**Practical-16**

**Computer Graphics and Multimedia**

1. Write a program to implement Sutherland Hodgeman polygon clipping algorithm.

#include<iostream.h>

#include<conio.h>

#include<graphics.h>

#define round(a) ((int)(a+0.5))

int k;

float xmin, ymin, xmax, ymax, arr[20], m;

void

clipl (float x1, float y1, float x2, float y2)

{

if (x2 - x1)

m = (y2 - y1) / (x2 - x1);

else

m = 100000;

if (x1 >= xmin && x2 >= xmin)

{

arr[k] = x2;

arr[k + 1] = y2;

k += 2;

}

if (x1 < xmin && x2 >= xmin)

{

arr[k] = xmin;

arr[k + 1] = y1 + m \* (xmin - x1);

arr[k + 2] = x2;

arr[k + 3] = y2;

k += 4;

}

if (x1 >= xmin && x2 < xmin)

{

arr[k] = xmin;

arr[k + 1] = y1 + m \* (xmin - x1);

k += 2;

}

}

void

clipt (float x1, float y1, float x2, float y2)

{

if (y2 - y1)

m = (x2 - x1) / (y2 - y1);

else

m = 100000;

if (y1 <= ymax && y2 <= ymax)

{

arr[k] = x2;

arr[k + 1] = y2;

k += 2;

}

if (y1 > ymax && y2 <= ymax)

{

arr[k] = x1 + m \* (ymax - y1);

arr[k + 1] = ymax;

arr[k + 2] = x2;

arr[k + 3] = y2;

k += 4;

}

if (y1 <= ymax && y2 > ymax)

{

arr[k] = x1 + m \* (ymax - y1);

arr[k + 1] = ymax;

k += 2;

}

}

void

clipr (float x1, float y1, float x2, float y2)

{

if (x2 - x1)

m = (y2 - y1) / (x2 - x1);

else

m = 100000;

if (x1 <= xmax && x2 <= xmax)

{

arr[k] = x2;

arr[k + 1] = y2;

k += 2;

}

if (x1 > xmax && x2 <= xmax)

{

arr[k] = xmax;

arr[k + 1] = y1 + m \* (xmax - x1);

arr[k + 2] = x2;

arr[k + 3] = y2;

k += 4;

}

if (x1 <= xmax && x2 > xmax)

{

arr[k] = xmax;

arr[k + 1] = y1 + m \* (xmax - x1);

k += 2;

}

}

void

clipb (float x1, float y1, float x2, float y2)

{

if (y2 - y1)

m = (x2 - x1) / (y2 - y1);

else

m = 100000;

if (y1 >= ymin && y2 >= ymin)

{

arr[k] = x2;

arr[k + 1] = y2;

k += 2;

}

if (y1 < ymin && y2 >= ymin)

{

arr[k] = x1 + m \* (ymin - y1);

arr[k + 1] = ymin;

arr[k + 2] = x2;

arr[k + 3] = y2;

k += 4;

}

if (y1 >= ymin && y2 < ymin)

{

arr[k] = x1 + m \* (ymin - y1);

arr[k + 1] = ymin;

k += 2;

}

}

void

main ()

{

int gdriver = DETECT, gmode, n, poly[20];

float xi, yi, xf, yf, polyy[20];

clrscr ();

cout << "Coordinates of rectangular clip window :\nxmin,ymin :";

cin >> xmin >> ymin;

cout << "xmax,ymax :";

cin >> xmax >> ymax;

cout << "\n\nPolygon to be clipped :\nNumber of sides :";

cin >> n;

cout << "Enter the coordinates :";

for (int i = 0; i < 2 \* n; i++)

cin >> polyy[i];

polyy[i] = polyy[0];

polyy[i + 1] = polyy[1];

for (i = 0; i < 2 \* n + 2; i++)

poly[i] = round (polyy[i]);

initgraph (&gdriver, &gmode, "C:\\TC\\BGI");

setcolor (RED);

rectangle (xmin, ymax, xmax, ymin);

cout << "\t\tUNCLIPPED POLYGON";

setcolor (WHITE);

fillpoly (n, poly);

getch ();

cleardevice ();

k = 0;

for (i = 0; i < 2 \* n; i += 2)

clipl (polyy[i], polyy[i + 1], polyy[i + 2], polyy[i + 3]);

n = k / 2;

for (i = 0; i < k; i++)

polyy[i] = arr[i];

polyy[i] = polyy[0];

polyy[i + 1] = polyy[1];

k = 0;

for (i = 0; i < 2 \* n; i += 2)

clipt (polyy[i], polyy[i + 1], polyy[i + 2], polyy[i + 3]);

n = k / 2;

for (i = 0; i < k; i++)

polyy[i] = arr[i];

polyy[i] = polyy[0];

polyy[i + 1] = polyy[1];

k = 0;

for (i = 0; i < 2 \* n; i += 2)

clipr (polyy[i], polyy[i + 1], polyy[i + 2], polyy[i + 3]);

n = k / 2;

for (i = 0; i < k; i++)

polyy[i] = arr[i];

polyy[i] = polyy[0];

polyy[i + 1] = polyy[1];

k = 0;

for (i = 0; i < 2 \* n; i += 2)

clipb (polyy[i], polyy[i + 1], polyy[i + 2], polyy[i + 3]);

for (i = 0; i < k; i++)

poly[i] = round (arr[i]);

if (k)

fillpoly (k / 2, poly);

setcolor (RED);

rectangle (xmin, ymax, xmax, ymin);

cout << "\tCLIPPED POLYGON";

getch ();

closegraph ();

}

