Date .....

Expt. No. ..... Page No. .....

PROGRAM 1

1. Write a Program to generate a line using bresenham's line drawing technique. Consider Slopes greater than one and Slopes less than one. User can Specify inputs through Keyboard/mouse.

```
#define BLACK 0
#include <Stdio.h>
#include <GL/glut.h>

/GL int Point [4] = {0};
int x1, x2, y1, y2;
void draw_ Pixel (int x, int y, int value)
{
        gl Begin (GL_ POINTS);
        gl Vertex 2i (x,y);
        gl End ();
}
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;
        incx = 1;
```

Teacher's Signature : _____

```
if (y2<y1) incy=-1;
x=x1; y=y1;
if (dx>dy)
{
        draw_Pixel(x,y,BLACK);
        e=2*dy-dx;
        inc1=2*(dy-dx);
        inc2=2*dy;
        for(i=0;i<dx;i++)
        {
                if(e>=0)
                {
                        y+=incy;
                        e+=inc1;
                }
                else e+=inc=2;
                x+=incx;
                draw_Pixel(x,y,BLACK);
        }
}
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)
```

Date

Expt. No. Page No.

```
            {
                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();
}
void myinit()
{
    glClearColor(1.0, 1.0, 1.0, 1.0)
    glColor3f(1.0, 0.0, 0.0);
    glPointSize(1.0);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();
    gluOrtho2D(0.0, 499.0, 0.0, 499.0);
}
void main(int argc, char** argv)
```

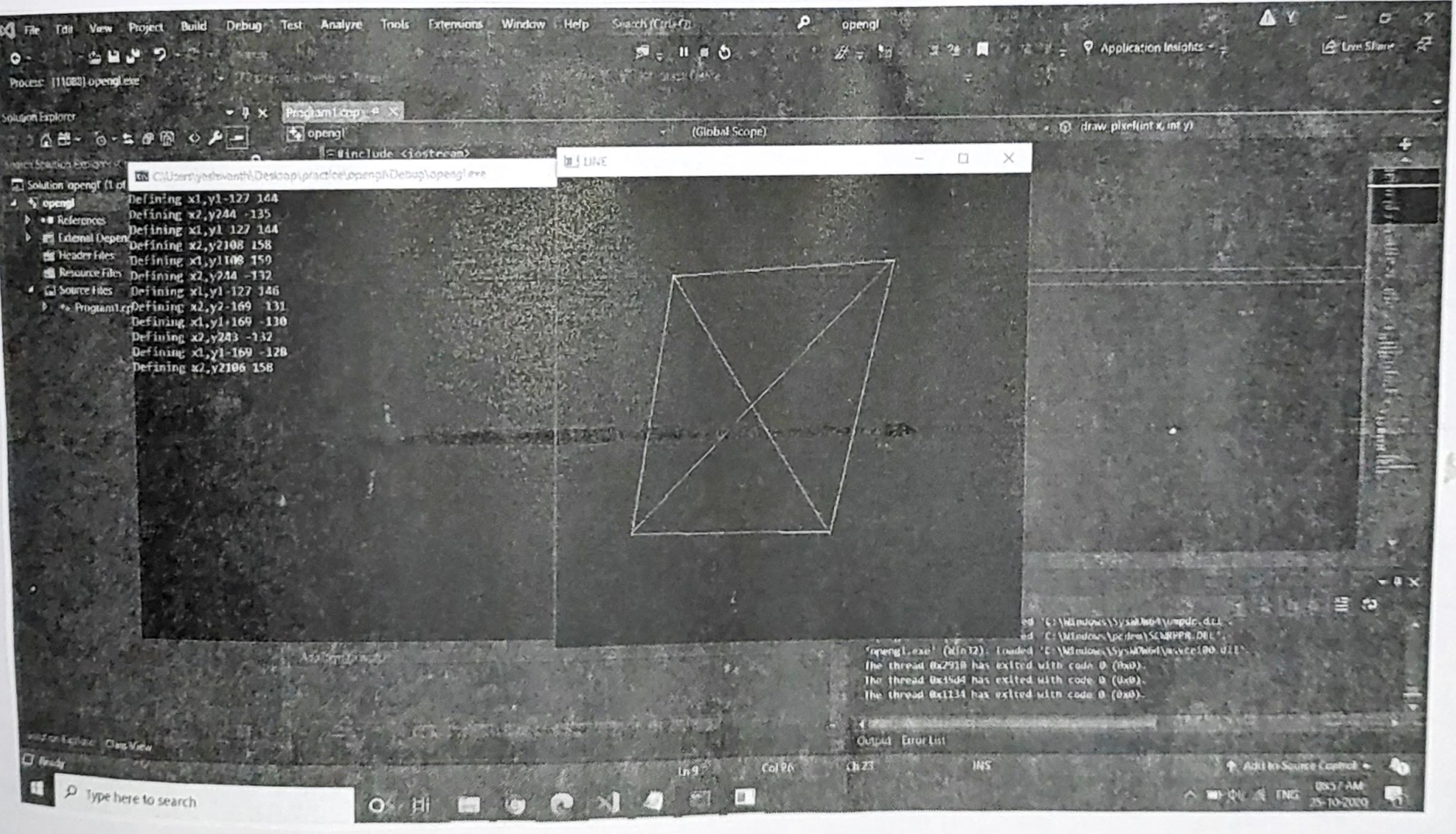
Teacher's Signature :

Date

Expt. No. Page No.

```
{
// int x1, x2, y1, y2;

Printf "Enter points : x1, y1, x2, y2")
Scanf("%d %d %d", &x1, &x2 &y, &2);
/*  point[0] = x1;
point[1] = y1;
point[2] = x2;
point[3] = y2;
glut init (&argc, argv);
glut init Display Mode (GLUT_SINGLE|GLUT_RGB);
glut Window Size (500,500);
glut nit Window Pasition (0,0,);
glut Create Window ("Bresenham.s Algorithm");
glut Display Func (display);
my init ();
glut Main Loop ();

}
```

Teacher's Signature :


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
opengl
Application Insights
Process: [11088] opengl.exe
Solution Explorer
Program1.cpp
opengl
(Global Scope)
draw pixel(int x, int y)
#include <iostream>
C:\Users\yashwanth\Desktop\practice\opengl\Debug\opengl.exe
Defining x1,y1-127 144
Defining x2,y244 -135
Defining x1,y1 127 144
Defining x2,y2108 158
Defining x1,y1108 159
Defining x2,y244 -132
Defining x1,y1-127 146
Defining x2,y2-169 -131
Defining x1,y1+169 -130
Defining x2,y243 -132
Defining x1,y1-169 -128
Defining x2,y2106 158
LINE
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\msvcr100.dll'
The thread 0x2918 has exited with code 0 (0x0).
The thread 0x14d4 has exited with code 0 (0x0).
The thread 0x1134 has exited with code 0 (0x0).
Output Error List
Ready
Ln 9 Col 26 Ch 23 INS
Add to Source Control
Type here to search
ENG
25-10-2019

Date

Expt. No. Page No.

PROGRAM 2

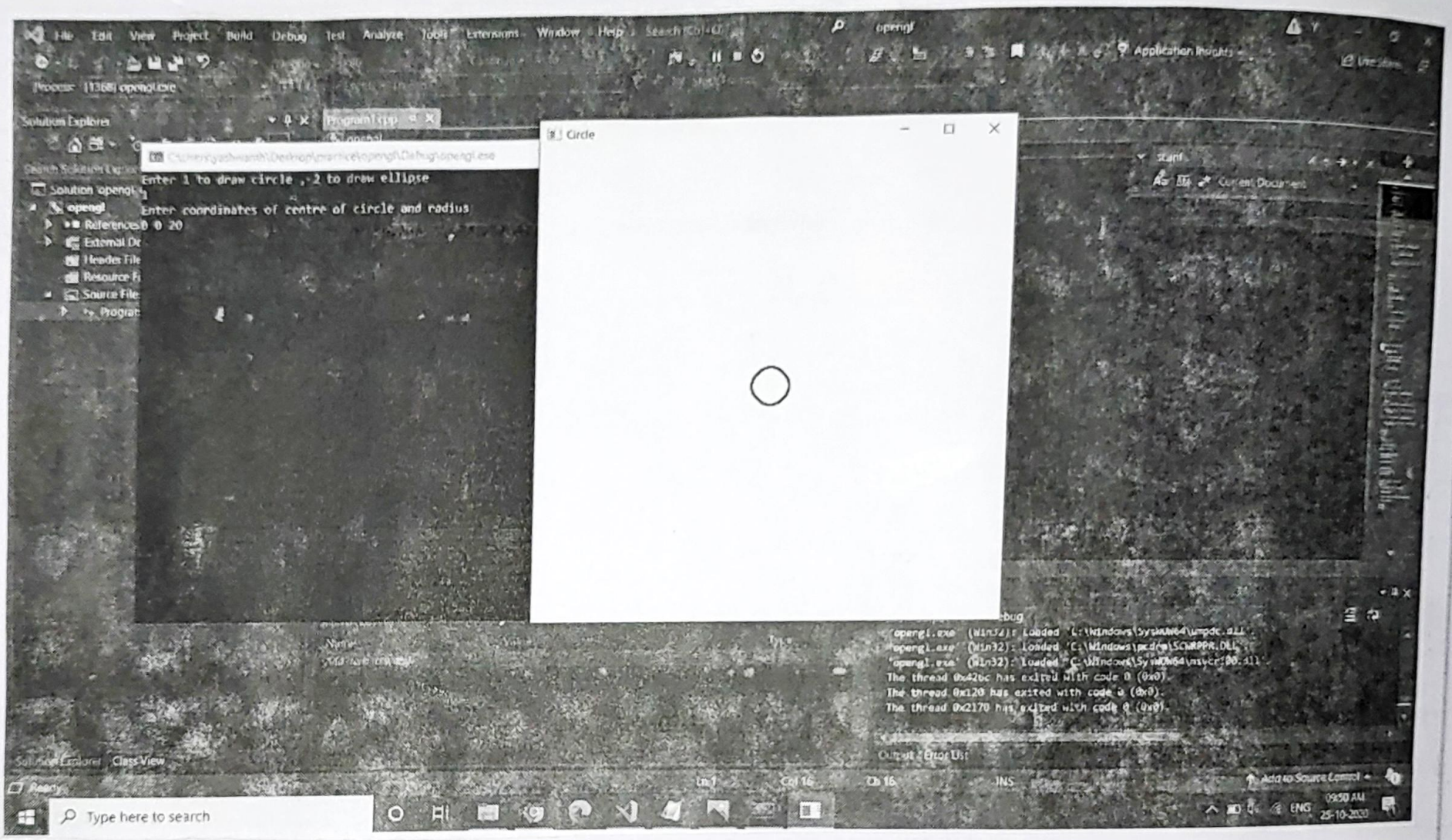
2. Write a Program to generate a circle Using Bresenham's Circle drawing technique. User can Specify inputs though Keyboard/mouse.

```
#include <stdio.h>
#include <gl/glut.h>
int xc, yc, r;
void drawCircle (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 (int xc, int yc, int r)
{
        int x=0, y=r;
        int d=3-2* r;
        While (x<=y)
```

Teacher's Signature :

```
{
        drawcircle(xc,yc,x,y);
        x++;
        if(d<0)
                d=d+4* x 6;
        else
        {
                y--;
                d=d+4*(x-y)+10;
        }
        drawcircle(xc,yc,x,y);
}
}
void display()
{
        int j;
        glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_
        BUFFER_BIT);
        CircleBres(xc,yc,r);
        glFlush();
}
void myinit()
{
        glClearColor(1.0,1.0,1.0,0.0);
        glColor3f(0.0,0.0,1.0);
```

```
    gl points Size (5.0);
    glu Ortho 2D (0.0, 500, 0.0, 500);
}

void main (int argc, Char * argv [])

{
    int j;
    print f(" Enter Coord of Centre of Circle & radius");
    Scanf("%d %d %d, & xc, & yc & r);
    glutInit ( & argc, argv);
    glutinit Display Mode (GLUT_SINGLE|GLUT_RGB
    GLUT_DEPTH);
    glutInit Window Size (500, 500);
    glut Init Window position (100, 100);
    glut Create Window ("Bresenhams circle");
    glut Display Func (display);
     myinit ();
     glut Main Loop ();

}
```


Circle
Enter 1 to draw circle , 2 to draw ellipse
1
Enter coordinates of centre of circle and radius
0 0 20
Type here to search

Date

Expt. No. Page No.

PROGRAM 3

3. Write a Program to Create a Cylinder and parallele - piped by extruding a circle and quadrilateral respectively. Allow the User to Specify the circle and the quadrilateral through Keyboard/mouse.

```
# include <GL/gl.h>

void draw_pixel (GLint cx, GLint cy)

{

        glColor3f (1.0,0.0,0.0);
        glBegin (GL_POINTS);
              glVertex2i (cx, cy);
        glEnd ();

}

Void plotpixels (GL int h, GLint k, GLint x,
GLint y)

{

        draw_Pixel (x+h, y+k);
        draw_Pixel (-x+h, y+k);
        draw_pixel (x+h, -y+k);
        draw_Pixel (-x+h, -y+k);
```

Teacher's Signature :

```
    drawpixel(y+h, x+k);
    drawpixel(-y+h, x+k);
    drawpixel(y+h, -x+k);
    drawpixel(-y+h, -x+k);

}

void circle_draw(GLint h, GLint k, GLint r)
// Midpoint Circle Drawing Algorithm

    GLint d = 1-r, x=0, y=r;
    while (y>x)

    {

        plotpixels(h, k, x, y);
        if (d<0)
        d+=2* x+3;
        else

        {
                d+=2*(x-y)+5;
                --y;

        }
            ++x;

}
plotpixels(h, k, x, y);
```

```
}

void Cylinder_draw()

{

        GLint xc=100, yc=100, r=50, i, n=5;

for(i=0; <n; i+=3)
        Circle_draw(xc, yc+i, r);

}
void Parallelepiped(int x1, int x2, int y1, int y2)

{
        glColor3f(0.0, 0.0, 1.0);
        glBegen(GL_LINE_LOOP);
            glvertex2i(x1, y1);
            glvertex2i(x2, y1);
            glvertex2i(x2, y2);
            glvertex2i(x1, y2);

    glEnd();

}

void Parallelepiped_draw()

{
```

Date

Expt. No. Page No.

```
void init (void)
{
        gl clear Color (1.0, 1.0, 0.0);
        gl Matrix Mode (GL_PROJECTION);
        glu Ortho 2D (0.0, 400, 0.0, 300.0);
}
Void display (void)
{
        gl clear (GL_COLOR_BUFFER_BIT);
        gl Color3f (1.0, 0.0, 0.0);
        Cylinder_draw ();
        Parallelepiped_draw ();
        gl Flush ();
}
Void main (int argc, Char ** argv)
{
        glut init ( & argc, argv );
        glutinit Display Mode (GLUT_SINGLE |
        GLUT_RGB);
        glutInit Window Size (400, 300);
        ghut Create Window ("Cylinders, Parallele-
        piped Disp by Extruding circle &
        Quadrilateral");
```

Teacher's Signature :

Date

Expt. No. Page No.

```
    init ();
    glutDisplayFunc(display);
    glutMainLoop();

}
```

Teacher's Signature :

Date

Expt. No. Page No.

PROGRAM 4

4. Write a Program to recursively Subdivides a tetrahedron to from 3D Sierpinski gasket. The number of recursive Steps is to be Specified at execution time.

```
#include <GL/glut.h>

/* initial triangle */
typedef GLfloat Point[3];
Point v[I] = {{ 30.0,50.0,100.0},{0.0,450.0,150.0},
    {-350.0,-400.0,-150.0},{350.,-400.,-150.0}};

int n; /* number of recursive Steps */

void triangle (Point a, Point b, Point c)

/* display one triangle */

{

   glBegin(GL_TRIANGLES);
     glvertex3fv (a);
     glvertex3fv (b);
     glvertex3fv (c);
          glEnd ();

}
```

Teacher's Signature :

```
void divide_triangle (Point a, Point b, Point c, int m)
{
/* triangle Subdivision Using Vertex numbers */
    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);
        divid_triangle (b, v2, v0, m-1);

    }
    else (triangle (a, b, c));

/* draw triangle at end of recursion */

}
void tetra (int m)
{
gl Color 3f (1.0, 0.0, 0.0);
```

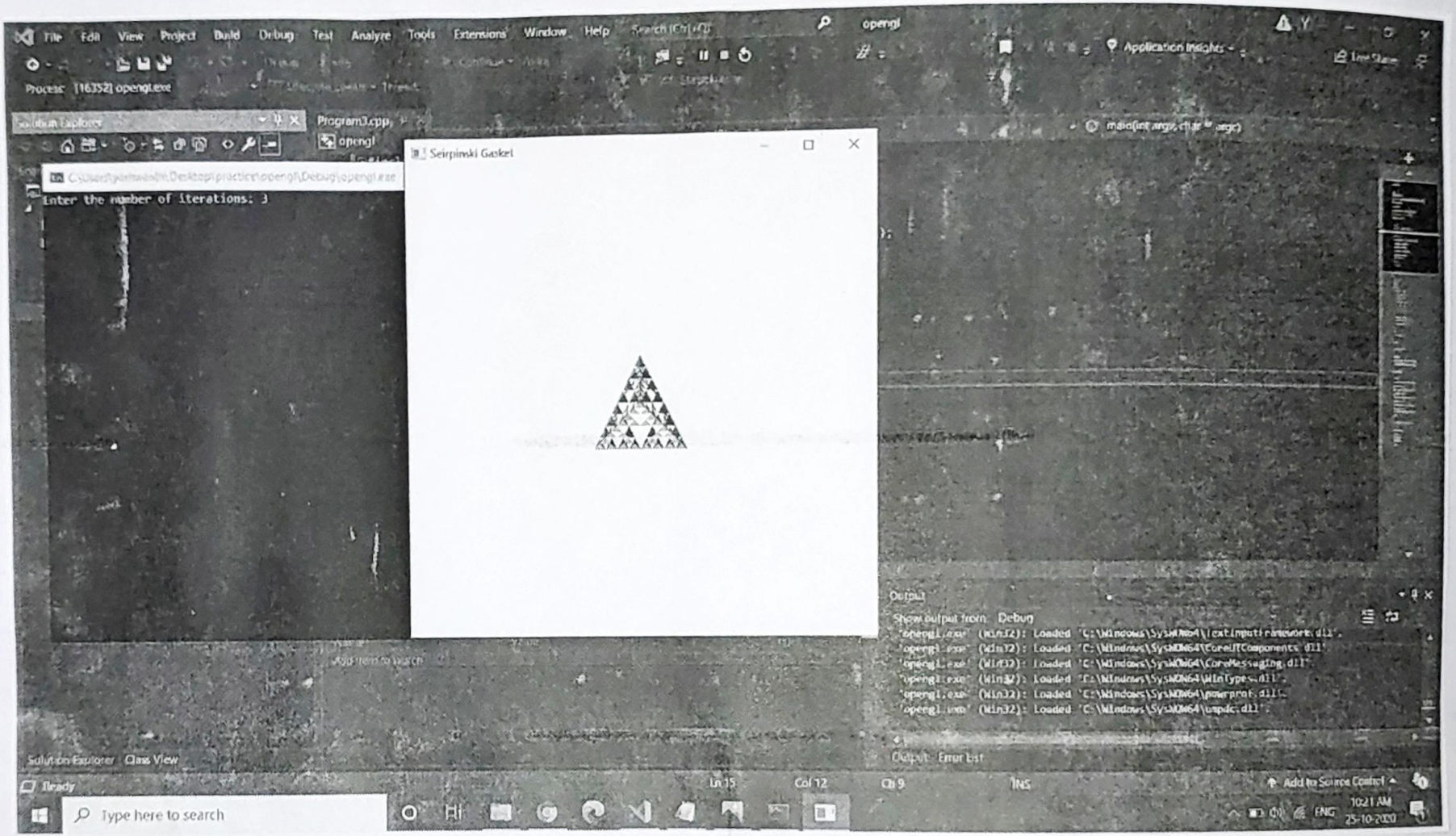
```
divide_triangle (v[0], v[1], v[2], m);
glColor3f (0.0, 1.0, 0.0);
divide_triangle(v[3], v[2], v[1], m);
glColor3f (0.0, 0.0, 1.0);
divide_triangle (v[0], v[3], v[1], m);
glColor3f (0.0, 0.0, 0.0,);
divide_triangle(v[0], v[2], v[3], m);

}
void display ()

{
        gl clear Color (1.0, 1.0, 1.0, 1.0);
  gl Clear (GL_COLOR_BUFFER_BIT);
   tetra(n);
  gl Flush ();

}
void myinit ()

{

 gl matrix Mode (GL_PROJECTION);
 gl Load Identity ();
 gl Ortho (-499.0, 499.0, -499.0, 499.0, -499.0, 499.0);
 gl Matrix Mode (GL_MODELVIEW);

 }
```

Date

Expt. No. Page No.

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

    n=5;
    glutInit ( &argc, argv);
    glutInitDisplayMode (GLUT_SINGLE|GLUT_
    RGB);
    glutInitWindowSize(500,500);
    glutCreateWindow("3D Gasket");
    glutDisplayFunc(display);
        myinit();
    glutMainLoop();

}
```

Teacher's Signature :


Seirpinski Gasket
Enter the number of iterations: 3

Date

Expt. No. Page No.

PROGRAM 5

5. Write a program to create a house like figure and rotate it about a given fixed point using open GL/ CUDA transformation function.

```
#define BLACK 0
#include <stdio.h>
#include <math.h>
#include <GL/glut.h>

GLfloat house[3][9]={{100.0,100.1,175.0,250.0,
250.0,150.0,150.0,200.0,200.0},
                    {100.0,300.0,400.0,300.0,100.0,100.0,
                    150.0,150.0,100.0},
                    {1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0,1.0}
                   };

GLfloat arbitrary_x=100.0;
GLfloat arbitrary_y100.0;
GLfloat rotation_angle;

void drawhouse()

{

glColor3f(0.0,0.0,1.0);
```

Teacher's Signature :

Date

Expt. No. Page No.

```
glBegin(GL_LINE_LOOP);
  glVertex*2f(house[0][0], house[1][0]);
  glVertex2f house[0][1], house[1][1]);
  glVertex 2f house[0][3], house[1][3]);
  glVertex 2f house[0][4], house[1][4]);
  glEnd();

glColor3f(1.0, 0.0, 0.0)

    glBegin(GL_LINE_LOOP);
    glVertex2f(house[0][5], house[1][5]);
    glvertex 2f(house[0][6], house[1][6]);
    glvertex 2f(house[0][7], house[1][7]);
    glvertex 2f(house[0][8], horese[1][8]);
    glEnd();

  glcolor3f(0.0, 0.0, 1.0,);

      glBegin(GL_LINE_LOOP);
      glvertex2f(house[0][1], house[1][1]);
      glVertex2f(house[0][2], house[1][2]);
      glvertex2f(house[0][3], house[1][3]);
      glEND();

}
void display()

{
```

Teacher's Signature :

```
gl Clear (GL_COLOR_BUFFER_BIT);
gl MatrixMode (GL_MODELIEW);
glLoadIdentity ();
draw house ();

gl Translatef (arbitrary_x, arbitrary_y, 0.0);
gl Rotatef (rotation_angle, 0.0, 0.0, 1.0);
gl Translatef (-(arbitrary_x), -(arbitrary_y) 0.0);

drawhouse ();

glFlush ();

}

void myinit ()

{

    gl clear Color (1.0, 1.0, 1.0, 1.0 );
    gl Color 3f (1.0, 0.0, 0.0);
    gl PointSize (1.0);
    gl MatrixMode (GL_PROJECTION);
    glLoadIdentity ();
    gluOrtho 2D (0.0, 499.0, 0.0, 499.0);

}

void main (int argc, Char ** argv)
```

Teacher's Signature : _____

Date

Expt. No. Page No.

```
{

    Printf ("Enter the rotation angle \n");
    Scanf ("%f", &rotation_angle);
    glutInit (&argc, argv);
    glutInitDisplayMode (GLUT_SINGLE | GLUT_
                                        RGB);
    glutInitWindowSize (500, 500);
    glutInitWindowPosition (0,0);
    glutCreateWindow ("house rotation");
    glutDisplayFunc (display);
    myinit ();
    glutMainLoop ();

}
```

Teacher's Signature :

Date

Expt. No. Page No.

PROGRAM 5

5. Write a Program to implement the Cohen-Sutherland line clipping algorithm. Provision to Specify the input for mulliple lines, window for Clipping and viewport for displaying the Clipped image

```
// Cohen-Suderland Line Clipping Algorithm with window to viewPort Mapping

#include <stdio.h>
#include <GL/glut.h>

#define outcode int
double xmin=50, ymin=50, xmax=100, ymax=100; // Window boundaries.
double xvmin=200, yvmin=200, xvmax=300, yvmax=300; // Viewport boundaries
  //bit codes for the right, left, top, & bottom
const int RIGHT=8;
Const int LEFT=2;
Const int TOP=4;
Const int BOTTOM=1;

//Used to compute bit codes of a Point
outCode Compute out Code (double x, double y);

// Cohen-Sutherland clipping algorithm clips
```

Teacher's Signature :

Date

Expt. No. Page No.

a line from
// P0 = (x0, y0) to P1 = (x1, y1) against a rectangle with
// diagonal from (xmin, ymin) to (xmax, ymax).
void CohenSutherlandLineClipAndDraw
(double y0, double x1, double y1)

{

// outcodes for P0, P1, and whatever point lies outside the clip rectangle
outCode outcode0, outcode1, outcodeOut;
bool accept = false, done = false;

// Compute outcodes
outcode0 = computeOutCode(x0, y0);
outcode1 = ComputeOutcode(x1, y1);

do {

if(!(outcode0 | outcode1))

{
accept = true;
done = true;

}
else if (outcode0 & outcode1)
// logical and is not 0, Trivially reject

Teacher's Signature :

```
and exit
        done = true;

else

{

    // failed both tests, So Calculate the
    line segment to clip.
    // from an outside point to an intersection
    with clip edge
    double x, y;

    // Now find the intersection point;
    // use formuls y = y0 + slope * (x - x0),
    x = x0 + (1/slope) * (y - y0)
    if(outcodeout & TOP)
                        // point is above the
                        clip rectangle

    {
                x = x0 + (x1 - x0) * (ymax - y0)/
                                    (y1 - y0);
                y = ymax;

    }
    else if (outcodeOut & BOTTOM) // point is
    below the clip rectangle

    {
```

```
        X = X0 + (X1 - X0) * (Ymin - Y0) / (Y1 - Y0);
        Y = Ymin;

    }
    else if (outCode Out & Right) //Point is to the
    right of clip rectangle

    Y = Y0 + (Y1 - Y0) * (xmax - X0) / (X1 - X0);
    x = xmax;

    }
    else

    {
            Y = Y0 + (Y1 - Y0) * (xmin - X0) / (X1 - X0);
            X = Xmin;

    }

    //Now we move outside point to intersection
    point to clip
    // and get ready for next Pass.
    if (outCode Out == outCode 0)

    {
            X0 = X;
            Y0 = Y;
            outCode 0 = Compute OutCode (X0, Y0);

    }
```

```
else
{
        x1 = x;
        y1 = y;
        outcode1 = ComputeOutCode(x1, y1);
    }
  }
}while (!done);
if (accept)
{
        // Window to viewport mappings
    double sx = (xvmax - xvmin)/(xmax - xmin);
    // Scale parameters
    double sy = (yvmax - yvmin)/(ymax - ymin);
    double vx0 = xvmin + (x0 - xmin) * sx;
    double vy0 = yvmin + (y0 - ymin) * sy;
    double vx1 = xvmin + (x1 - xmin) * sy;
    double vy1 = xvmin + (x1 - xmin) * sx;
    double vy1 = yvmin + (y1 - ymin) * sy;

        // draw a red colored viewport
    glColor3f(1.0, 0.0, 0.0);
    glBegin(GL_LINE_LOOP);
```

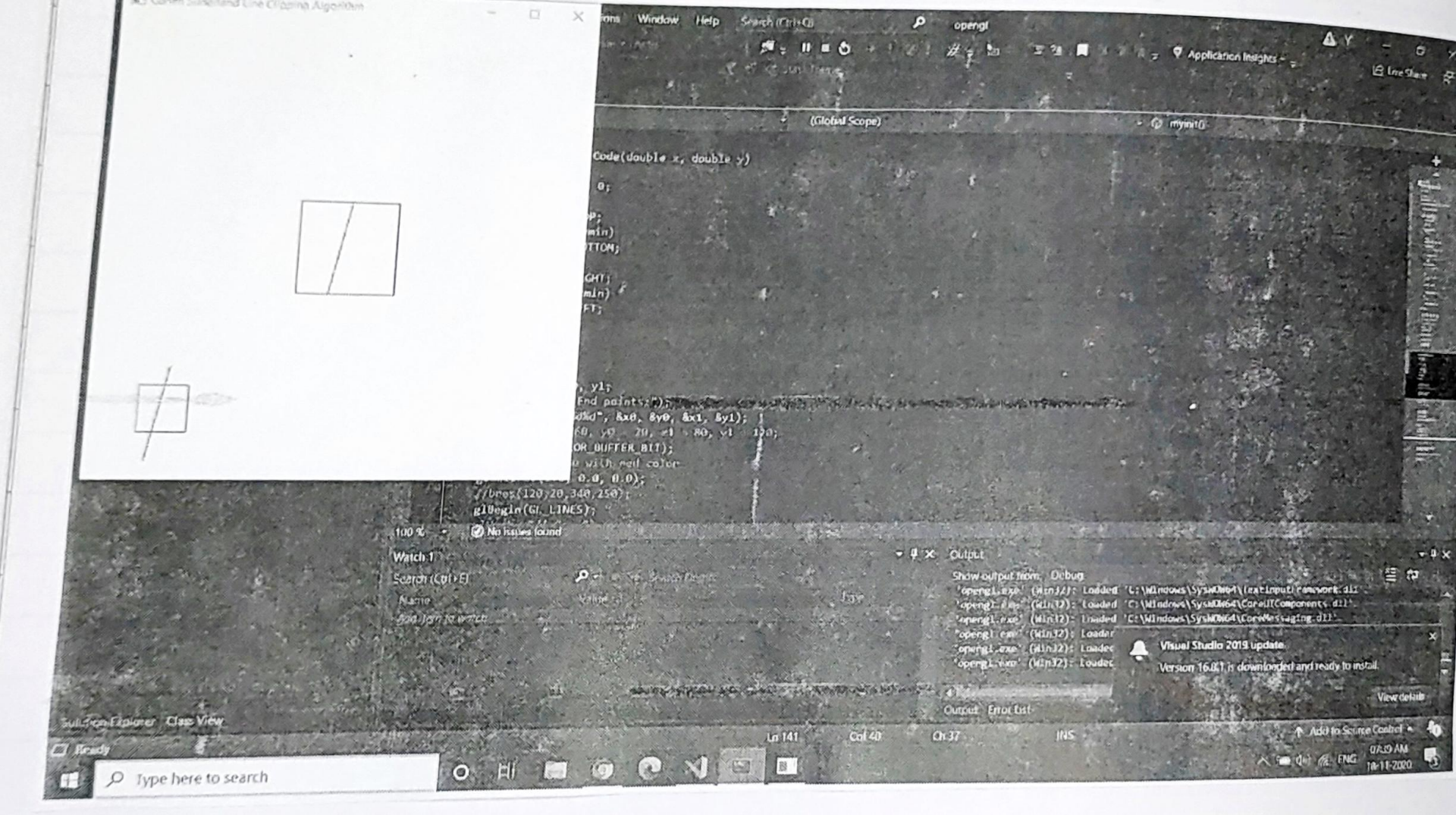
```
            gl Vertex 2f (xvmin, yvmin);
            gl Vertex 2f (xvmax, yvmin);
            gl Vertex 2f (xvmax, yvmax);
            gl Vertex 2f (xvmin, yvmax);
    glEnd ();
    glColor 3f (0.0,0.0,1.0); //draw blue colored clipped line
    gl Begin (GL_LINES);
            gl Vertex 2d (vx0, vy0);
            gl Vertex x2d (vx1, vy1);
     gl End ();

    }

  }

// Compute the bit code for a point (x, y) using clip
rectangle
// bounded diagonally by (xmin, ymin), and (xmax, ymax)
out Code Compute outcode (double x, double y)

{
        outcode Code = 0;
        if (y > ymax)     //above the clip Window
                Code |= TOP;
        else if (y < ymin) //below the clip Window
                Code |= BOTTOM;
        if (x > xmax)     //to the right of clip Window
                Code |= RIGHT;
        else if (x < xmin) //to the left of Clip Window
```

Teacher's Signature :

```
        code| = LEFT;
    return code;

}

void display ()

{

double X0 = 60; Y0 = 20, X1 = 80, Y1 = 120;
glClear(GL_COLOR_BUFFER_BIT);

// draw the line with red color

glColor3f (1.0, 0.0, 0.0);
//bres (120, 20, 340, 250);
glBegin(GL_LINES);
                glVertex2d(X0, Y0);
                glVertex2d(X1, Y1);
        glEnd ();

// draw a blue colored window
glColor3f(0.0, 0.0, 1.0);

glBegin GL_LINE_LOOP);
  glvertex2f (xmin, ymin);
  glvertex2f(xmax, ymin);
  glVertex2f(xmax, ymax);
  glvertex 2f(xmin, ymax);
```

```
glEnd ();
Cohen Suther land Line Clip And Draw
(X0, Y0, X1, Y1);
glFlush ();

}
void myinit ()

{
    glClear Color (1.0, 1.0, 1.0, 1.0);
    glColor 3f (1.0, 0.0, 0.0);
    glpoint Size (1.0);
    glMatrix Mode (GL_PROJECTION);
    glLoad Identity ();
    gluOrtho 2D (0.0, 499.0, 0.0, 499.0);

}
void main (int argc, char ** argv)

{
    glutnit ( & argc, argv);
    glutInit Display Mode (GLUT_SINGLE GLUT
                           RGB);
    glut Init windoro Size (500, 500);
    glut Create Window (cohen Suder land Line
      Clipping Algorithm");
    glut Display Func (display);
    Myinit ();
    glut Main Loop ();

}
```

Teacher's Signature :


Window
Help
Search (Ctrl+Q)
opengl
Application Insights
Live Share
(Global Scope)
myinit()
Code(double x, double y)
End points:
%d", &x0, &y0, &x1, &y1);
OR_BUFFER_BIT);
0.0, 0.0);
//bres(120,20,340,250);
glBegin(GL_LINES);
100 %
No issues found
Watch 1
Search (Ctrl+E)
Name
Value
Type
Add item to watch
Output
Show output from: Debug
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\CoreUIComponents.dll'
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\CoreMessaging.dll'
'opengl.exe' (Win32): Loaded
'opengl.exe' (Win32): Loaded
'opengl.exe' (Win32): Loaded
Visual Studio 2019 update
Version 16.8.1 is downloaded and ready to install.
View details
Output
Error List
Solution Explorer
Class View
Ready
Ln 141
Col 40
Ch 37
INS
Add to Source Control
Type here to search
ENG
07:39 AM
10-11-2020

PROGRAM 6

6. 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.

```
// Liang-Barsky Line Clipping Algorithm With
Window to Viewport Mapping */
#include <stdio.h>
#include <GL/glut.h>

doreble xmin =50, ymin =50, xmax=100, ymax=100;
                                    /Window boundaries
double xv min =200, yv min =200, xv max =300,
yv max = 300; // viewport boundaries.

int Cliptest (doreble P, dorble q, double *t1,
                                    double *t2)
{ double t = q/p;
  if (p <0.0) // Potentially enry point, update te

  {
        if (t > *t1) *t1=t;
        if (t > *t2) return (false); // line portion is
                                    out side


}
else
if (p > 0.0) // portentially leaving point, Update t1
```

Date

Expt. No. Page No.

```
    {
        if (t< *t2) *t2 =t ;
        if (t< *t1) return (false); //line portion is outside

    }
    else
        if (p==0.0)

        {

        }
return (true);

}
void LiangBarskyLineClipAndDraw(double x0, double
        y0, double x1, double y1)

{
    double dx=x1-x0, dy-y0, te=0.0, t1=1.0;
    if (cliptest(-dx, x0-xmin, &te, &t1)) //inside test
                                    wart left edge
    if(cliptest(dx, xmax-x0, &te, &t1)) // insid test
                                    wart right edge
    if(cliptest(-dy, y0-ymin, &te, &t1)) // inside
                                    test wart bottom edge
    if(cliptest(-dy, y0-ymin, &te, &t1)) // inside
                                    test wart top edge
    {
        if(t1<1.0)
```

Teacher's Signature :

Date

Expt. No. Page No.

```
    {
        x1 = x0 + t1 * dx;
    }
    if (te > 0.0)
    { x0 = x0 + te * dx;
      y0 = y0 + te * dy;
    }
```

//Window to viewport mappings

```
double sx = (xvmax - xvmin) / (xmax - xmin);
                    // Scale parameters
double sy = (yvmax - yvmin) / (ymax - ymin);
double vx0 = xvmin + (x0 - xmin) * sx;
double vy0 = yvmin + (y0 - xmin) * sy;
double vx1 = xvmin + (y1 - ymin) * sx;
double vy1 = yvmin + (y1 - ymin) * sy;
        /draw a red colored viewport
glColor3f(1.0, 0.0, 0.0);
glBegin(GL_LINE_LOOP);
        glVertex2f(xvmin, yvmin);
        glVertex2f(xvmax, yvmin);
        glVertex2f(xvmax, yvmax);
        glVertex2f(xvmin, yvmax);
glEnd();
glColor3f(0.0, 0.0, 1.0); //draw blue colored clippe line
```

Teacher's Signature :

Date

Expt. No. Page No.

```
            glBegin (GL_LINES);
                glVertex2d (vx0, vy0);
                glVertex2d (vx1, vy1);
            glEnd ();

        }
    } // end of line clipping

void display ()
{
double x0 = 60, y0 = 20, x1 = 80, y1 = 120;
glClear (GL_COLOR_BUFFER_BIT);
// draw the line with red color
glColor3f (1.0, 0.0, 0.0);
// bres (120, 20, 340, 250);
glBegin (GL_LINES);
                glVertex2d (x0, y0);
                glVertex2d (x1, y1);
            glEnd ();

// draw a blue colored window
glColor3f (0.0, 0.0, 1.0);

 glBegin (GL_LINE_LOOP);
 glVertex2f (xmin, ymin);
 glVertex2f (xmax, ymin);
 glVertex2f (xmax, ymax);
 glVertex2f (xmin, ymax);
 glEnd ();
```

Teacher's Signature :

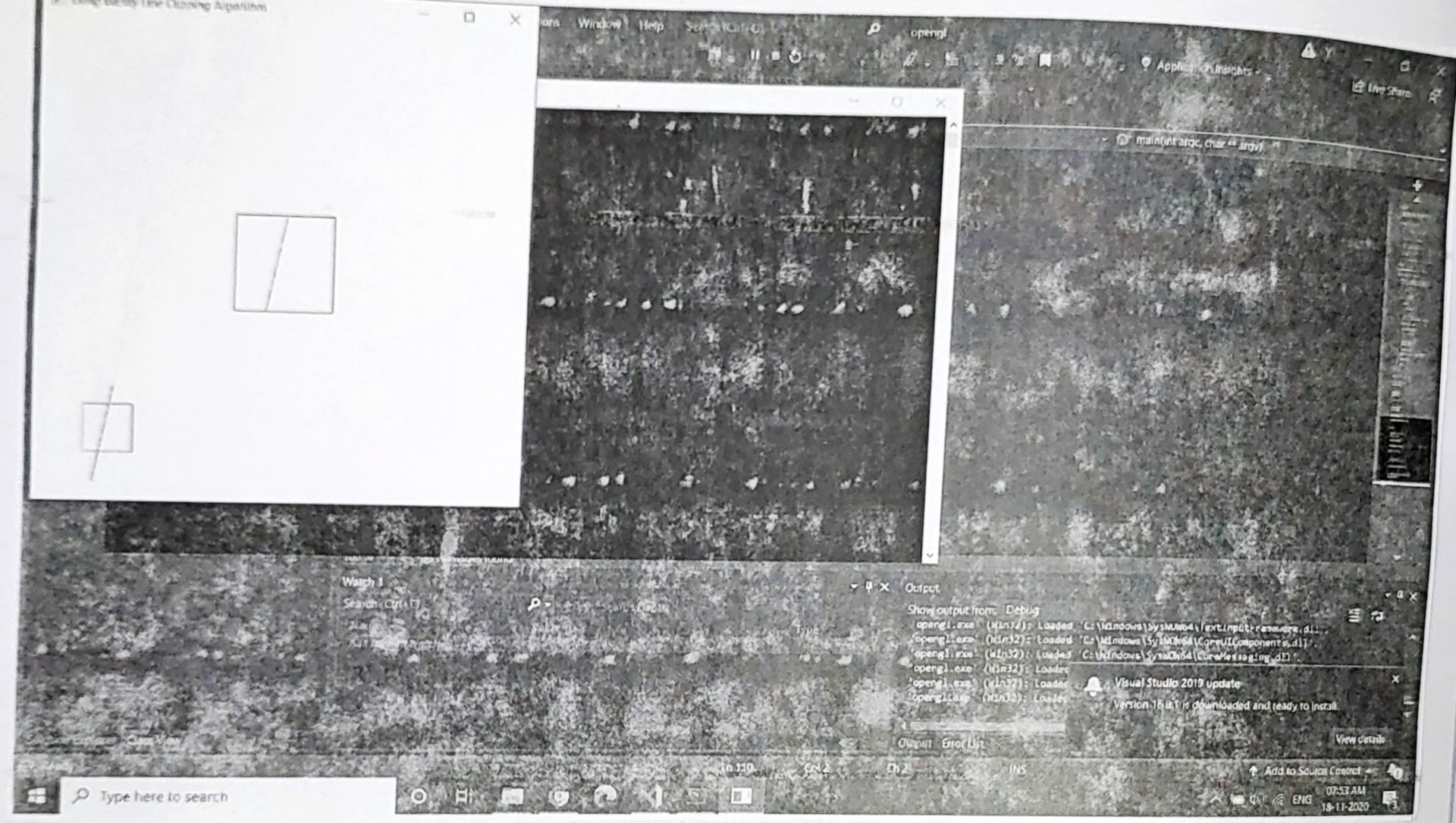
```
Liang Barsky Line clip and draw(x0, y0, x1, y1);
glFlush ();

}
void myinit ()

{
        glClearColor (1.0,1.0,1.0,1.0);
        glColor3f(1.0,0.0,0.0);
        glPointSize(1.0);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity ();
        gluOrtho2D(0.0, 499.0, 0.0, 499.0);

}

void main(int argc, char** argv)

{
 //int x1, x2, y1, y2;
 //printf("Enter End Points:");
 //scanf("%d %d %d %d", &x1, &x2, &y1, &y2);

glutInit(&argc, argv);
glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
glutInitWindowSize(500,500);
glutInitWindowPosition (0,0);
glutCreateWindow("Liang Barsky Line Clipping
                  Algorithm");
```

Date

Expt. No. Page No.

```
glutDisplayFunc(display);
myinit();
glutMainLoop();

}
```

Teacher's Signature :


Line Clipping Algorithm
Window
Help
opengl
Live Share
main(int argc, char** argv)
Watch 1
Output
Show output from: Debug
opengl.exe (Win32): Loaded
Visual Studio 2019 update
Version 16.8.1 is downloaded and ready to install.
View details
Output
Error List
Ln 110
Col 2
Ch 2
INS
Add to Source Control
Type here to search
ENG
07:53 AM
18-11-2020

Date

Expt. No. Page No.

Program 7

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

```
#include <windows.h>
#include <gl/glut.h>

Struct point {
    float x, y;
} w[4], Over[4];
int NOut;

void draw poly (point P[], int n) {
    glBegin(GL_POLYGON);
    for(int i=0; i<n; i++)
        glVertex2f(P[i].x, P[i].y);
    glEnd();

}

bool inside ver (point P) {
    if (P.x >= w[0].x && (P.x <= w[2].x))
        if((P.y >= w[0].y) && (P.y < w[2].y))
            return true;
    return false;
```

Teacher's Signature :

```
}

void addVer(Point P){
    Over[NOut] = P;
    NOut = NOut+1;

}

Point getInterSect(point S, Point P, int edge){
        Point in;
        float m;
        if(w[edge].x == w[[(edge+1)%4].x){ //
                                    Vertical Line
            m = (P.Y - S.Y)/(P.X - S.X);
            in.x = w[edge].x;
            in.y = in.x * m + S.Y;

        }
        else{ //Horizontal Line
            m = (P.Y - S.Y)(P.X - S.X);
            in.y = w[edge].Y;
            in.x = (in.Y - S.Y)/m;

        }
        return in;

}

void clip and Draw(point in ver[], int Nin){
```

Date

Expt. No. Page No.

```
Point S, P, interSec;
for (int i=0; i<4; i++)

{
  NOut = 0;
  S = inver[Nin-1];
  for (int j=0; j<Nin; j++)

  {
    P = inver[j];
    if (insidever(P) == true) {
      if (insidever(S) == true) {
        addver(P);

    }
    else {
      interSec = getInterSect(S, P, i);
      addver(interSec);
      addver(P);

    }

}

else {
  if (insidever(S) == true) {
    interSec = getInterSect(S, P, i);
    addver(interSec);
}
```

Teacher's Signature :

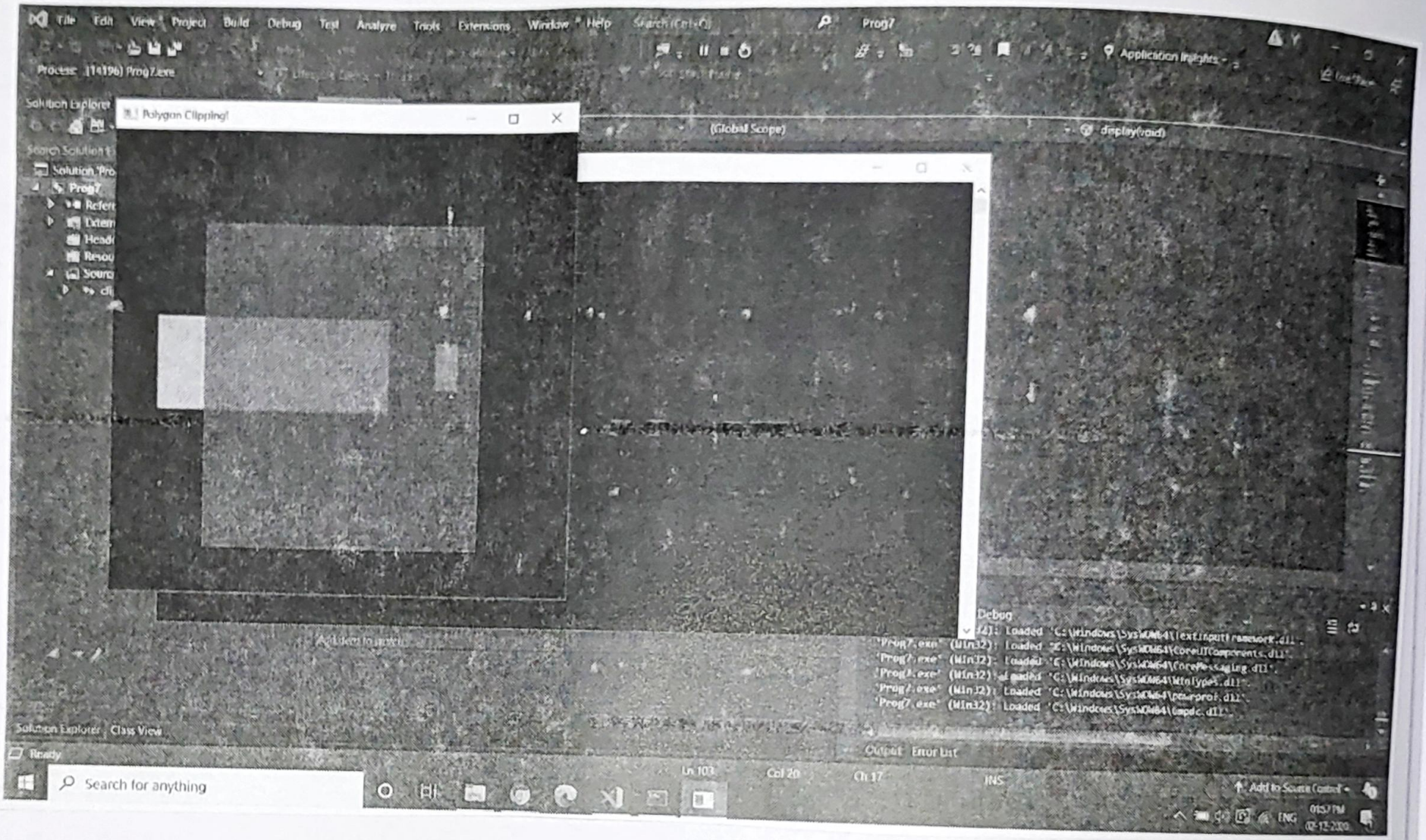
```
    }
    s=p;

  }
  inVer=0 Ver;
  NIn=NOut;

 }
 Draw Poly (0 Ver, 4);


}

void init (){

    gl Clear Color(0.0f,0.0f,0.0f,0.0f);
    gl MatrixMode(GL_PROJECTION);
    gl LoadIdentity();
    glOrtho(0.0,100.0,0.0,100.0,0.0,100.0);
    gl Clear(GL_COLOR_BUFFER_BIT);
    W[0].x=20,W[0].y=10;
    W[1].x=20,W[1].y=80;
    W[2].x=80,W[2].y=80;
    W[3].x=80,W[3].y=10;

  }
  void display (void ){
     point inver[4];
     init ();
```

Date

Expt. No. Page No.

```
// As window for clipping
glColor3f(1.0f,0.0f,0.0f);
drawpoly(w,4);
// As Rect

glcolor3f(0.0f,1.0f,0.0f);
inver[0].x=10, inver[0].Y=40;
inven[1].x=10, inver[1].Y=60;
inver[2].x=60, inver[2].Y=60;
inver[3].x=60, inver[3].Y=40;
drawpoly(inver,4);
// As Rect
glColor3f(0.0f,0.0f,1.0f);
Clip And Draw(inver,4);
// print
glFlush();

}

int main(int argc, Char *argv[]){
    glutInit(&argc,argv);
    glutInitDisplayMode(GLUT_SINGLE_GLUT_RGB);
    glutInitWindowSize(400,400);
    glutInitWindowPosition(100,100);
    glutCreateWindow("polygon clipping!");
    glutDisplayFunc(display);
    glutMainLoop();
    return 0;
}
```

Teacher's Signature :


Polygon Clipping!
Prog7
(Global Scope)
display(void)
Debug
Solution Explorer
Class View
Ready
Search for anything

PROGRAM 8

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

```
#include <GL/glut.h>
#include<windows.h>
float X1, X2, X3, X4, Y1, Y2, Y3, Y4;
void edgedetect (float X1, float Y1, float X2, Y2,
                 int *le, int *re)
{
    float mx, x, temp;
    int i;
    if ((Y2 - Y1) < 0)
    {
        temp = Y1; Y1 = Y2; Y2 = temp;
        temp = X1; X1 = X2; X2 = temp;
    }
    if ((Y2 - Y1) != 0)
    mx = (X2 - X1) / (Y2 - Y1);
    else mx = X2 - X1;
    x = X1;
    for (i = Y; i <= Y2; i++)
    {
```

Date

Expt. No. Page No.

```
        if(x<(float)le[i])
        le[i]=(int)x;
        if(x>(float)re[i])
        re[i]=(int)x;
        x+=mx;

    }

}
void draw_pixel(int x,int y)
{

    glColor3f(1.0,0.0,0.0);
    Sleep(10);
    glBegin(GL_POINTS);
    glVertex2i(x,y);
    glEnd();
    glFlush();
}
void scanfill(float x1,float y1,float x2,float y2,
              float x3,float y3,float x4,float y4)
{
    int le[500],re[500],i,y;
    for(i=0;i<500;i++)
    {
        le[i]=500;
        re[i]=0;
    }
    edgedetect(x1,y1,x2,y2,le,re);
```

Teacher's Signature :

Date

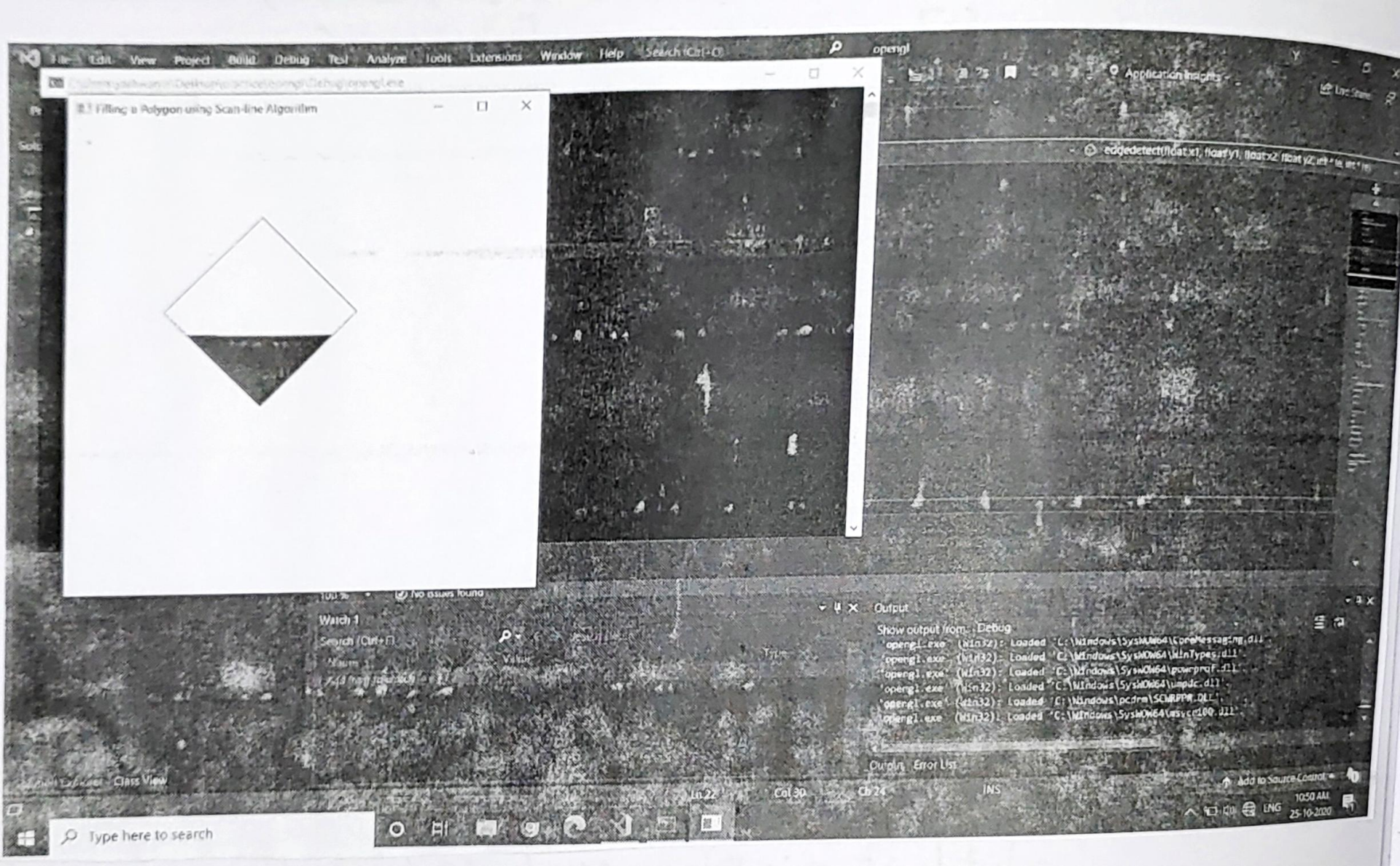
Expt. No. Page No.

```
edgedetect(x2,y2,x3,y3,le,re);
edgedetect(x3,y3,x4,y4,le,re);
edgedetect(x4,y4,x1,y1,le,re);
for(y=0;y<500;y++)
{
        if(le[y]<=re[y])
        for(i=int)le[y];i<(int)re[y];i++)
        draw_Pixel(i,y);

}
}
void display()

{

    x1=200.0;y1=200.0;x2=100.0;y2=300.0;
                x3=200.0;y3=400.0;x4=300.3
                    y4=300.0;
  glClear(GL_COLOR_BUFFER_BIT);
  glColor3f(0.0,0.0,1.0);
  glBegin(GL_LINE_Loop);
        glvertex2f(x1,y1);
        glvertex2f(x2,y2);
        glvertex2f(x3,y3);
        glvertex2f(x4,y4);
  glEnd();
  scanfill(x1,y1,x2,y2,x3,y3,x4,y4);
  glFlush();
```

Teacher's Signature :

```
}

void myinit ()

{
        glClearColor(1.0,1.0,1.0,1.0);
        glColor3f(1.0,0.0,0.0);
        glMatrixMode(GL_PROJECTION);
        glLoadIdentity();
        gluOrtho2D(0.0,499.0,0.0,499.0);

}
void main(int argc, char ** argv)

{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE_
                        GLUT_RGB);
    glutInitWindowSize(500,500);
    glutCreateWindow("Filling a polygon
                      Scan-line Algorithm");
    glutDisplayFunc(display);
    myinit();
    glutMainLoop();

}
```

Teacher's Signature : _____


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
opengl
Application Insights
Live Share
8.3 Filling a Polygon using Scan-line Algorithm
edgedetect(float x1, float y1, float x2, float y2
Watch 1
Search (Ctrl+E)
Name
Value
Type
No issues found
Output
Show output from: Debug
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\CoreMessaging.dll'
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\WinTypes.dll'
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\powrprof.dll'
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\umpdc.dll'
'opengl.exe' (Win32): Loaded 'C:\Windows\pcdrm\SCMRPPM.DLL'
'opengl.exe' (Win32): Loaded 'C:\Windows\SysWOW64\msvcr100.dll'
Output
Error List
Solution Explorer
Class View
Ln 22
Col 30
Ch 24
INS
Add to Source Control
Type here to search
LNG
10:50 AM
25-10-2020

PROGRAM 9

9. Write a program to Create a color cube and spin it using OpenGL transformations.

```
#include <GL/glut.h>

GLfloat vertices [8][3] ={ {-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[8][3] = {   {0.0,0.0,1.0},{1.0,0.0,1.0},
                           {1.0,1.0},{0.0,1.0,1.0},
                           {0.0,0.0,0.0},{1.0,0.0,0.0},
                           {1.0,1.0,0.0} {0.0,1.0.0}
                       };

GLfloat theta[] = {0.0,0.0,0.0};
GLint axis =2;
GLdouble viewer[] = {0.0,0.0,5.0}; /* initial viewer
                                      location */
void polygon(int a, int b, int c, int d)
{
    glBegin(GL_POLYGON);
```

Date

Expt. No. Page No.

```
        glColor3fv(colors[a]);
        glVertex3fv(vertices[a]);
        glColor3fv(colors[b]);
        glVertex3fv(vertices[b]);
        glColor3fv(colors[c]);
        glVertex3fv(vertices[c]);
        glColor3fv(colors[d]);
        glVertex3fv(vertices[d]);
    glEnd();

}

void colorcube()
{
    Polygon(0,3,2,1);// front face - Counter clockwise
    Polygon(4,5,6,7);// back face - clockwise

    Polygon(2,3,7,6);// front face - Counter clockwise
    Polygon(1,5,4,0);// back face - clockwise

    Polygon(1,2,6,5);// front face - Counter clockwise
    Polygon(0,4,7,3);// back face - clockwise

}

Void display(void)
{
    glClear(GL_COLOR_BUFFER_BIT|GL_DEPTH_
                BUFFER_BIT);
```

Teacher's Signature :

```
// Update Viewer Position in modelview matrix
glLoadIdentity ();
gluLookAt (viewer[0], viewer[1], viewer[2], 0.0, 0.0, 0.0,
                                        1.0, 0.0, 0.0);
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);
colorcube ();
glFlush ();
glutSwapBuffers ();

}
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;
     theta[axis] += 2.0;
     if (theta[axis] > 360.0) theta[axis] = 360.0;
     display ();

}
```

Date

Expt. No. Page No.

```
void Keys (unsined char key, int x, int y)
{
    if(key == 'X') viewer[0] -= 1.0;
    if(key == 'x') viewer[0] += 1.0;
    if(key == 'Y') viewer[1] -= 1.0;
    if(key == 'y') viewer[1] += 1.0;
    if(key == 'Z') viewer[2] -= 1.0;
    if(key == 'z') viewer[2] += 1.0;
     display ();

}
Void my Reshape (int w, int h)

{
        glViewport (0,0,w,h);
        /* Use a perspective view */
        gl Matrix Mode (GL_PROJECTION);
        gl LoadIdentity ();
        if (w<=h)
         gl Frustum (-2.0,2.0,-2.0*(GLfloat)h/
                (GL float)w,2.0*(GLfloat)h/(GLfloat)
                w,2.0,20.0);
        else
        gl Frustum(-2.0,2.0,-2.0*(GLfloat)w/(GL
         float)h,2.0*(GLfloat]w/(GLfloat)h,2.0,20.0);
        gl MatrixMode (GL_MODELVIEW);

}
Void main (int argc, Char **argv)
```

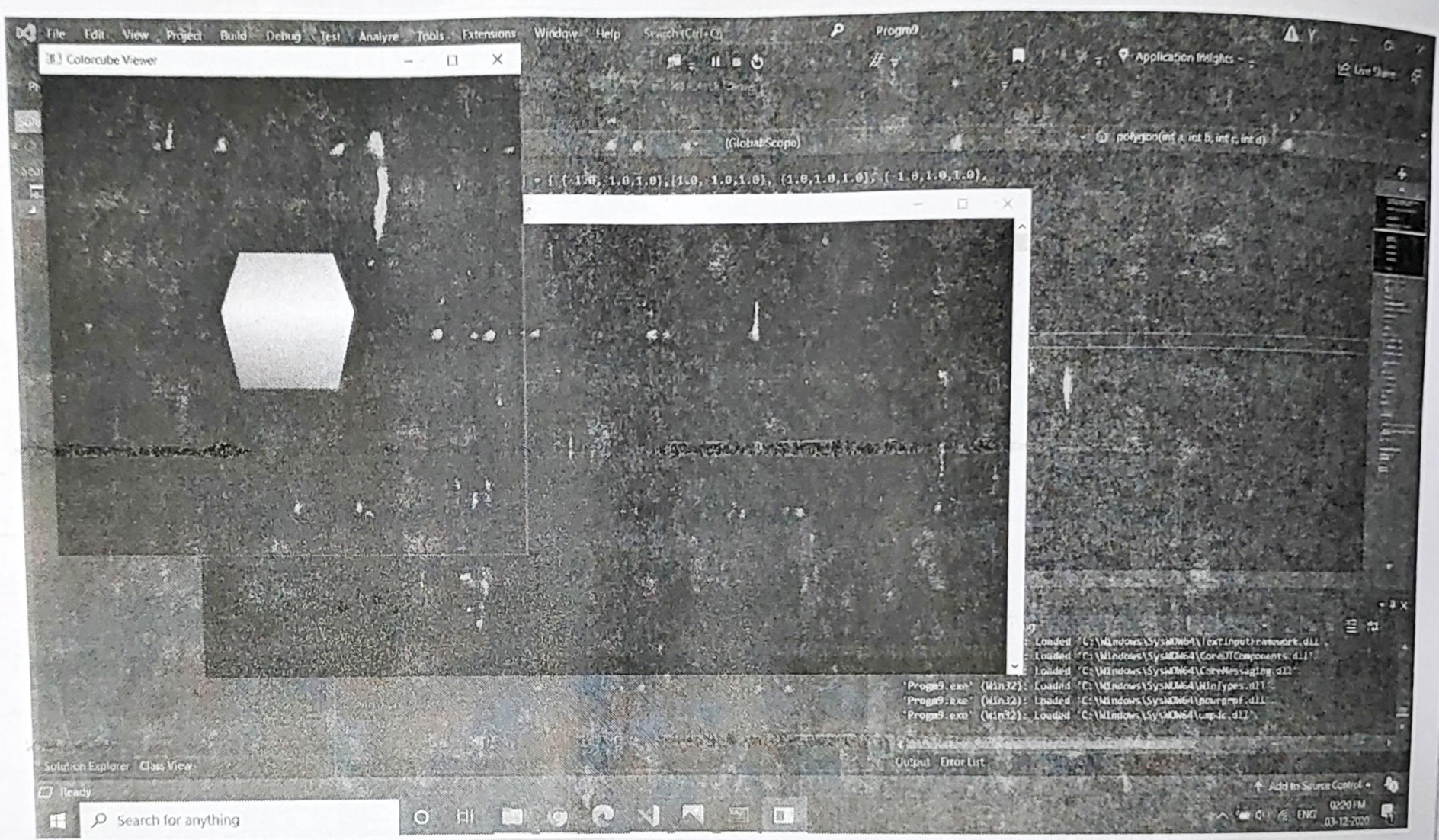
Teacher's Signature :

Date

Expt. No. Page No.

```
{
    glutInit(&argc, argv);
    glutnit Display mode (GLUT_DOUBLE|GLUT_RGB|
                          GLUT_DEPTH);
    glut Window Size (500,500);
    glut Create Window ("Colorcube Viewer");
    glut Reshape Func (my Reshape);
    glut Display Fanc (display);
    glut Mouse Func (mouse);
    glut keyboard Func (Keys);
    glEnable(GL_DEPTH_TEST);
    glut Main Loop ();

}
```

Teacher's Signature :


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Search (Ctrl+Q)
Progm9
Application Insights
Live Share
Colorcube Viewer
(Global Scope)
projection(int a, int b, int c, int d)
Solution Explorer Class View
Output Error List
Ready
Add to Source Control
Search for anything
ENG
03-12-2020

Date

Expt. No. Page No.

PROGRAM 10

10. Write a program to model a car like figure using display lists.

```
#include <GL/glut.h> // Header File for The GLUT Library
#include<GL/gl.h> // Header File For The open GL32 Library
#include <GL/glu.h> // Header File For The GLU32 Library
//#include <unistd.h> // Header File For Sleeping.

/* ASCII Code for the escape key. */
# define ESCAPE 27

/* rotation angle for the triangle. */
float rtri = 0.0f;

/* A general OpenGL initialization function. Sets all of the
initial parameters. */
// We Call this right after our OpenGL window is created.
Void InitGL(int Width, int Height )
{

// This will Clear The Back ground color To Black
glClearColor(0.0f,0.0f,0.0f,0.0f);
glClearDepth(1.0);          // Enables clearing of the Depth Buffer
glDepthFUNC(GL_LESS);       // The Type of Depth Test To DO
glEnable(GL_DEPTH_TEST);    // Enables Depth Testing
glShadeModel(GL_SMOOTH);    // Enbles Smooth Color Shading
```

Teacher's Signature :

```
    glu perspective (45.0f, (GLfloat) Width (GLfloat) Height, 0.1f,
                                                    100.0f);
    glMatrixMode (GL_MODELVIEW);
}

/* The function Called When our window is resized (which
   Shouldn't happen, because we're fullScreen) */
void ReSizeGLScene (int Width, int Height)
{
  if (Height == 0)     // Prevent a Divide By Zero if the window
    Height = 1;                                    is Too Small

    glViewport (0.0, Width, Height); // Reset The Current Viewport
                                    and perspective Transformation
    glMatrixMode (GL_PROJECTION);
    glLoadIdentity ();
    glu Perspective (45.0f. (GLfloat) Width (GLfloat) Height, 0.1f,
                                                    100.0f);
    glMatrix Mode (GL_MODELVIEW);
}

float ballX = 0.5f;
float ballY = 0.0f;
float ballZ = 0.0f;

void drawBall (void) {
    glColor3f (0.0, 1.0, 0.0) // Set ball colour
    glTranslatef (ballX, ballY, ballZ); // moving it toward the
                    screen a bit on Creation.
```

Teacher's Signature : _____

Date

Expt. No. Page No.

```
//glRotatef(ballx, ballx, bally, ballz);
glut SoidSphere (0.3, 20, 20); //Create ball.
gl Translatef (ballX+1.5, ballY, ballZ); //moving it toward
                                    the Screen a bit on Creation
glut SolidSphere (0.3, 20, 20); //
}

/* The main drawing function. */
void Draw GL Scene ()
{
gl Translatef (rtri, 0.0f, -6.0f);
gl Clear (GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER Bit);
                        /Clear The Screen and the Depth Buffer
gl Load Identity (); //Rest The view

 gl Translatef (rtri, 0.0f, -6.0f ); //Move left 1.5 Units and
                                    into the Screen 6.0
// glRotatef (rtri, 1.0f, 0.0f, 0.0f); //Rotate the Triangle on the
                                    Y axis
// draw a triangle (in Smooth Coloring mode)
gl Begin (GL_POLYGON);  //Start drawing a Polygon
gl Color 3f (1.0f, 0.0f, 0.0f); //Set the Color TO Red
gl Vertex 3f (-1.0f, 1.0f, 0.0f); //TOP left
 glvertex 3f (0.4f, 1.0f, 0.0f);

//glvertex 3f ();
 gl End (); //We're done with the Polygon (Smooth Color interpolation)
 draw ball ();
```

Teacher's Signature :

```
rtri += 0.005f;    //increase The Rotation variable For The Triangle
if (rtri > 2)
    rtri = -2.0f;
rquad -= 15.0f;    //Decrease Rotation variable For the Quad

//Swap the buffers to display, since double buffering is used.
glutSwapBuffers ();
}

/* The function called whenever a key is Pressed */
void keyPressed (unsigned char key, int x, int y)
{

    /* Sleep to avoid thrashing this procedure */
    // usleep (100);

    /* if escape is Pressed, Kill everything */
    if (key == ESCAPE)
    {

    /* Shut down our window */
    glutDestroyWindow (window);

    /* exit the program... normal termination. */
     exit (0);

    }

}
```

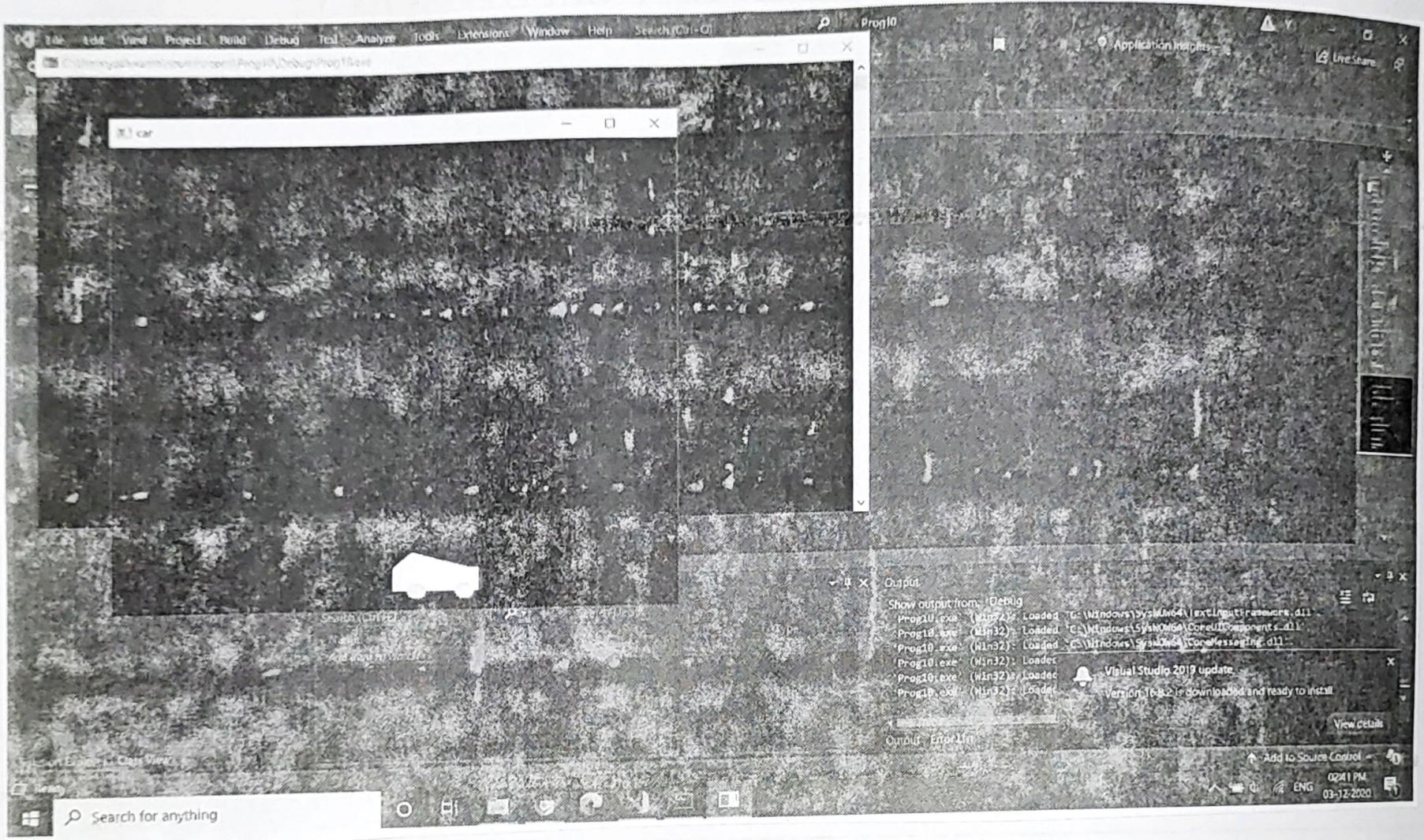
Teacher's Signature : _____

Date

Expt. No. Page No.

```
int main (int argc, Char **argv)
{

glutInit (&argc, argv);

glutInit Display Mode(GLUT_RGBA|GLUT_DOUBLEL
                      GLUT__ALPHA|GLUT-DEPTH);
/* get a 640 x 480 window */
glutInit window Size (640, 480);

/* Open a window */
window = glut Create window ("Moving Car');

/* Resister the function to do all our openGL drawing. */
glut Display Func ( &Draw GL Scene);

/* Go full screen. This is as soon as possible. */
// glutfull Screen ();

/* Register the function called when our window is
                                    relized. */
glut Reshape Func ( &Resize GL Scene);

/* Register the function called when the Key boad is Pressed. */
glut Keyboard Func ( &Key Pressed);
/* Initialize our Window. */
 Init GL (640, 480);
/* Start Event Processing Engine */
 glut Main Loop ();
 return 1;
}
```

Teacher's Signature :


File Edit View Project Build Debug Test Analyze Tools Extensions Window Help
Prog10
Application Insights
Live Share
car
Output
Show output from: Debug
Prog10.exe (Win32): Loaded
Visual Studio 2019 update
Version 16.8.2 is downloaded and ready to install
View details
Add to Source Control
Search for anything
ENG
02:41 PM
03-12-2020

Date

Expt. No. Page No.

PROGRAM 11

11. Write a program to generate a Limacon, Cardiod, Three leaf curve, Spiral.

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

Struct ScreenPt
{
        int x;
        int y;
};
typedef enum {limacon=1, Cardioid=2, threeLeaf=3,
                                    Spiral=4} curveName;
int w=600, h=500;

Void myinit (void)
{
        glClearColor (1.0, 1.0, 1.0);
        glMatrixMode (GL_PROJECTION);
        glOrtho2D (0.0, 200.0, 0.0, 150.0);
}
Void lineSegment (ScreenPt P1, ScreenPt P2)
```

Teacher's Signature :

```
{
    glBegin(GL_LINES);
    glVertex2i(p1.x, p1.y);
    glVertex2i(p2.x, p2.y);
glEnd();

}
void drawCurve(int CureNum)
{

    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];

    glColor3f(0.0,0.0,0.0);
     Curvept[0].x=x0;
    Curvept[0].y=y0;

    Switch(CurveNum)

        case limacon: Curvept[0].x+=a+b; break;
        case cardioid: Curvept[0].x+=a+a; break;
        case threeleaf;     curvept[0].x+=a; break;
        case Spiral: break
        default: break;
}
theta=dtheta;
```

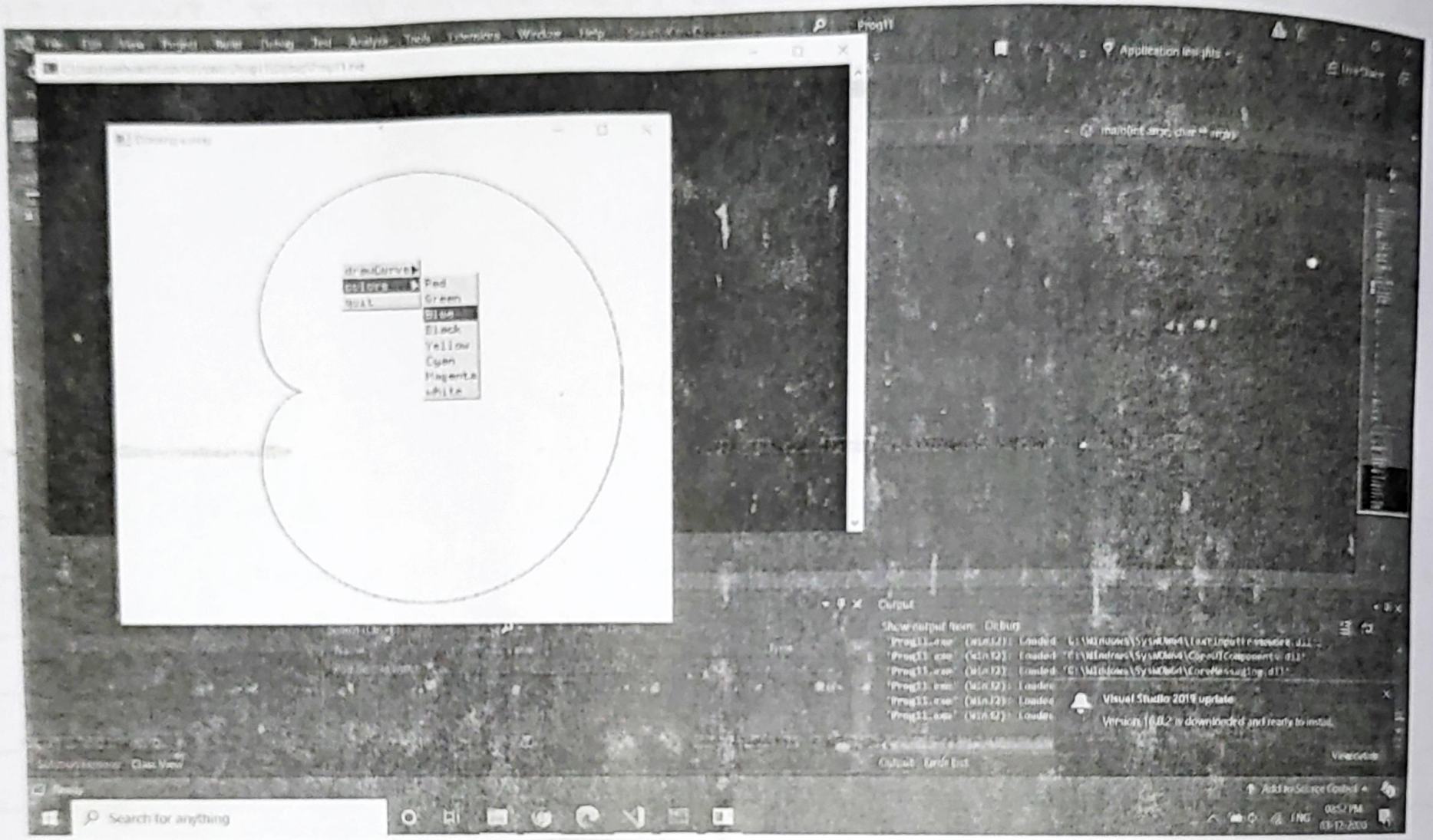
```
While (theta < two Pi)
{
    switch (curveNum)
    {
        case limacon: r = a*cos(theta)+b; break;
        case cardioid: r = a*(1+cos(theta)); break;
        case ThreeLeaf: r = a*cos(3*theta); break;
        case Spiral; r = (a/4.0)*theta; break;
        default:      break;
    }
curvept[1].x = x0 + r* cos(theta);
Curvept[1].y = y0 + r* Sin(theta);
line Segment (Curvept[0], Curve pt[1]);

Curve pt[0].x = curvept[1].x;
Curve pt[0].y = Curve pt[1].y;
theta += dtheta;

}

}

void mydisplay(){
}
```

Date

Expt. No. Page No.

```
void main(int argc, char **argv)
{
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE|GLUT_RGB);
    glutInitWindowSize(w, h);
    glutInitWindowPosition(100, 100);
    glutInitCreateWindow("Drawing Curves");
    myinit();
    glutDisplayFunc(mydisplay);
    glutReshapeFunc(myreshape);

    glutMainLoop();

}
```

Teacher's Signature :


drawCurve
colors
Exit
Red
Green
Blue
Black
Yellow
Cyan
Magenta
White
Output
Visual Studio 2019 update
Search for anything

PROGRAM 12

12. Write a Program to Construct Bezier curve. Control points are supplied through Key board / mouse.

```
#include<iostream>
#include<math.h>
Using nameSpace Std;
#include<gl/glut.h>

float f,g,r,x[4],y[4];

Void myinit()
{
        glclear Color(1.0,1.0,1.0,1.0);
        glColor(1.0,0,0,0,0);
        glPoint Size(5.0);
        gluOrtho 2D(0.0,800,0.0,800);

}
void draw_Pixel(float x, float y)

{

        glBegin(GL_POINTS);
        glvertex2f(x,y);
        glEnd();

}
```

Teacher's Signature : _____

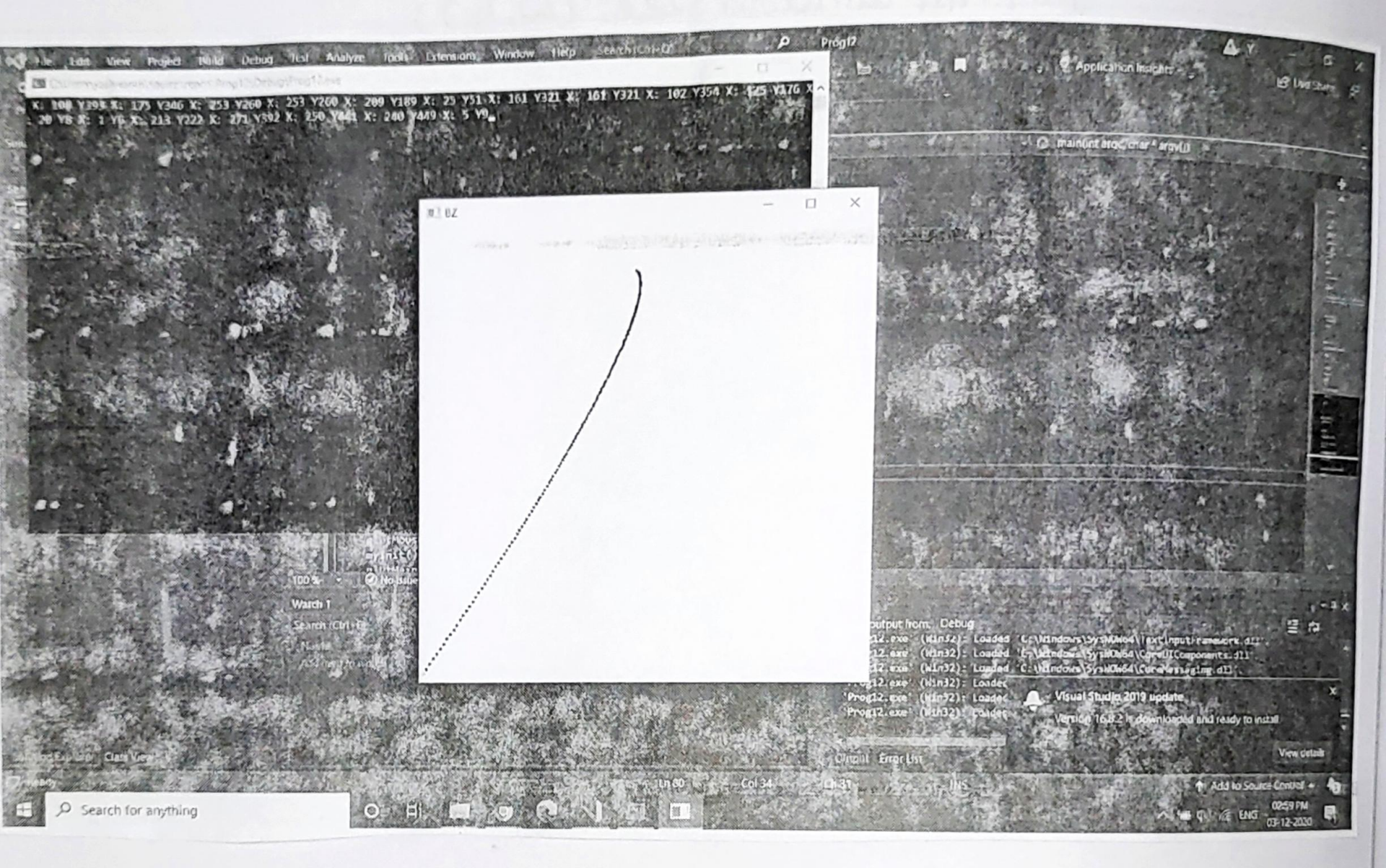
Date

Expt. No. Page No.

```
void display ()
{
    glClear(GL_COLOR_BUFFER_BIT);
    int i;
    double t;
    glColor3f(0.0,0.0,0.0);
    glBegin(GL_POINTS);
    for(t=0.0;t<1.0;t+=0.0005)
    {
        double xt = pow(1-t,3)*x[0]+3*t*Pow(1-1.2)*
        x[1]+3*Pow(t,2)*(1-t)*x[2]+Pow(t,
3)*x[3];
        double yt=Pow(1-t,3)*y[0]+3*t*pow(1+t.2)*
        y[1]+3*Pow(t,2)*1-
t)*y[2]+Pow(t,3)*y[3];
            glVertex2f(xt,yt);
    }
    glColor3f(1.0,1.0,0.0);
    for(i=0;i<4;i++)
        glVertex2f(x[i],y[i]);

    glEnd();
    glFlush();
}
```

Teacher's Signature :

```
cout<<"Enter x coordinates \n";
cin>>x[0]>>x[1]>>[2]>>x[3];
cout<<"Enter y coordinates \n";
cin>>Y[0]>>Y[1]>>Y[2]>>Y[3];
glutInit(&argc, argv);
glutInitDisplaymode(GLUT_SINGLE|GLUT_RGB);
glutInitWindowSize(500,500);
glutInitWindowPosition(100,100);
glutInitCreateWindow("new window");
glutDisplayFunc(display);
 myinit();
 glutmainLoop();

}
```


Prog12
Application Insights
Live Share
main(int argc, char * argv[])
Watch 1
Search (Ctrl+E)
Visual Studio 2019 update
Version 16.3.2 is downloaded and ready to install
View details
Search for anything
Add to Source Control
ENG
03-12-2020