Experiment 11

(Cohen Sutherland)

Definition:

The Cohen-Sutherland line clipping algorithm is a method for clipping a line segment against a rectangular region. The algorithm uses a divide-and-conquer strategy to efficiently clip the line segment.

The Cohen-Sutherland algorithm is efficient because it quickly identifies and discards line segments that are completely outside the clipping rectangle without calculating the intersection points. For line segments that are partially inside the clipping rectangle, the algorithm calculates only the necessary intersection points.

Algorithm:

1. Assign an outcode to each endpoint of the line segment. An outcode is a 4-bit binary number, where each bit represents a region relative to the clipping rectangle (left, right, top, or bottom). If the point is inside the clipping rectangle, the outcode is 0000 (all bits are 0).
2. Perform a bitwise AND operation on the outcodes of the two endpoints. If the result is non zero, the line segment is completely outside the clipping rectangle, and no further processing is needed.
3. If both outcodes are zero, the line segment is completely inside the clipping rectangle. The original line segment can be drawn without any clipping.
4. If the bitwise AND operation results in zero, the line segment may be partially inside the clipping rectangle. Choose one of the endpoints that has a non-zero outcode (if both have nonzero outcodes, choose either one).
5. Use the chosen endpoint's outcode to determine which edge of the clipping rectangle the line segment intersects. Calculate the intersection point using the slope of the line segment.
6. Replace the chosen endpoint with the intersection point and update its outcode.
7. Repeat steps 2-6 until the line segment is either completely inside, completely outside, or one of the endpoints has been replaced by an intersection point and the other endpoint is inside the clipping rectangle.
8. If the line segment is completely or partially inside the clipping rectangle, draw the clipped line segment.

Code:

```
#include <stdio.h>
#include <graphics.h>
#define INSIDE 0
#define LEFT 1
#define RIGHT 2
#define BOTTOM 4
#define TOP 8
const int screenWidth = 640;
const int screenHeight = 480;
const int clippingXMin = 100;
const int clippingYMin = 100;
const int clippingXMax = 540;
const int clippingYMax = 380;
int computeOutCode(int x, int y) {
int code = INSIDE;
if (x < clippingXMin) {
code |= LEFT;
} else if (x > clippingXMax) {
code |= RIGHT;
}
if (y < clippingYMin) {
code |= BOTTOM;
} else if (y > clippingYMax) {
code |= TOP;
} return code;
}
void cohenSutherlandClip(int x0, int y0, int x1, int y1) {
int outcode0 = computeOutCode(x0, y0);
int outcode1 = computeOutCode(x1, y1);
int accept = 0;
while (1) {
if (!(outcode0 | outcode1)) {
accept = 1;
break;
} else if (outcode0 & outcode1) {
```

```
break;
} else {
int x, y;
int outcodeOut = outcode0 ? outcode0 : outcode1;
if (outcodeOut & TOP) {
x = x0 + (x1 - x0) * (clippingYMax - y0) / (y1 - y0);
y = clippingYMax;
} else if (outcodeOut & BOTTOM) {
x = x0 + (x1 - x0) * (clippingYMin - y0) / (y1 - y0);
y = clippingYMin;
} else if (outcodeOut & RIGHT) {
y = y0 + (y1 - y0) * (clippingXMax - x0) / (x1 - x0);
x = clippingXMax;
} else {
y = y0 + (y1 - y0) * (clippingXMin - x0) / (x1 - x0);
x = clippingXMin; }
if (outcodeOut == outcode0) {
x0 = x;
y0 = y;
outcode0 = computeOutCode(x0, y0);
} else {
x1 = x;
y1 = y;
outcode1 = computeOutCode(x1, y1);
}
}
}
if (accept) {
line(x0, y0, x1, y1);
}
}
int main() {
int gd = DETECT, gm;
initgraph(&gd, &gm, "");
setbkcolor(WHITE);
```

```
cleardevice();
rectangle(clippingXMin, clippingYMin, clippingXMax, clippingYMax);
int x0 = 50, y0 = 50, x1 = 600, y1 = 400;
setcolor(RED);
line(x0, y0, x1, y1); setcolor(GREEN);
cohenSutherlandClip(x0, y0, x1, y1);
getch();
closegraph();
return 0;
}
```

Output: