**SSN College of Engineering**

**Department of Computer Science and Engineering**

## UCS1712 – GRAPHICS AND MULTIMEDIA LAB

**Ex 3 – Drawing 2D Primitives –Line – Bresenham’s Algorithm**

SATHEESH KUMAR G R

185001136

**1]AIM:**

To plot points that make up the line with endpoints (x0,y0) and (xn,yn) using Bresenham’s line drawing algorithm.

**ALGORITHM:**

1. Find dy and dx and calculate m.

2. Slope m < 1:

a. Find p = 2\*dy - dx

b. Run a while loop till x == x2

i. Plot the pixel for the points x and y.

ii. Increment x value

iii. If p is less than 0, update p += 2 \* dy

iv. If p is greater than 0, increment y and update p += 2 \* dy - 2 \* dx.

3. If slope m >= 1:

a. Find p = 2\*dx-dy

b. Run a while loop till y == y2

i. Plot the pixel for the points x and y.

ii. Increment y value

iii. If p is less than 0, update p += 2 \* dx

iv. If p is greater than 0, increment x and update p += 2 \* dx - 2 \* dy

CODE:

#include <gl/glut.h>

#include <stdio.h>

int x1, y1, x2, y2;

int arrx1[500],arry1[500],arrx2[500],arry2[500];

void myInit() {

    glClear(GL\_COLOR\_BUFFER\_BIT);

    glClearColor(0.0, 0.0, 0.0, 0.0);

    glMatrixMode(GL\_PROJECTION);

    gluOrtho2D(-200, 200, -200, 200);

}

void draw\_pixel(int x, int y) {

    glBegin(GL\_POINTS);

    glColor3ub(200,100,200);

    glVertex2i(x, y);

    glEnd();

}

void draw\_line(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);

        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);

        }

    } else {

        draw\_pixel(x, y);

        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);

        }

    }

}

void myDisplay() {

    for(int i =0;i<2;i++)

    {

    x1 = arrx1[i];

    x2 = arrx2[i];

    y1 = arry1[i];

    y2 = arry2[i];

    draw\_line(x1, x2, y1, y2);

    glFlush();

    }

}

int main(int argc, char \*\*argv) {

    for(int i=0;i<2;i++)

    {

        printf("Enter the points for line: %d\n",i+1);

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

        printf("Slope of line: %lf\n",(y1-y2 + 0.0)/(x1-x2 + 0.0));

        arrx1[i] = x1;

        arrx2[i] = x2;

        arry1[i] = y1;

        arry2[i] = y2;

    }

    glutInit(&argc, argv);

    glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB);

    glutInitWindowSize(400, 400);

    glutInitWindowPosition(0, 0);

    glutCreateWindow("Bresenham's Line Drawing");

    myInit();

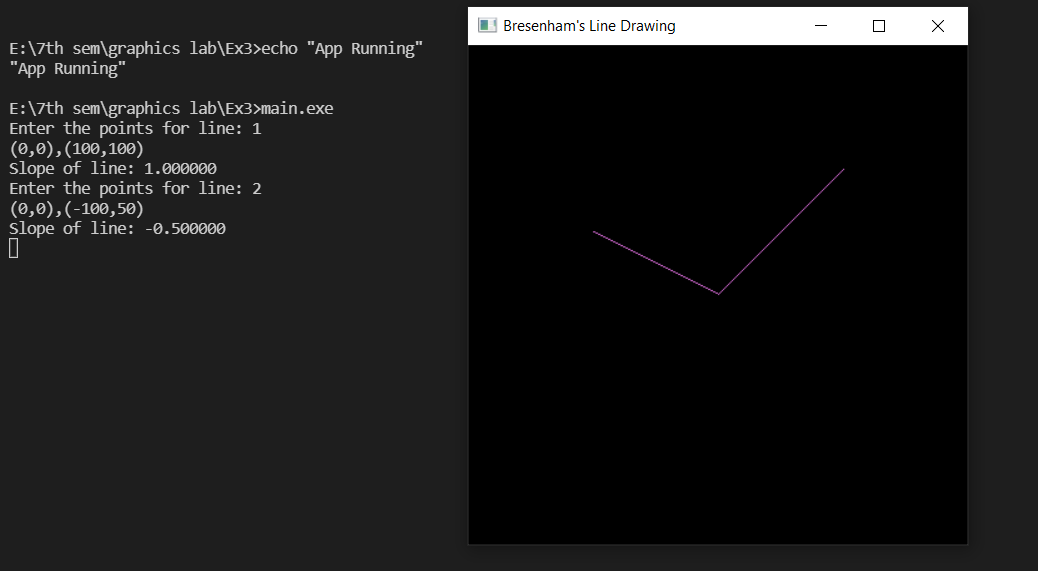
    glutDisplayFunc(myDisplay);

    glutMainLoop();

    return 0;

}

**OUTPUT:**



**RESULT:**

Thus, the lines were drawn using Bresenham Line Drawing Algorithm.

**2]AIM:**

To write a C++ program using OPENGL to write any Alphabet (using sleeping, slanting, standing lines) with the help of Bresenham’s line drawing algorithm.

**ALGORITHM:**

1. Get inputs for the required alphabet to be drawn.
2. Find dy and dx and calculate m.
3. Slope m < 1:
   1. Find p = 2\*dy - dx
   2. Run a while loop till x == x2
      1. Plot the pixel for the points x and y.
      2. Increment x value
      3. If p is less than 0, update p += 2 \* dy
      4. If p is greater than 0, increment y and update p += 2 \* dy - 2 \* dx.
4. If slope m >= 1:

a. Find p = 2\*dx-dy

b. Run a while loop till y == y2

i. Plot the pixel for the points x and y.

ii. Increment y value

iii. If p is less than 0, update p += 2 \* dx

iv. If p is greater than 0, increment x and update p += 2 \* dx - 2 \* dy

**CODE:**

#include <gl/glut.h>

#include <stdio.h>

int x1, y1, x2, y2;

int arrx1[500],arry1[500],arrx2[500],arry2[500];

void myInit() {

    glClear(GL\_COLOR\_BUFFER\_BIT);

    glClearColor(0.0, 0.0, 0.0, 0.0);

    glMatrixMode(GL\_PROJECTION);

    gluOrtho2D(-200, 200, -200, 200);

}

void draw\_pixel(int x, int y) {

    glBegin(GL\_POINTS);

    glColor3ub(200,100,200);

    glVertex2i(x, y);

    glEnd();

}

void draw\_line(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);

        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);

        }

    } else {

        draw\_pixel(x, y);

        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);

        }

    }

}

void myDisplay() {

    for(int i =0;i<2;i++)

    {

    x1 = arrx1[i];

    x2 = arrx2[i];

    y1 = arry1[i];

    y2 = arry2[i];

    draw\_line(x1, x2, y1, y2);

    glFlush();

    }

}

int main(int argc, char \*\*argv) {

    for(int i=0;i<2;i++)

    {

        printf("Enter the points for line: %d\n",i+1);

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

        arrx1[i] = x1;

        arrx2[i] = x2;

        arry1[i] = y1;

        arry2[i] = y2;

    }

    glutInit(&argc, argv);

    glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB);

    glutInitWindowSize(400, 400);

    glutInitWindowPosition(0, 0);

    glutCreateWindow("Alphabet T");

    myInit();

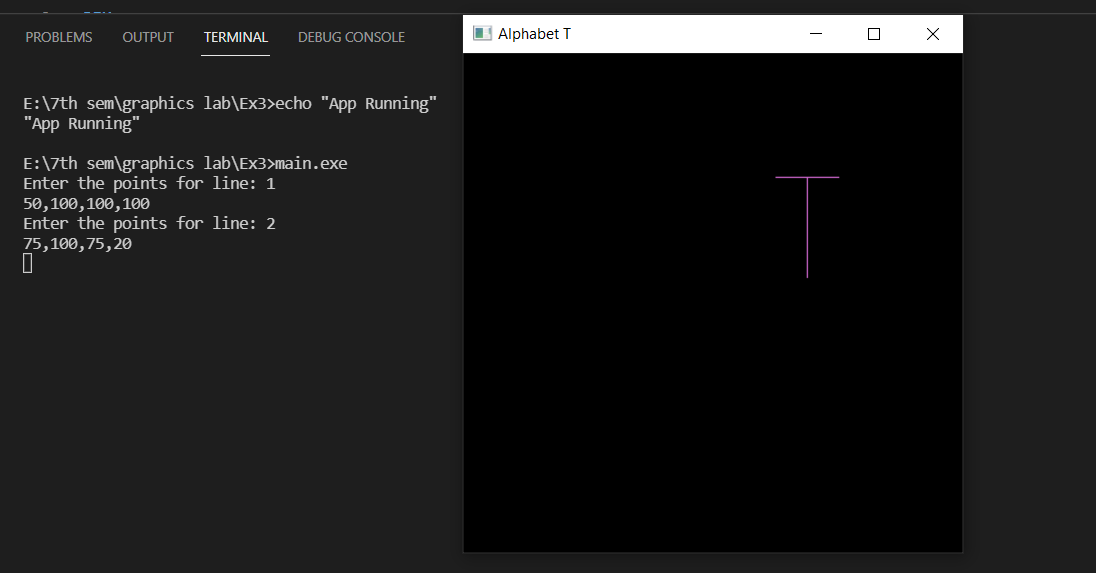
    glutDisplayFunc(myDisplay);

    glutMainLoop();

    return 0;

}

OUTPUT:



RESULT:

Thus, an alphabet was drawn using Bresenham Line Drawing Algorithm.