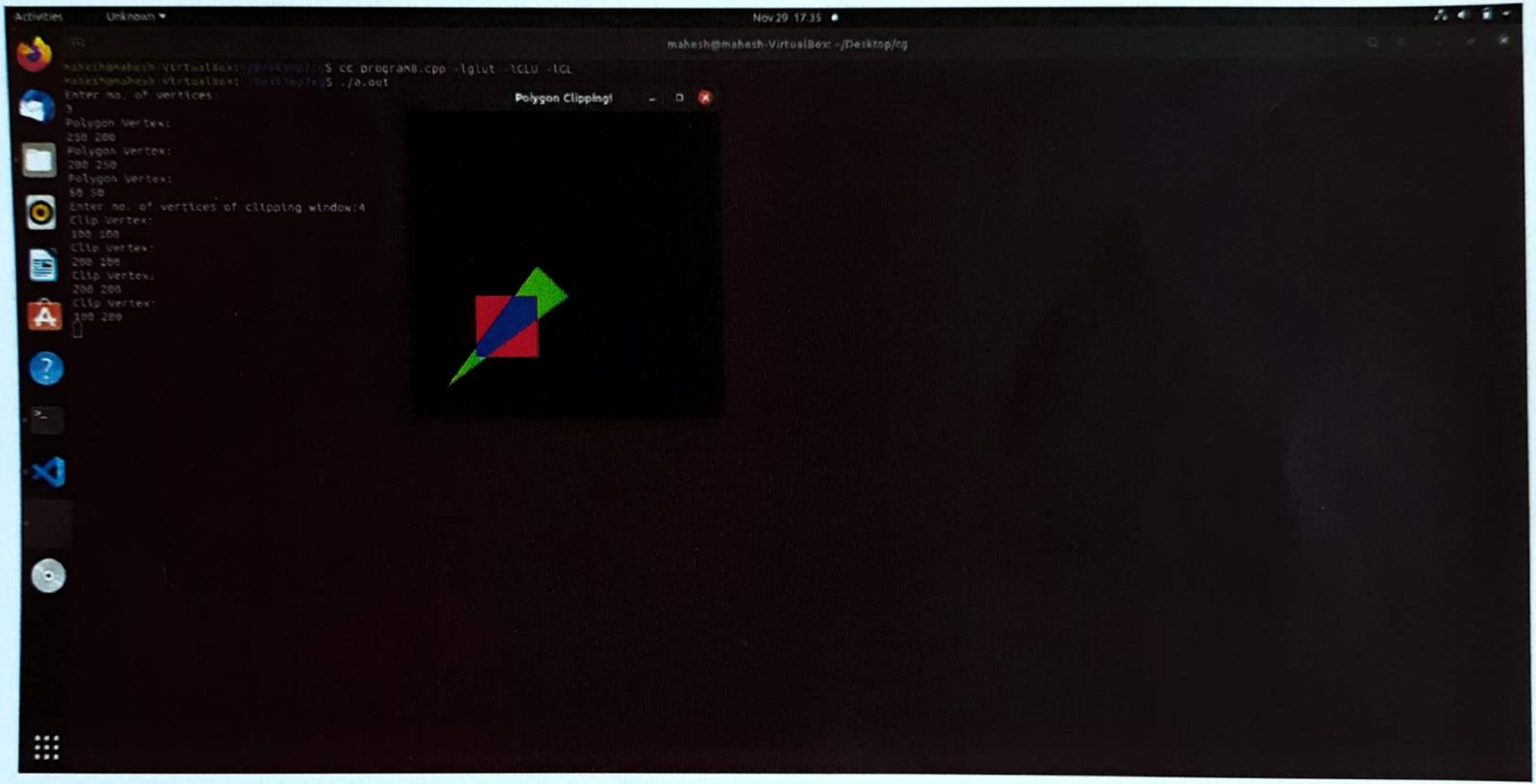
Teacher's Signature _____

Date _____

Expt. No. _____ Page No. _____

```
for(int i=0; i< poly_size; i++)
 {
    printf("Polygon Vertex");
    scanf("%d %d", &poly_points[i][0], &poly_points[i][1]);
    org_poly_points[i][0] = poly_points[i][0];
    org_poly_points[i][1] = poly_points[i][1];
 }
printf("Enter no. of vertices of clipping window : ");
scanf("%d", &clipper_size);
for(int i=0; i< clipper_size; i++)
 {  printf("Clip Vertex : \n");
    scanf("%d %d", &clipper_points[i][0], &clipper_points[i][1]);
 }
glutInit(&argc, argv);
glutInitDisplayMode (GLUT_SINGE | GLUT_RGB);
glutInitWindowSize(400,400);
glutInitWindowPosition(100,100);
glutCreateWindow("Polygon Clipping");
glutDisplayFunc(display);
glutMainLoop();
return 0;
}
```

Teacher's Signature _____


Activities
Unknown
Nov 29 17:35
mahesh@mahesh-VirtualBox: ~/Desktop/cg
mahesh@mahesh-VirtualBox:~/Desktop/cg$ cc program8.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg$ ./a.out
Enter no. of vertices:
3
Polygon Vertex:
230 200
Polygon Vertex:
200 250
Polygon Vertex:
50 50
Enter no. of vertices of clipping window:4
Clip Vertex:
100 100
Clip Vertex:
200 100
Clip Vertex:
200 200
Clip Vertex:
100 200
Polygon Clipping!

Date _____

Expt. No. 9 Page No. _____

Write a program to model a car like figure using display lists and move a car from one end of the screen to other end. User is able to control speed with mouse.

```
#include <GL/glut.h>
#include <math.h>
#include <stdio.h>
#define CAR 1
#define WHEEL 2
float s=1;
 void carlist(){
            glNewList(CAR, GL_COMPILE);
            glColor3f(1,1,1);
            glBegin(GL_POLYGON);
            glVertex3f(0,25,0);
            glVertex3f(90,25,0);
            glVertex3f(90,55,0);
            glVertex3f(80,55,0);
            glVertex3f(20,75,0);
            glVertex3f(0,55,0);
            glEnd();
            glEndList();
    }
void wheellist(){
          glNewList(WHEEL, GL_COMPILE_AND_EXECUTE);
          glColor3f(0,1,1);
          glutSolidSphere(10,25,25);
          glEndList();
       }
```

Teacher's Signature _____

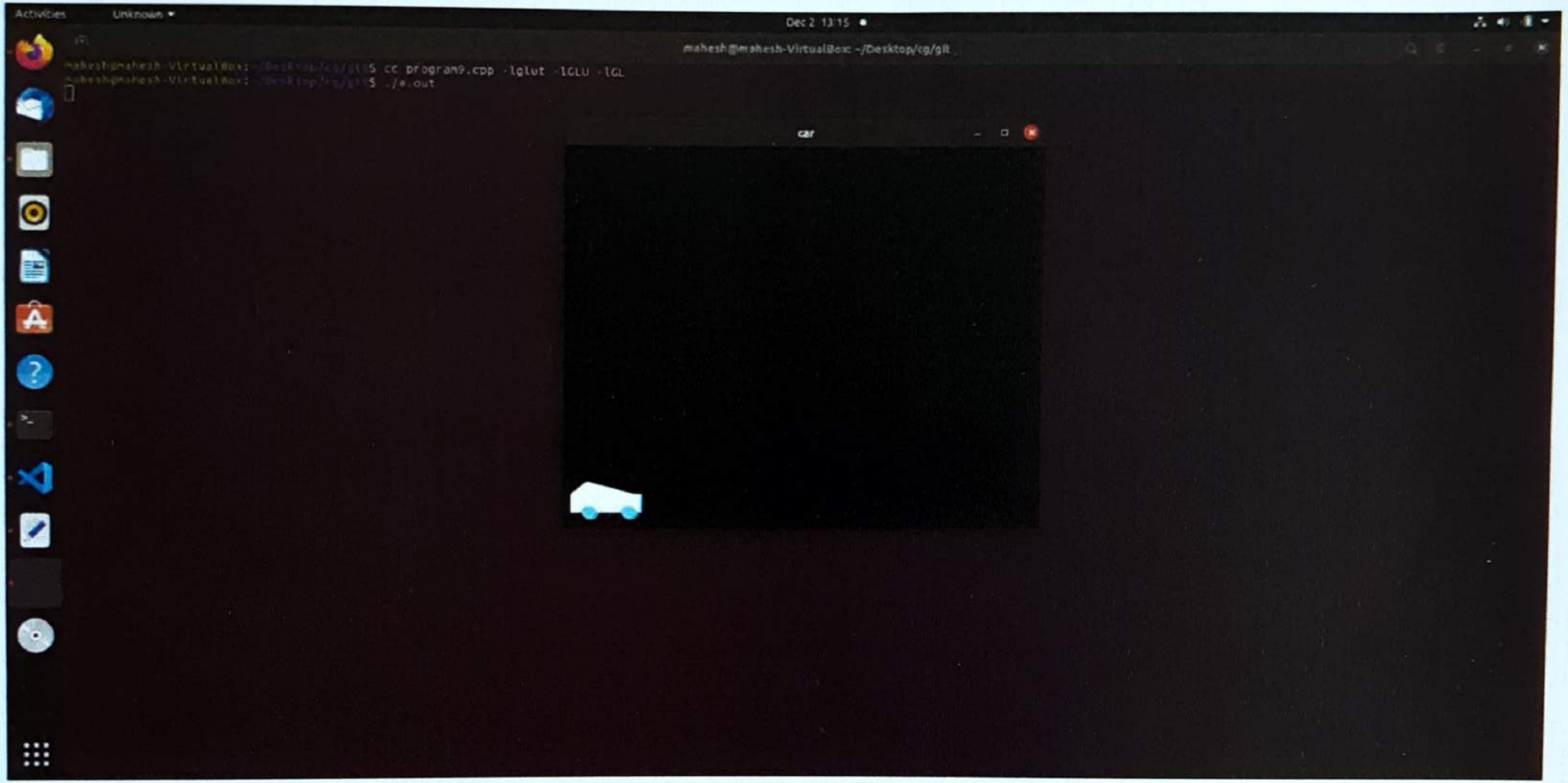
```
void mykeyboard (unsigned char key, int x, int y) {
  switch (key) {
      case 't' : glutPostRedDisplay();
                 break;
     case 'q' : exit (0);
      default :  break;
     }
  }
void myInit() {
          glClearColor(0,0,0,0);
          glOrtho (0, 600, 0, 600, 0, 600);

void draw_wheel() {
            glColor3f (0,1,1);
            glutSolidSphere (10,25,25);
        }
void moveCar( float s) {
             glTranslatef (s, 0.0, 0.0);
             glCallList (CAR);
             glPushMatrix();
             glTranslatef (25, 25, 0.0);
             glCallList (WHEEL);
             glPopMatrix();
             glPushMatrix();
             glTranslatef (75, 25, 0.0);
             glCallList (WHEEL);
             glPopMatrix();
              glFlush();
      }
```

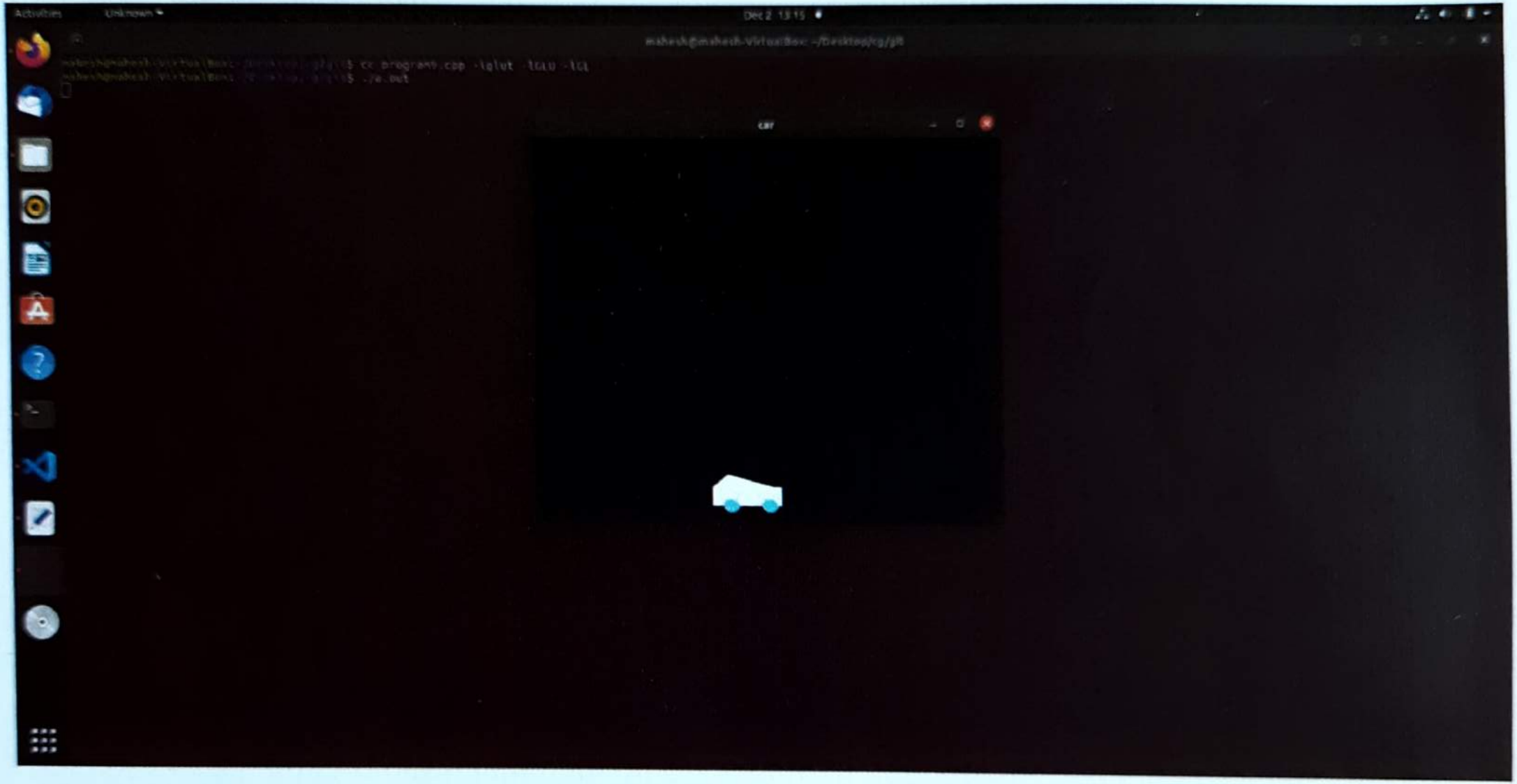
Date _____

Expt. No. _____ Page No. _____

```
void myDisp(){   glClear(GL_COLOR_BUFFER_BIT);
                 carlist();
                 moveCar(s);
                 wheellist();
             }
void mouse(int btn, int state, int x, int y){
      if(btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN){
                 s += 5;
                 myDisp(); }
      else if (btn = GLUT_RIGHT_BUTTON && state == GLUT_DOWN){
                 s += 2;
                 myDisp();
             }
}
int main(int argc, char* argv[]){
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
  glutInitWindowSize(600, 500);
  glutInitWindowPosition(100, 100);
  glutCreateWindow("car");
  myInit();
  glutDisplayFunc(myDisp);
  glutMouseFunc(mouse);
  glutKeyboardFunc(mykeyboard);
  glutMainLoop();
}
```

Teacher's Signature _____


Activities
Unknown
Dec 2 13:15
mahesh@mahesh-VirtualBox: ~/Desktop/cg/glt
mahesh@mahesh-VirtualBox:~/Desktop/cg/glt$ cc program9.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg/glt$ ./a.out
car


Activities
Unknown
Dec 2 13:15
car
$ cc program9.cpp -lglut -lGLU -lGL
$ ./a.out

Date _____

Expt. No. 10 Page No. _____

Write a gfx program to create a color cube and spin it using OpenGL transformations.

```
#include <stdlib.h>
#include <GL/glut.h>
#include <stdio.h>
#include <time.h>
GLfloat vertices[] = {-1.0,-1.0,-1.0, 1.0,-1.0,-1.0, 1.0,1.0,-1.0, -1.0,1.0,-1.0,
                      -1.0,-1.0,1.0, 1.0,-1.0,1.0, 1.0,1.0,1.0, -1.0,1.0,1.0};
GLfloat normals[] = {-1.0,-1.0,-1.0, 1.0,-1.0,-1.0, 1.0,1.0,-1.0, -1.0,1.0,-1.0,
                     -1.0,-1.0,1.0, 1.0,-1.0,1.0, 1.0,1.0,1.0, -1.0,1.0,1.0};
GLfloat colors[] = {0.0,0.0,0.0, 1.0,0.0,0.0, 1.0,1.0,0.0, 0.0,1.0,0.0,
                    0.0,0.0,1.0, 1.0,0.0,1.0, 1.0,1.0,1.0, 0.0,1.0,1.0};
GLcubeIndices[] = {0,3,2,1,2,3,7,6,0,4,7,3,1,2,6,5,4,5,6,7,0,1,5,4};
static GLfloat theta[] = {0.0,0.0,0.0};
static GLfloat beta[] = {0.0,0.0,0.0};
static GLint axis = 2;
void delay(float secs){
    float end = clock() / CLOCKS_PER_SEC + secs;
    while((clock() / CLOCK_PER_SEC) < end);
}
void displaySingle(void){
        glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
        glLoadIdentity();
        glRotatef(theta[0], 1.0,0.0,0.0);
        glRotatef(theta[1]; 0.0,1.0,0.0);
        glRotatef(theta[2]; 0.0,0.0,1.0);
        glDrawElements(GL_QUADS,24,GL_UNSIGNED_BYTE,cubeIndices);
        glBegin(GL_LINES);
```

Teacher's Signature _____

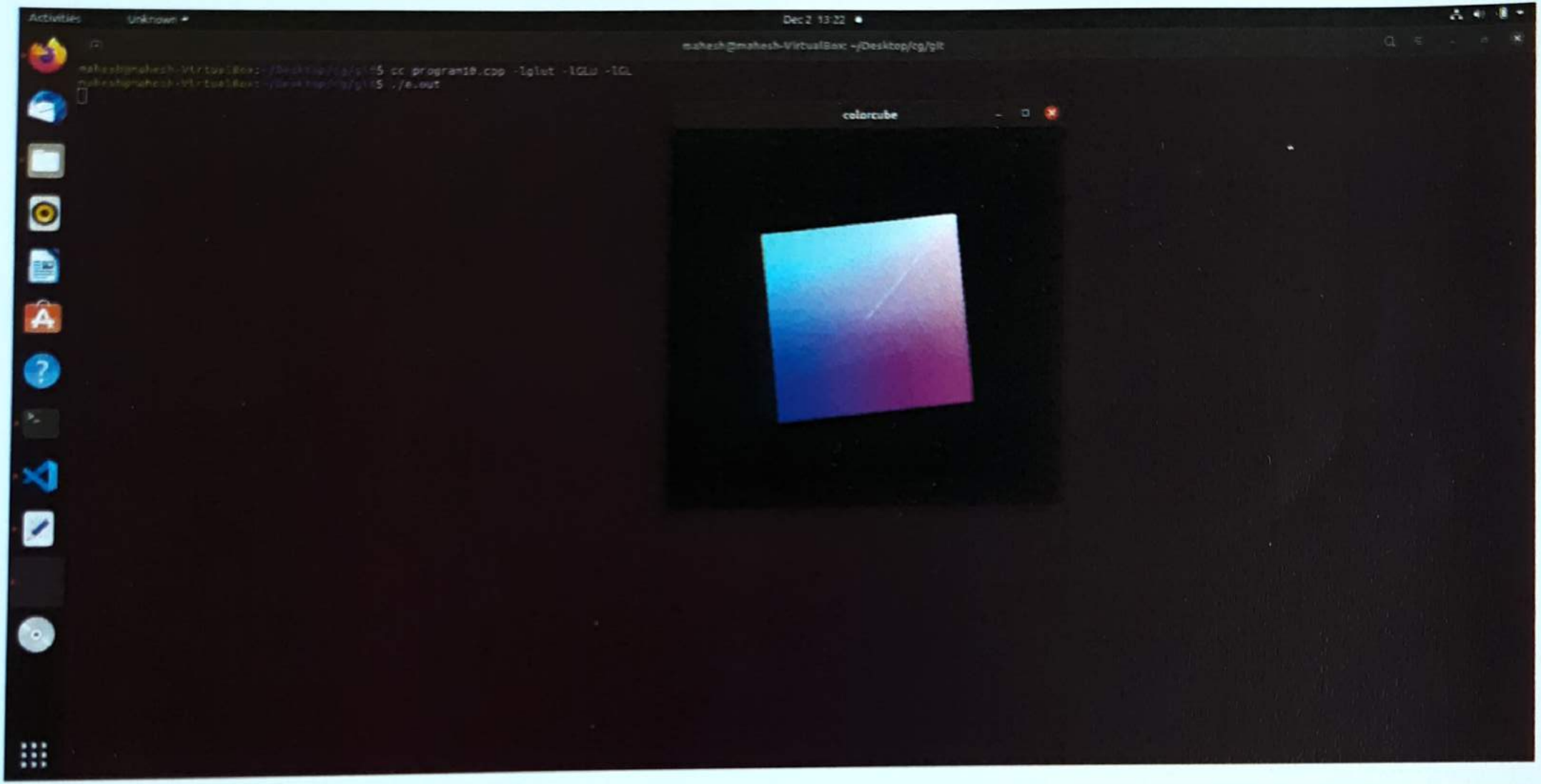
```
    glVertex3f(0.0,0.0,0.0);
    glVertex3f(1.0, 1.0, 1.0);
    glEnd();
    glFlush();
}
    void spinCube() {
            delay(0.01);
            theta[axis] += 2.0;
          if(theta[axis] > 360.0) theta[axis] -= 360.0;
          glutPostRedisplay();
        }
void mouse(int btn, int state, int x, int y) {
    if(btn == GLUT_LEFT_BUTTON && state == GLUT_DOWN) axis = 0;
  if (btn == GLUT_MIDDLE_BUTTON && state == GLUT_DOWN) axis = 1;
 if (btn == GLUT_RIGHT_BUTTON && state == GLUT_DOWN) axis = 2;
}
void myReshape(int w, int h)
{
    glViewport(0,0, w, h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
     if(w <= h)
          glOrtho(-2.0, 2.0, -2.0* (GLfloat)h / (GLfloat)w,
                  2.0 *(GLfloat)h /(GLfloat)w, -10.0, 10.0);
     else
          glOrtho( -2.0* (GLfloat)w/ (GLfloat)h,
                  2.0* (GLfloat)w/ (GLfloat)h, -2.0, 2.0, -10.0,10.0);
        glMatrixMode(GL_MODELVIEW);
}
```

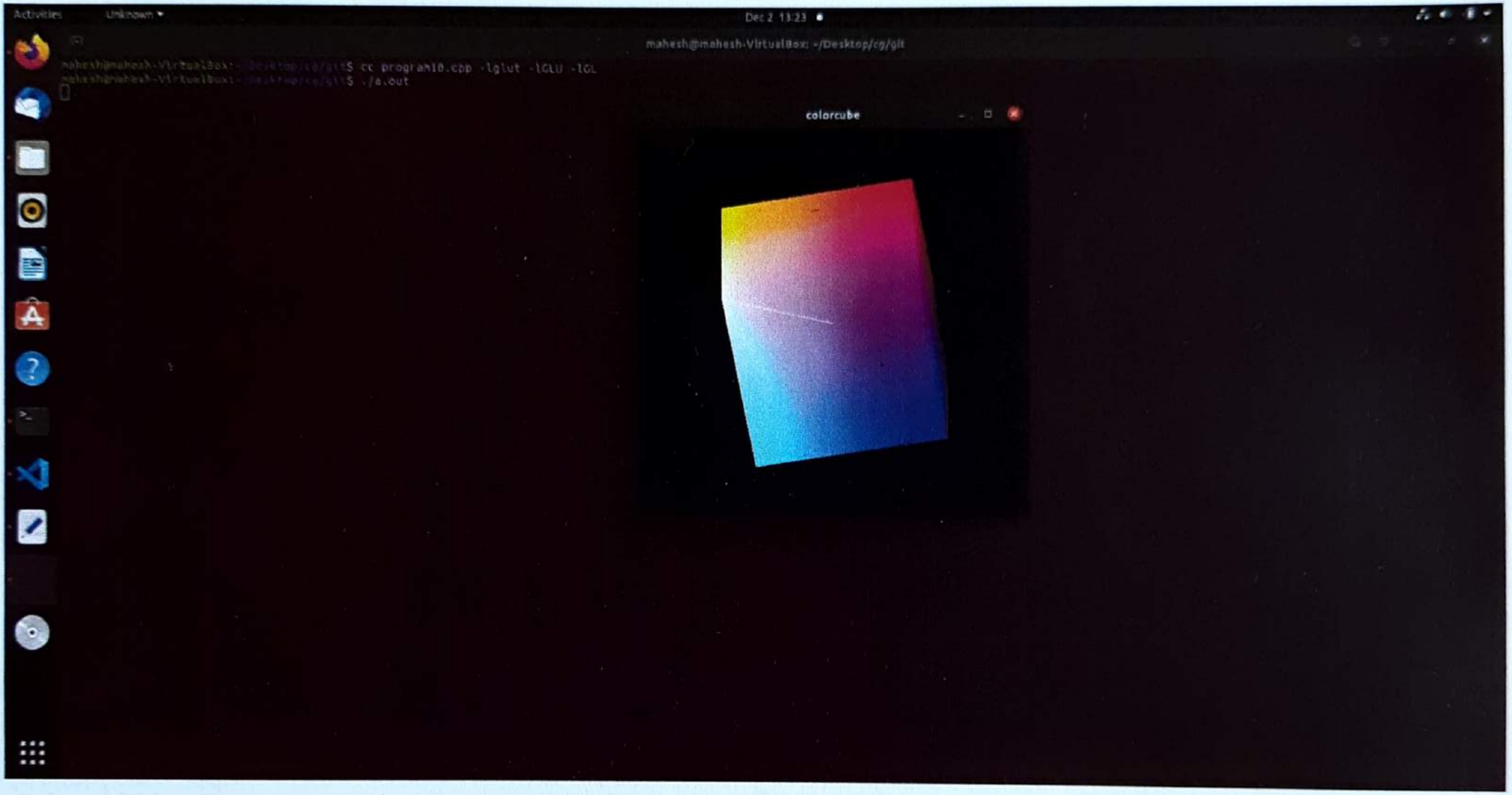
Date _____

Expt. No. _____ Page No. _____

```
int main(int argc, char** argv){
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowPosition(100,100);
    glutInitWindowSize(500,500);
    glutCreateWindow("colorcube");
    glutReshapeFunc(myReshape);
    glutDisplayFunc(displaySingle);
    glutIdleFunc(spinCube);
    glutMouseFunc(mouse);
    glEnable(GL_DEPTH_TEST);
    glEnableClientState(GL_COLOR_ARRAY);
    glEnableClientState(GL_NORMAL_ARRAY);
    glEnableClientState(GL_VERTEX_ARRAY);
    glVertexPointer(3,GL_FLOAT,0,vertices);
    glColorPointer(3,GL_FLOAT,0,colors);
    glNormalPointer(GL_FLOAT,0,normals);
    glColor3f(1.0,1.0,1.0);
    glutMainLoop();
}
```

Teacher's Signature _____


Activities
Unknown
Dec 2 13:22
mahesh@mahesh-VirtualBox: ~/Desktop/cg/glt
mahesh@mahesh-VirtualBox:~/Desktop/cg/glt$ cc program10.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg/glt$ ./a.out
colorcube


Activities
Unknown
Dec 2 13:23
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program10.cpp -lglut -lGLU -lGL
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
colorcube

Date _____

Expt. No. 11 Page No. _____

Create a menu with three entries named curves, colors and quit. The entry curves has a sub menu which has four entries namely Limacon, Cardiod, Three-leaf and spiral. The color menu has sub menu with all eight colors of RGB color model. Write a program to create the above hierarchical menu and attach appropriate services to each menu entries with mouse buttons.

```
#include <GL/glut.h>
#include <stdlib.h>
#include <math.h>
#include <stdio.h>
struct screenPt {int x,
                 int y;
                };
typedef enum {limacon = 1, cardoid = 2, threeleaf = 3, spiral = 4}
                curveName;
int w = 600, h = 500;
int curve = 1;
int red = 0, green = 0, blue = 0;
void myinit (void) {
        glClearColor (1.0, 1.0, 1.0, 1.0);
        glMatrixMode (GL_PROJECTION);
        gluOrtho2D (0.0, 200.0, 0.0, 150.0);
}
void lineSegment (screenPt p1, screenPt p2) {
        glBegin (GL_LINES);
        glVertex2i (p1.x, p1.y);
        glVertex2i (p2.x, p2.y);
        glEnd();
        glFlush();
}
```

Teacher's Signature _____

```
void drawCurve (int curveNum) {
    const double twoPi = 6.283185;
    const int a = 175, b = 60;
   float r, theta, dtheta = 1.0/ float (a);
    int x0 = 200, y0 = 250;
  screenPt curvePt[2];
   curve = curveNum;
   glColor3f (red, green, blue);
   curvePt[0].x = x0;
   curvePt[0].y = y0;
   glClear (GL_COLOR_BUFFER_BIT);
  switch (curveNum) {
   case limacon : curvePt[0].x += a + b; break;
   case cardiod : curvePt[0].x += a+b; break;
   case threeleaf : curvePt[0].x += a; break;
   case spiral : break;
   default : break;
}
 theta = dtheta;
 while (theta < dtheta) {
            switch (curveNum) {
              case limacon : r = a * cos(theta) + b; break;
              case cardiod : r = a*(1+cos(theta)); break;
            case threeleaf : r = a*cos(3*theta)); break;
            case spiral : r = (a/4.0)*theta; break;
            default: break; }
     curvePt[1].x = x0 + r * cos(theta);
     curvePt[2].y = y0 + r * sin(theta);
      lineSegment (curvePt[0], curvePt[1]);
       curvePt[0].x = curvePt[1].x;
       curvePt[0].y = curvePt[1].y;
        theta += dtheta;
} }
```

Date _____

Expt. No. _____ Page No. _____

```
void colorMenu(int id){
  switch(id){
      case 0: break;
      case 1: red=0;
              green=0;
              blue=1; break;
      case 2: red=0;
              green=1;
              blue=0; break;
      case 3: red=0;
              green=1;
              blue=1; break;
      case 4: red=0;
              green=0;
              blue=0; break;
      case 5: red=1;
              green=0;
              blue=1; break;
      case 6: red=1;
              green=1;
              blue=0; break;
      case 7: red=1;
              green=1;
              blue=1; break;
      default: break;
  }
  drawCurve(curve);
}
```

Teacher's Signature _____

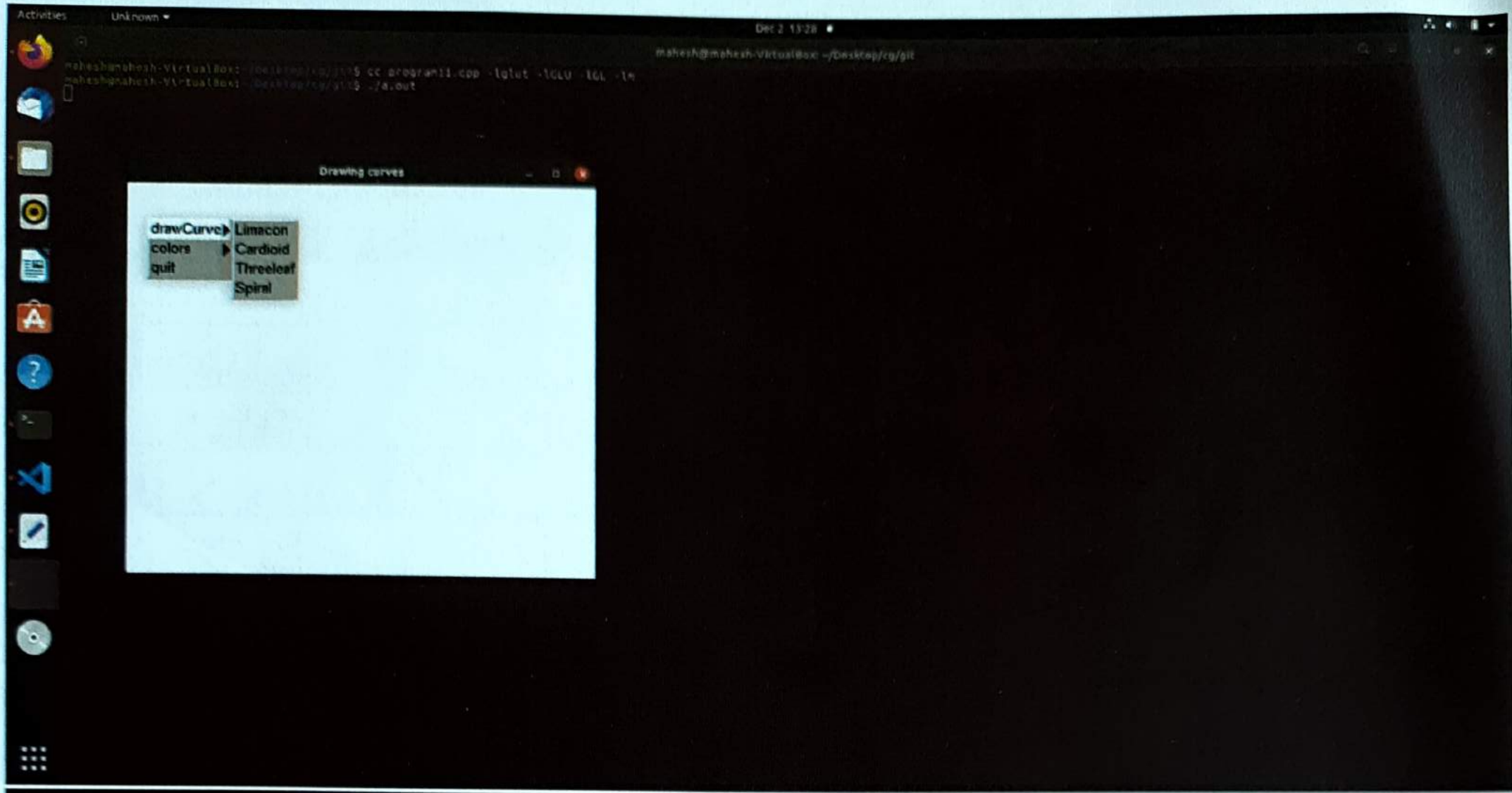
```
void main_menu(int id){
    switch(id){
            case 3: exit(0);
            default: break;
            }
  }
void mydisplay(){}
void myreshape(int nw, int nh){
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(0.0, (double)nw, 0.0, (double)nh);
        glClear(GL_COLOR_BUFFER_BIT);
    }

int main(int argc, char** argv){
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
        glutInitWindowSize(w,h);
        glutInitWindowPosition(100,100);
        glutCreateWindow("Drawing curves");
       int curveId = glutCreateMenu(drawCurve);
       glutAddMenuEntry("Limacon",1);
        glutAddMenuEntry("Cardiod",2);
        glutAddMenuEntry("Threeleaf",3);
        glutAddMenuEntry("Spiral",4);
        glutAttachMenu(GLUT_LEFT_BUTTON);
        int colorId= glutCreateMenu(colorMenu);
      glutAddMenuEntry("Red",4);
        glutAddMenuEntry("Green",2);
        glutAddMenuEntry("Blue",1);
        glutAddMenuEntry("Black",0);
```

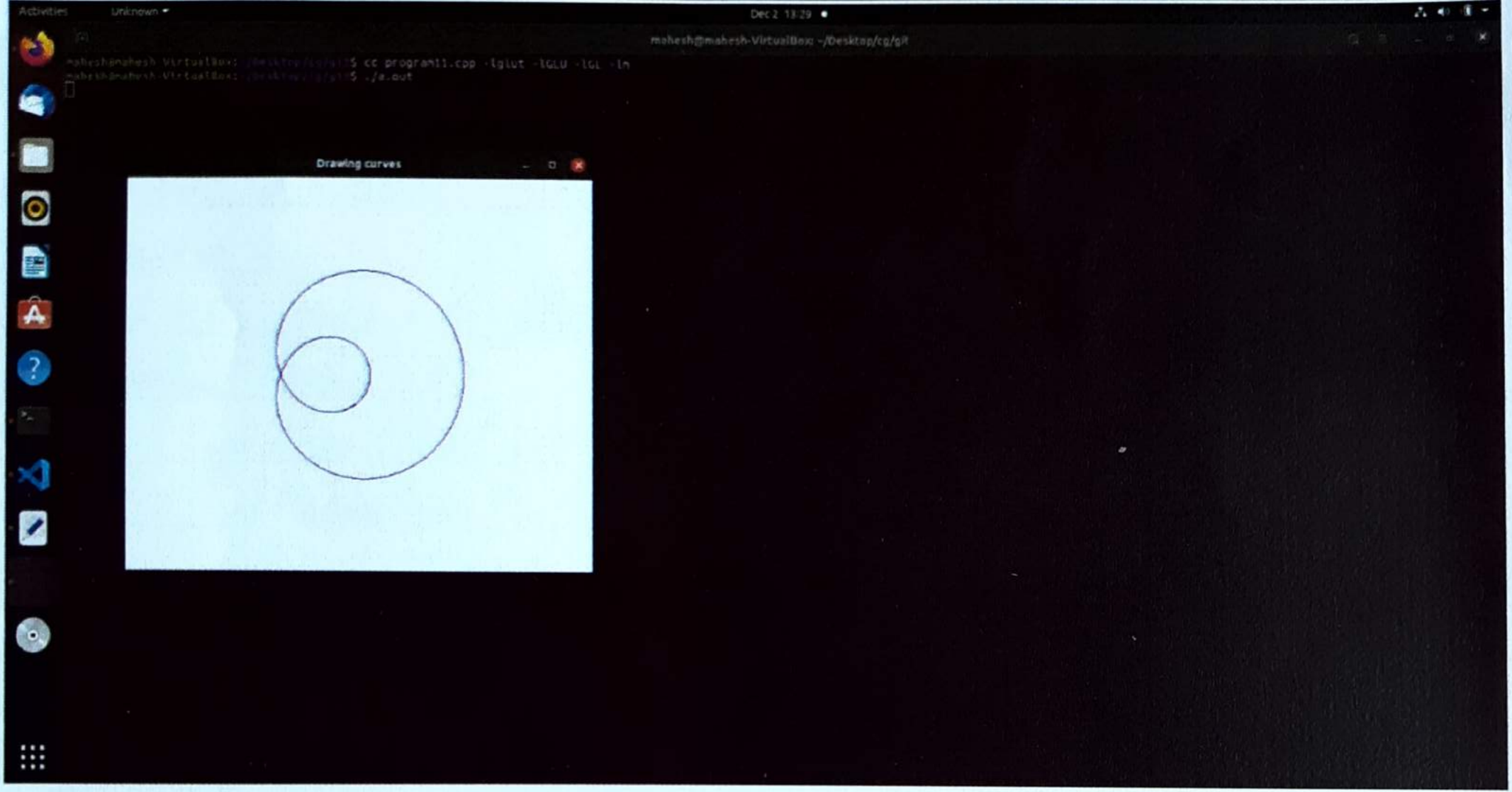
Date _____

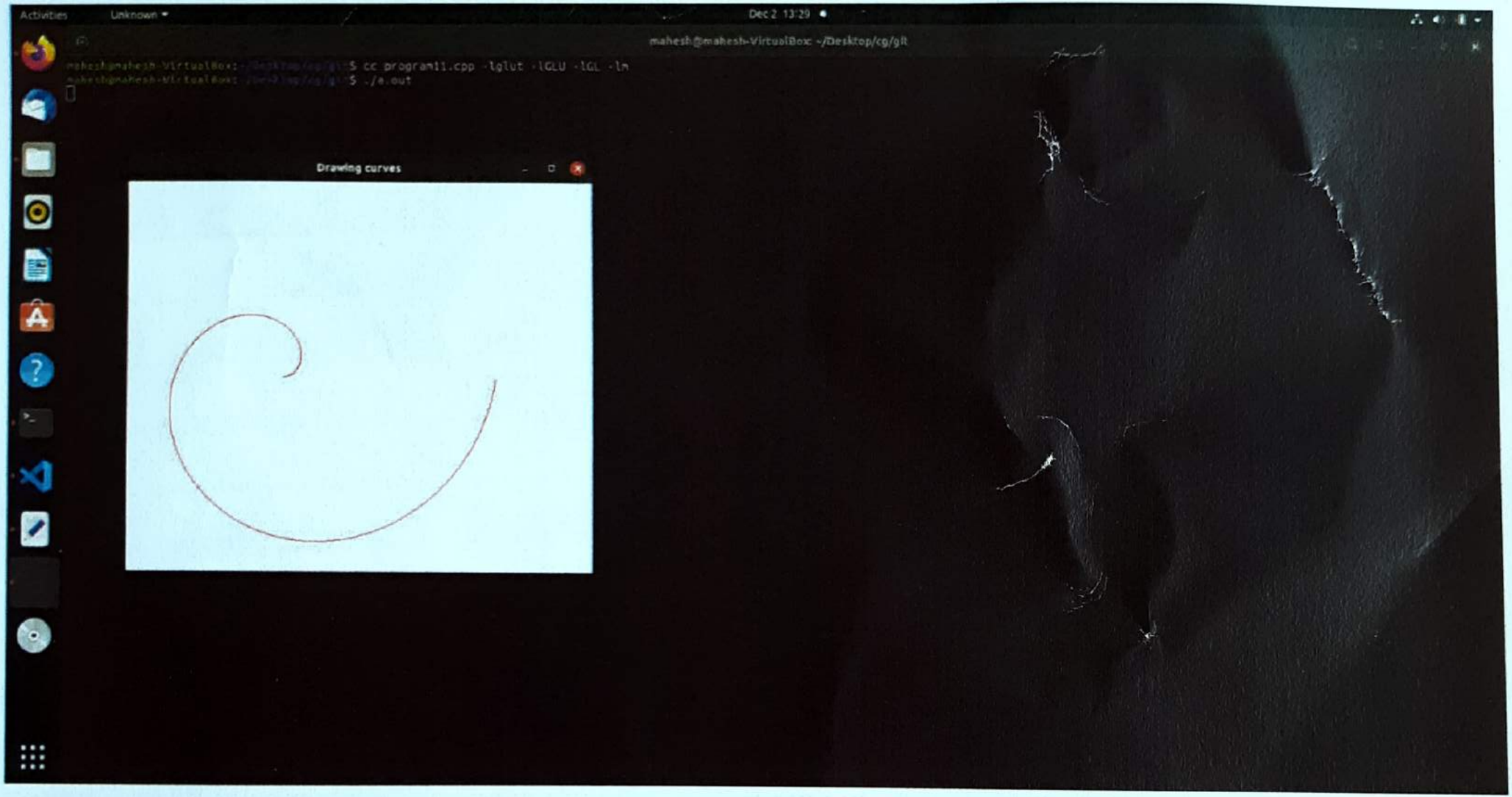
Expt. No. _____ Page No. _____

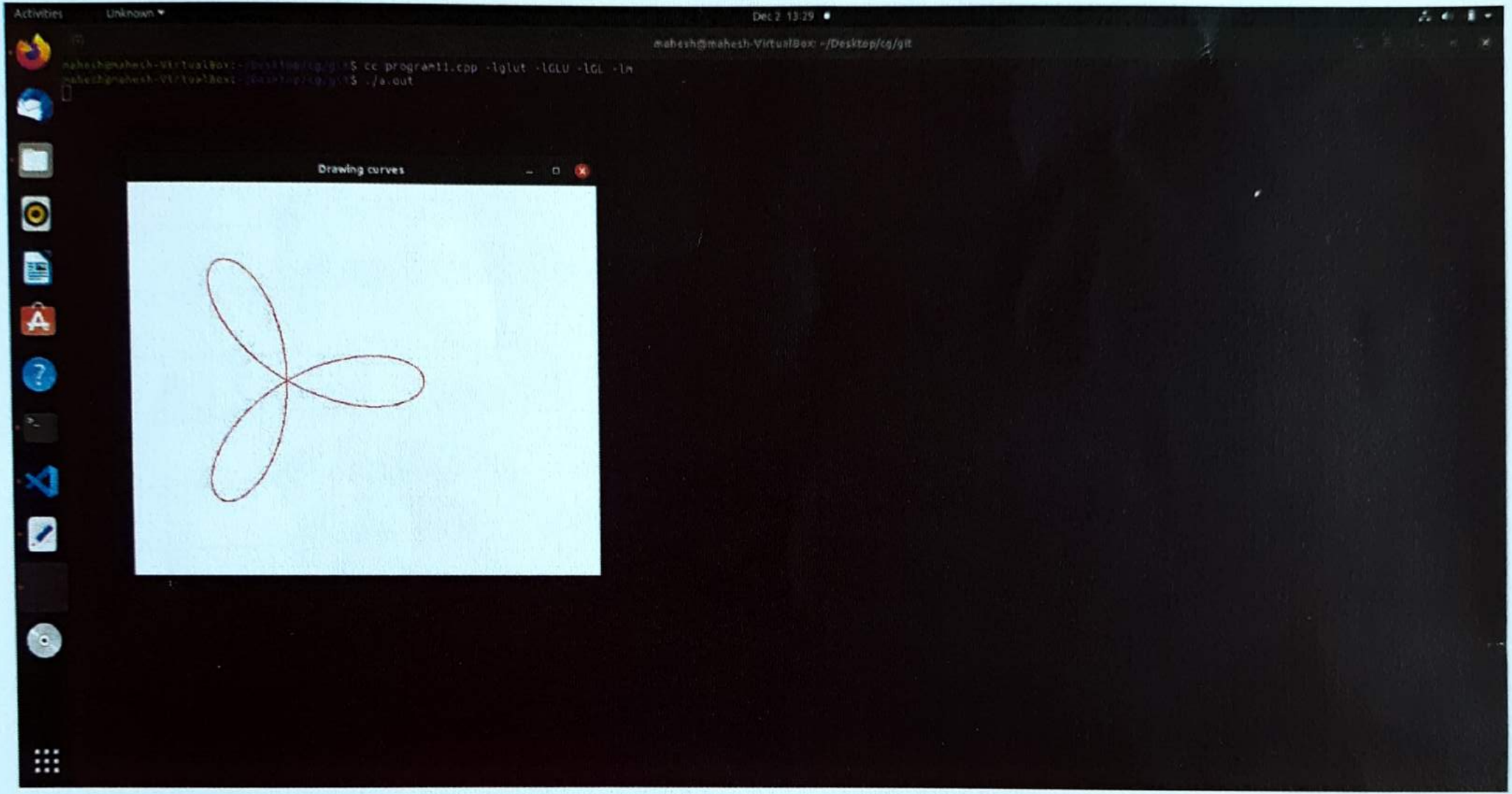
```
glutAddMenuEntry("Yellow", 6);
glutAddMenuEntry("Cyan", 3);
glutAddMenuEntry("Magenta", 5);
glutAddMenuEntry("White", 7);
glutAttachMenu(GLUT_LEFT_BUTTON);
glutCreateMenu(main_menu);
glutAddSubMenu("drawCurve", curveId);
glutAddSubMenu("colors", colorId);
glutAddMenuEntry("quit", 3);
glutAttachMenu(GLUT_LEFT_BUTTON);
myinit();
glutDisplayFunc(mydisplay);
glutReshapeFunc(myreshape);

glutMainLoop();
}
```

Teacher's Signature _____


Activities
Unknown
Dec 2 13:28
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program1.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
Drawing curves
drawCurve
colors
quit
Limacon
Cardioid
Threeleaf
Spiral


Activities
Unknown
Dec 2 13:29
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ cc program11.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
Drawing curves


Activities
Unknown
Dec 2 13:29
mahesh@mahesh-VirtualBox: ~/Desktop/cg/gl
mahesh@mahesh-VirtualBox:~/Desktop/cg/gl$ cc program11.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/gl$ ./a.out
Drawing curves


Activities
Unknown
Dec 2 13:29
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
$ cc program11.cpp -lglut -lGLU -lGL -lm
$ ./a.out
Drawing curves

Date _____

Expt. No. 12 Page No. _____

Write a program to construct Bezier curve. Control points are supplied through keyboard / mouse.

```
#include <iostream.h>
#include<math.h>
#include <stdio.h>
#include <stdlib.h>
#include <GL/glut.h>
using namespace std;
float f, g, r, x1[4], yc[4];
 int flag=0;
void myInit(){
            glClearColor(1,1,1,1);
            glColor3f(1,1,1);
            glPointSize(5);
            gluOrtho2D(0,500,0,500);
    }
void drawPixel(float x, float y){
            glBegin(GL_POINTS);
            glVertex2f(x,y);
            glEnd();
}
void display(){
            glClear(GL_COLOR_BUFFER_BIT);
            int i;
            double t;
            glColor3f(0,0,0);
            glBegin(GL_POINTS);
```

Teacher's Signature _____

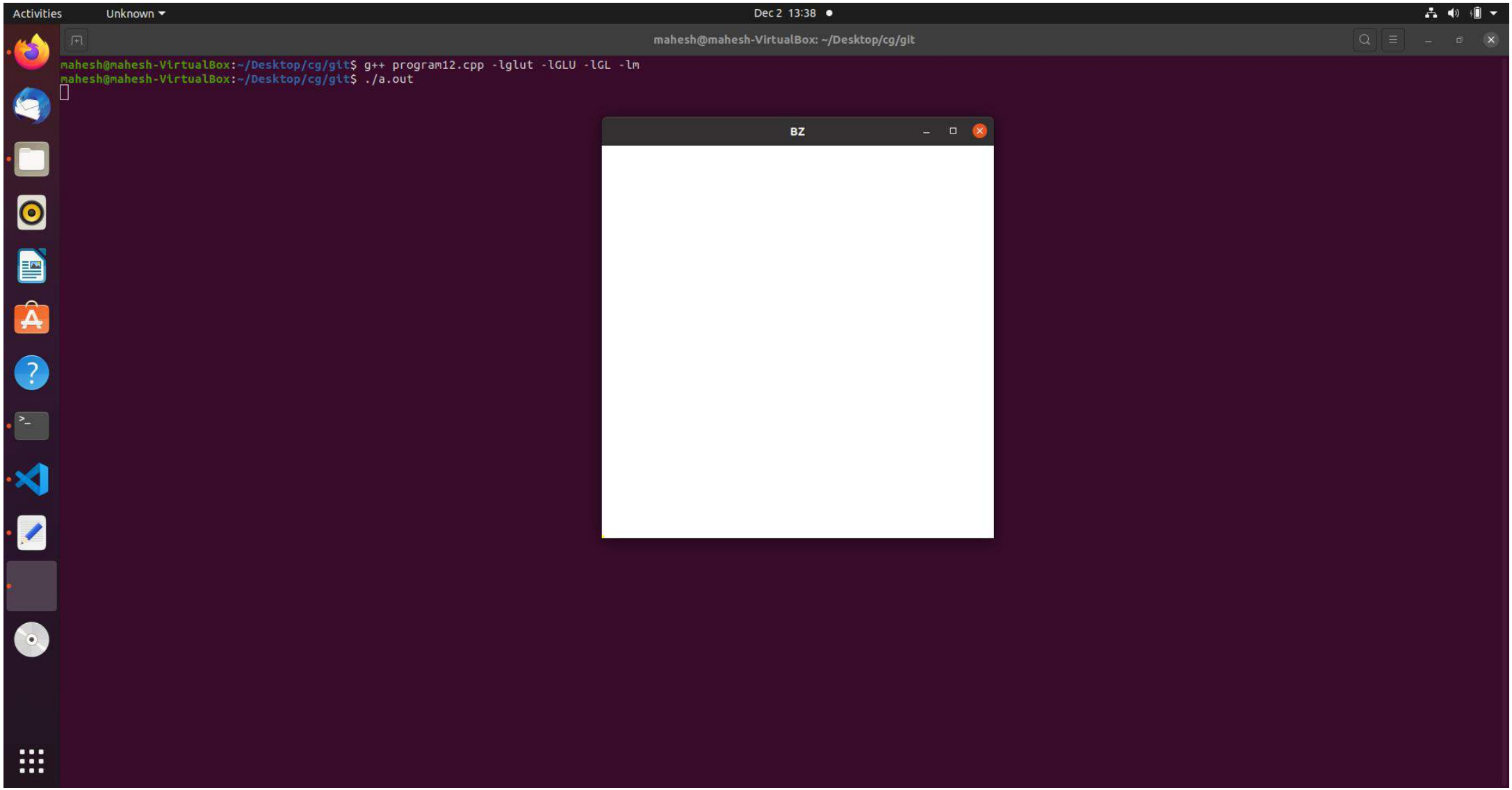
```
for(t=0; t<1; t=t+0.005){

double xt = pow(1-t,3)*x1[0] + 3*t*pow(1-t,2)*x1[1]
          + 3*pow(t,2)*(1-t)*x1[2] + pow(t,3)*x1[3];
double yt = pow(1-t,3)*yc[0] + 3*t*pow(1-t,2)*yc[1]+
          3*pow(t,2)*(1-t)*yc[2] + pow(t,3)*yc[3];
  glVertex2f(xt,yt);
}
 glColor3f(1,1,0);
   for(i=0; i<4; i++){
                    glVertex2f(x1[i], yc[i]);
                     glEnd();
                     glFlush();
                  }
 }
void mymouse(int btn, int state, int x, int y){

    if(btn==GLUT_LEFT_BUTTON && state==GLUT_DOWN && flag<4)
     {
         x1[flag] = x;
         yc[flag] = 500-y;
          cout << "X:" << x << "Y" << 500-y;
         glPointSize(3);
         glColor3f(1,1,0);
          glBegin(GL_POINTS);
          glVertex2f(x, 500-y);
           glEnd();
           glFlush();
            flag++;
     }
```

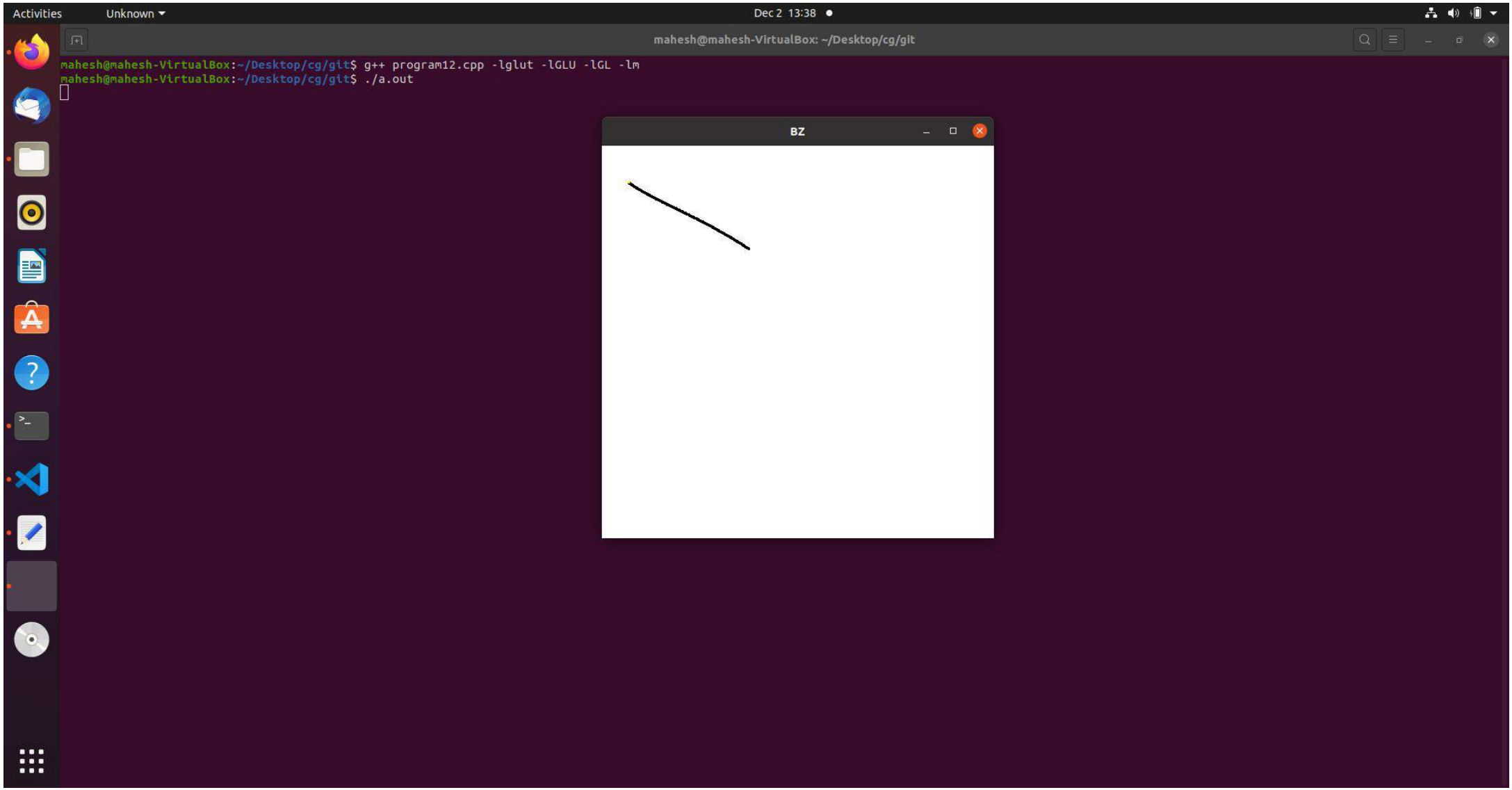
Date _____

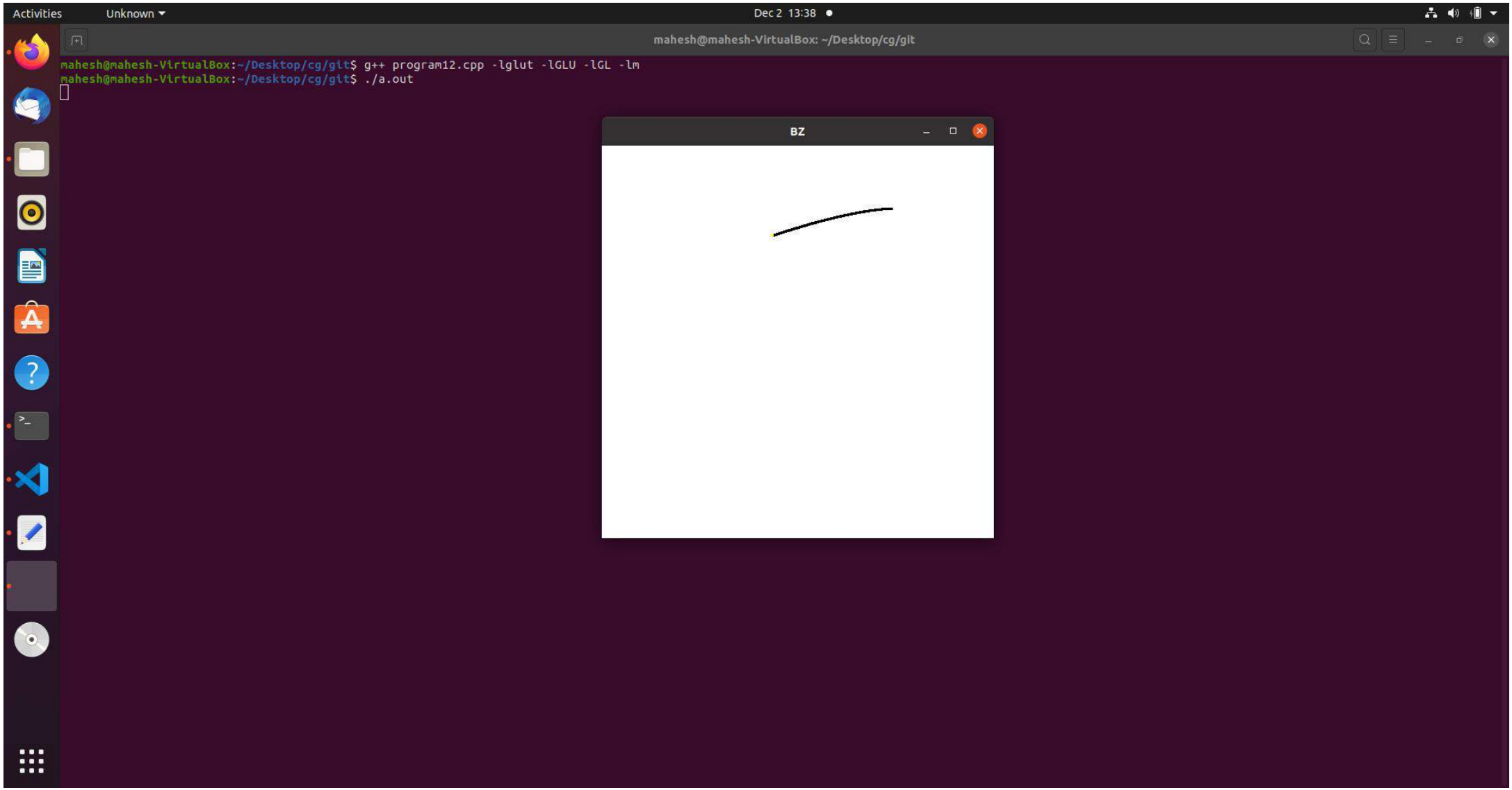
Expt. No. _____ Page No. _____

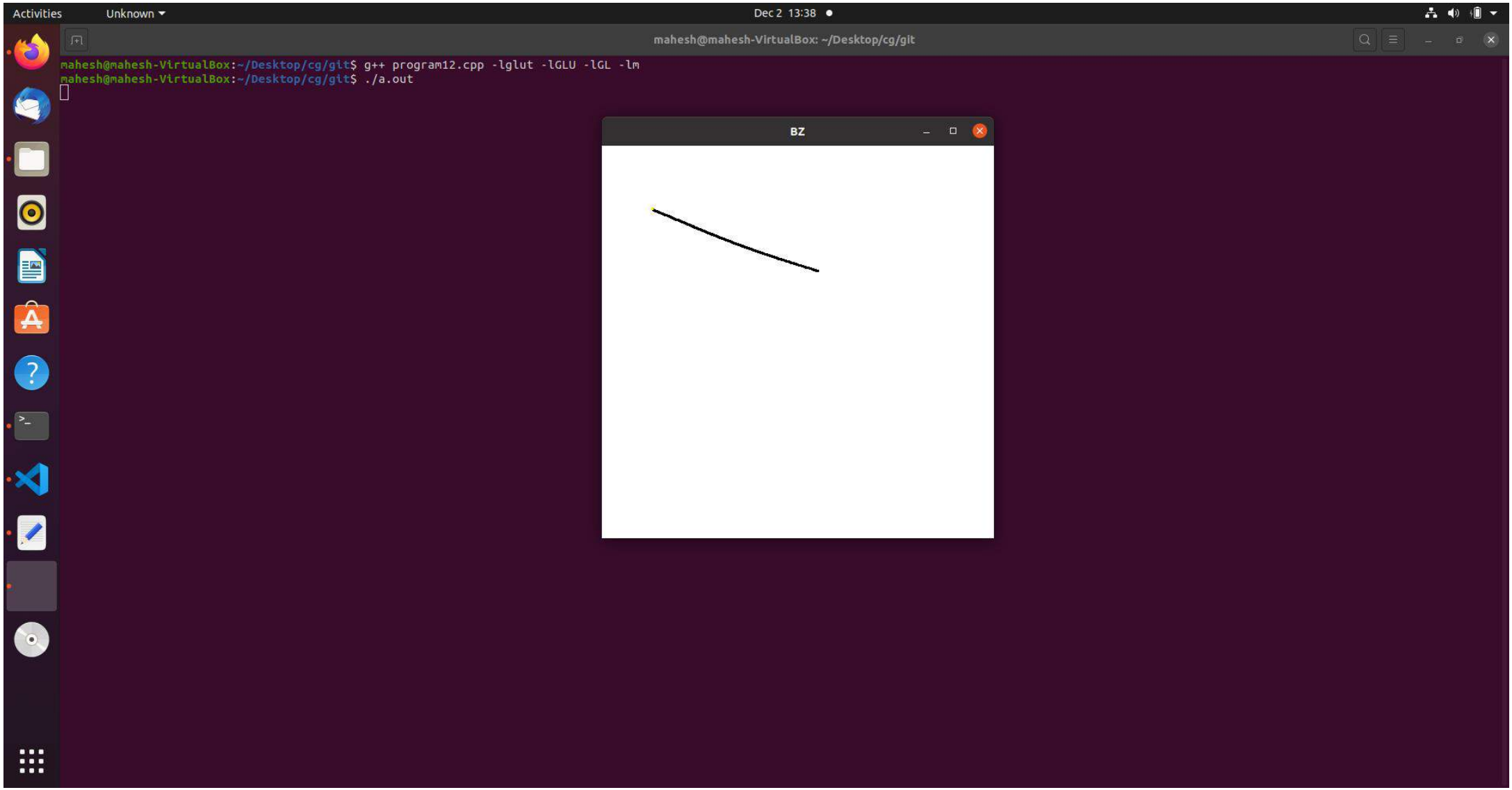
```
if( flag >=4 && btn == GLUT_LEFT_BUTTON)
{
        glColor3f (0,0,1);
        display();
        flag = 0;
  }
}
int main ( int argc, char* argv[] ) {
     glutInit (&argc, argv);
     glutInitDisplayMode (GLUT_SINGE | GLUT_RGB);
      glutInitWindowSize (500, 500);
      glutInitWindowPosition (0, 0);
      glutCreateWindow ("BZ");
      glutDisplayFunc ( display );
      glutMouseFunc ( mymouse );
      myInit ();
       glutMainLoop ();
  }
```

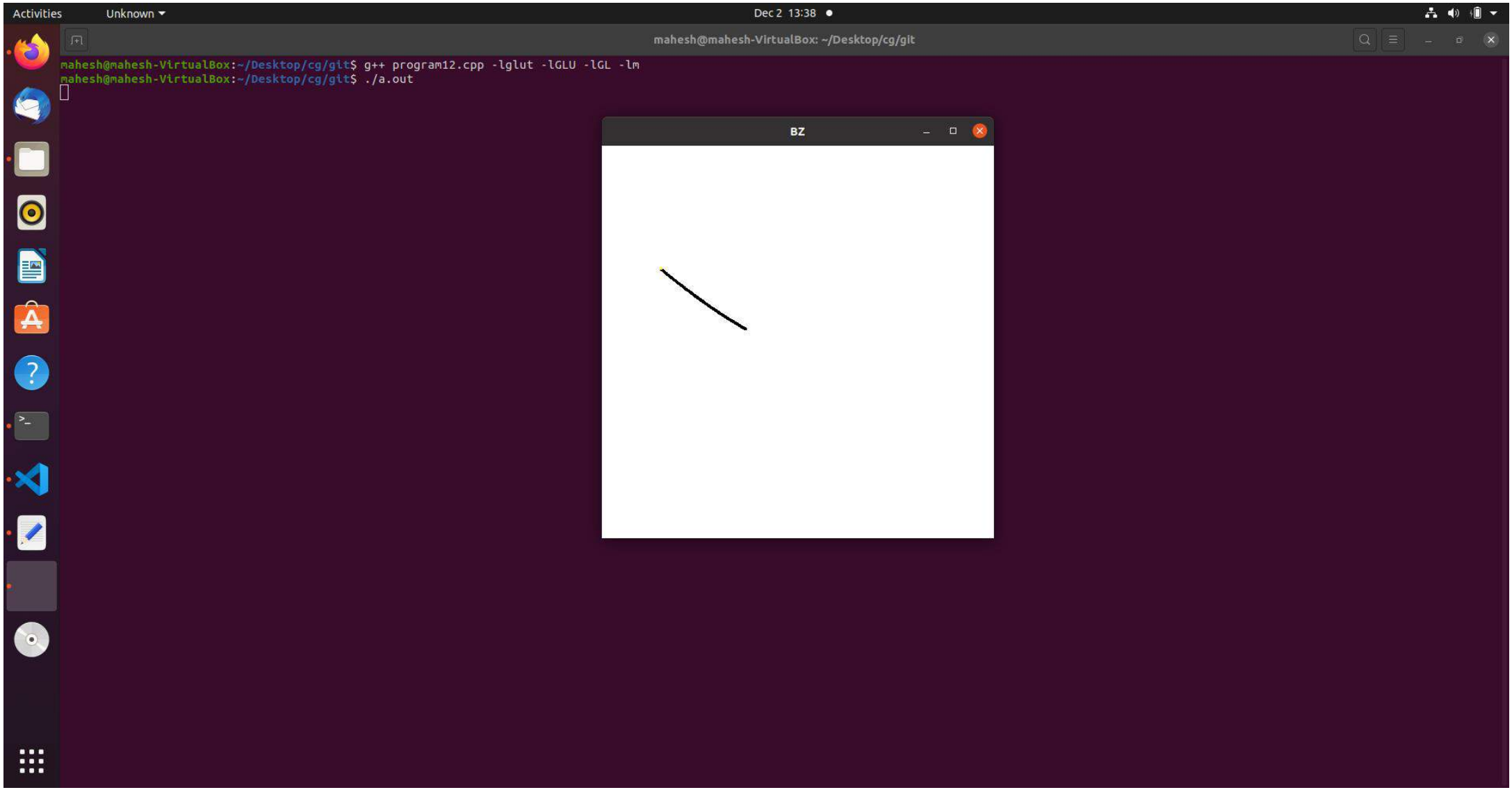
Teacher's Signature _____

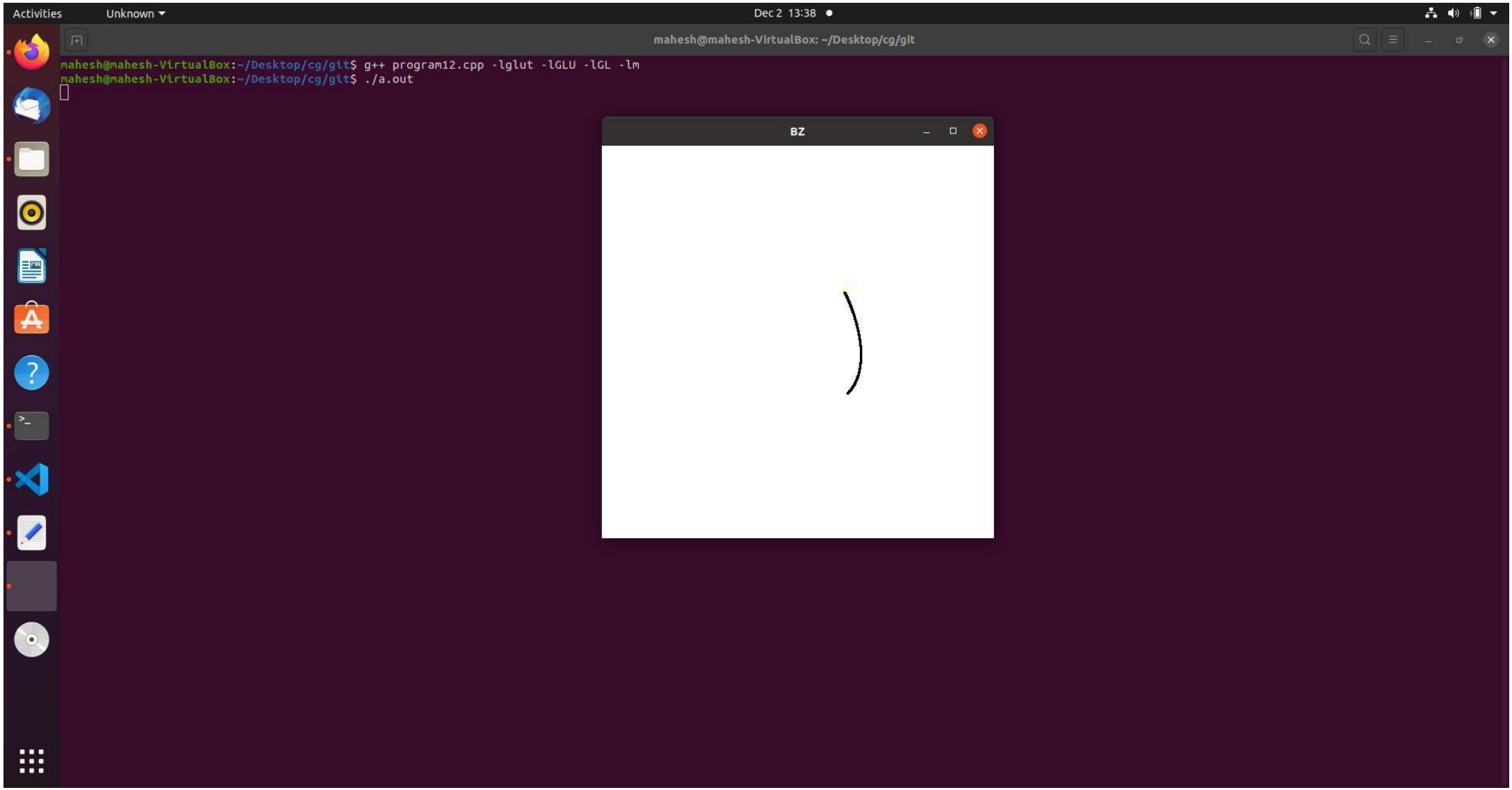

Activities
Unknown
Dec 2 13:38
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ g++ program12.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
BZ


Activities
Unknown
Dec 2 13:38
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ g++ program12.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
BZ


Activities
Unknown
Dec 2 13:38
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ g++ program12.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
BZ


Activities
Unknown
Dec 2 13:38
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ g++ program12.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
BZ


Activities
Unknown
Dec 2 13:38
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ g++ program12.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
BZ


Activities
Unknown
Dec 2 13:38
mahesh@mahesh-VirtualBox: ~/Desktop/cg/git
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ g++ program12.cpp -lglut -lGLU -lGL -lm
mahesh@mahesh-VirtualBox:~/Desktop/cg/git$ ./a.out
BZ