Experiment 7

Student Name: Sumit Kumar **UID:22BCS10048**
Branch: B.E CSE **Section/Group: KRG-1/A**
Semester: 6th **Date: 13/03/2025**
Subject Name: Computer Graphics **Subject Code: 22CSH-352**

1. **Aim:**
   Evaluate the 4-bit region code for line endpoints and determine whether the line lies inside or outside the screen.

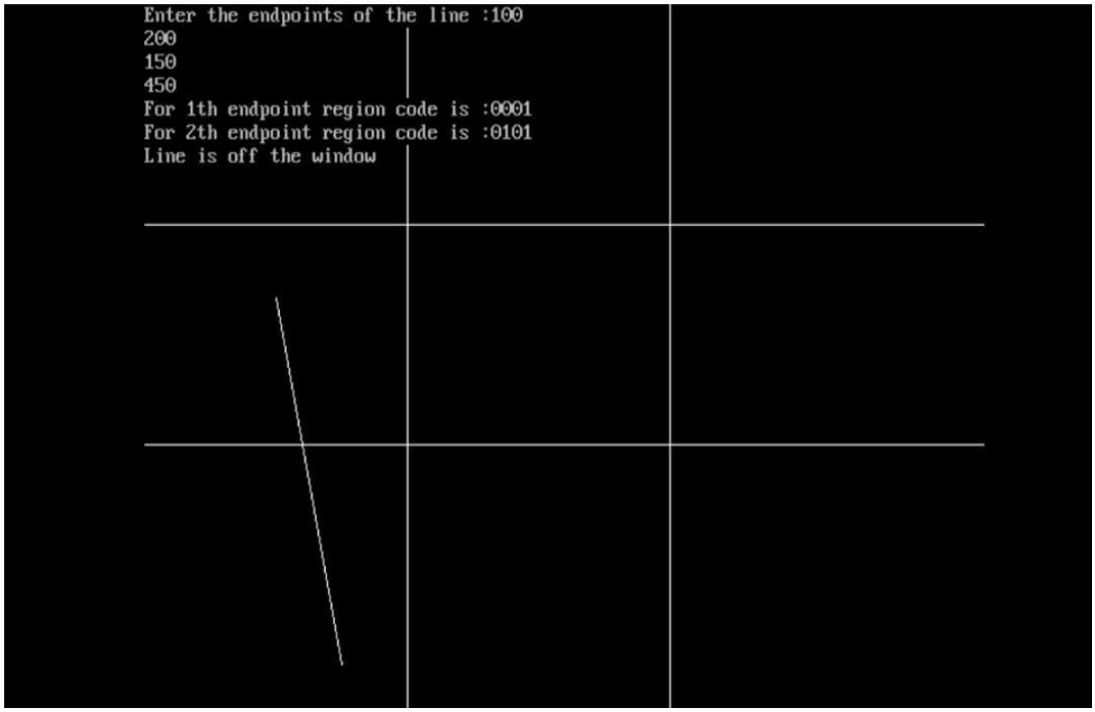
2. **Objective:**
   To calculate and display the 4-bit region code for line endpoints and determine whether the line lies within the screen boundaries.

3. **Implementation/Code:**

```
#include<iostream.h>
#include<conio.h>
#include<graphics.h>

void main() {
   int gd = DETECT, gm;
   initgraph(&gd, &gm, "C:\\TURBOC3\\BGI"); // Corrected BGI path

   // Defining clipping window boundaries
   int xmax = 400, ymax = 300, xmin = 200, ymin = 150;

   // Drawing clipping window boundary
   line(xmin, 0, xmin, getmaxy()); // Left boundary
   line(xmax, 0, xmax, getmaxy());  // Right boundary
   line(0, ymax, getmaxx(), ymax);  // Top boundary
   line(0, ymin, getmaxx(), ymin); // Bottom boundary

   // Getting user input for line endpoints
   cout << "Enter the endpoints of the line: ";
   int x[2], y[2], num[2];
   cin >> x[0] >> y[0] >> x[1] >> y[1];

   // Drawing the original line
   setcolor(WHITE);
```

```
    line(x[0], y[0], x[1], y[1]);

    // Calculating region codes
    for (int i = 0; i < 2; i++) {
        int bit1 = 0, bit2 = 0, bit3 = 0, bit4 = 0;

        if (y[i] < ymin) bit1 = 1; // Below ymin
        if (y[i] > ymax) bit2 = 1; // Above ymax
        if (x[i] > xmax) bit3 = 1; // Right of xmax
        if (x[i] < xmin) bit4 = 1; // Left of xmin

        // Printing region codes
        cout << "For " << i << "th endpoint region code is: "
             << bit1 << bit2 << bit3 << bit4 << endl;

        num[i] = bit4 * 1 + bit3 * 2 + bit2 * 4 + bit1 * 8;
    }

    // Checking if the line is completely inside, partially inside, or outside
    if (!(num[0] | num[1])) {
        cout << "Line is completely inside the window." << endl;
    } else if (!(num[0] & num[1])) {
        cout << "Line needs to be clipped." << endl;
    } else {
        cout << "Line is completely outside the window." << endl;
    }

    getch();
    closegraph();
}
```

4. Output



5. Learning Outcome

i. Draws a clipping window with boundaries at (xmin, xmax, ymin, ymax).
ii. Determines region codes for both endpoints of the given line.
iii. Classifies the line into three categories:
   a) Completely inside → No clipping needed.
   b) Partially inside → Clipping is required.
   c) Completely outside → Line is rejected.