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 must able to draw as many lines and specify inputs through keyboard/mouse.**

#define BLACK 0

#include <stdio.h>

#include <GL/glut.h>

GLint Point[4] = { 0 };

int x1, x2, y1, y2;

void draw\_pixel(int x, int y, int value)

{

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

}

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;

if (x2 < x1)incx = -1;

incy = 1;

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 += inc2;

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)

{

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(0.0, 1.0, 0.0);

glPointSize(2.5);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(0.0, 499.0, 0.0, 499.0);

}

int main(int argc, char\*\* argv)

{

// int x1, x2, y1, y2;

printf("Enter points: x1, y1, x2, y2\n");

scanf\_s("%d %d %d %d", &x1, &x2, &y1, &y2);

Point[0] = x1;

Point[1] = y1;

Point[2] = x2;

Point[3] = y2;

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(500, 500);

glutInitWindowPosition(0, 0);

glutCreateWindow("Bresenham,s Algorithm");

glutDisplayFunc(display);

myinit();

glutMainLoop();

return 0;

}

Output:



