Assignment - 6

Name : Emmadi Sumith Kumar
Branch : CSE - B1
Subject : Computer Graphics
Roll No : UI20CS21

Write a program for polygon clipping using Cohen Hodgeman Clipping Algorithm.

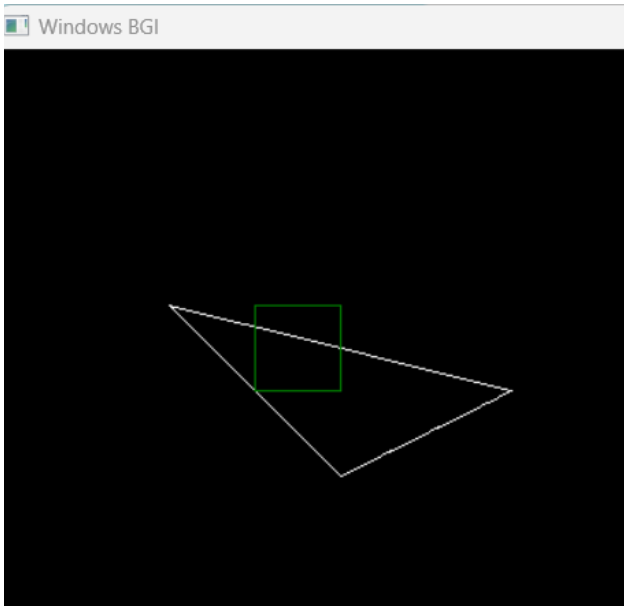
```
#include <iostream>
#include <graphics.h>
using namespace std;

const int MAX_POINTS = 20;

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;
```

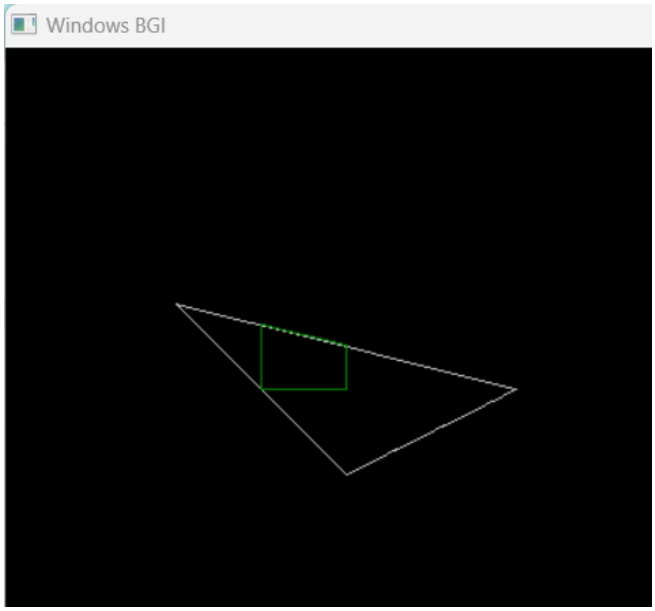
```
    for (int i = 0; i < poly_size; i++)
    {

        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);

        if (i_pos < 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] = 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++;
        }

        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];
    }
}

void suthHodgClip(int poly_points[][2], int poly_size,
                  int clipper_points[][2], int clipper_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]);
    }

    // Printing vertices of clipped polygon
    // for (int i=0; i < poly_size; i++)
    //  cout << '(' << poly_points[i][0] <<
    //          ", " << poly_points[i][1] << ") ";
```

```
    line(poly_points[0][0], poly_points[0][1], poly_points[1][0],
poly_points[1][1]);
    line(poly_points[1][0], poly_points[1][1], poly_points[2][0],
poly_points[2][1]);
    line(poly_points[2][0], poly_points[2][1], poly_points[3][0],
poly_points[3][1]);
    line(poly_points[3][0], poly_points[3][1], poly_points[0][0],
poly_points[0][1]);
}

void drawwindow()
{
    line(150, 100, 450, 100);
    line(450, 100, 450, 350);
    line(450, 350, 150, 350);
    line(150, 350, 150, 100);
}

// Driver code
int main()
{

    int gd = DETECT, gm;
    initgraph(&gd, &gm, NULL);
    int poly_size = 3;

    int poly_points[20][2] = {
        {100, 150},
        {200, 250},
        {300, 200},
    };

    line(100, 150, 200, 250);
    line(200, 250, 300, 200);
    line(300, 200, 100, 150);

    int clipper_size = 4;
    int clipper_points[][2] = {{150, 150}, {150, 200}, {200, 200}, {200, 150}};
```

```
    setcolor(2);
    line(150, 150, 150, 200);
    line(150, 200, 200, 200);
    line(200, 200, 200, 150);
    line(200, 150, 150, 150);
    delay(5000);

    cleardevice();

    setcolor(7);
    line(100, 150, 200, 250);
    line(200, 250, 300, 200);
    line(300, 200, 100, 150);
    setcolor(2);
    suthHodgClip(poly_points, poly_size, clipper_points,
                 clipper_size);
    delay(10000);

    return 0;
}
```


Windows BGI


Windows BGI