Name : SAMRUDDHI NEVAGI

Roll no : 23540

**CG ASSIGNMENT – 2**

**CODE:**

#include <GL/glut.h>

#include <iostream>

#include <math.h>

using namespace std;

int ch = 0;

void display(int x, int y)

{

glColor3f(0,0,0);

glPointSize(2);

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

}

void displaydotted(int x, int y)

{

glColor3f(0,0,0);

glPointSize(2);

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

}

void displaydashed(int x, int y)

{

glColor3f(0,0,0);

glPointSize(3);

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

}

void displaysolid(int x, int y)

{

glColor3f(0,0,0);

glPointSize(6);

glBegin(GL\_POINTS);

glVertex2i(x, y);

glEnd();

}

void SimpleLine(int x1, int y1, int x2, int y2){

float dx = x2 - x1;

float dy = y2 - y1;

int steps;

if(abs(dx) >= abs(dy)){

steps = abs(dx);

}

else{

steps = abs(dy);

}

float Xin = dx / (float) steps;

float Yin = dy / (float) steps;

float x = x1;

float y = y1;

for(int i=0; i<=steps; i++){

display(x, y);

x = x + Xin;

y = y + Yin;

}

glFlush();

}

void DottedLine(int x1, int y1, int x2, int y2){

float dx = x2 - x1;

float dy = y2 - y1;

int steps;

if(abs(dx) >= abs(dy)){

steps = abs(dx);

}

else{

steps = abs(dy);

}

float Xin = dx / (float) steps;

float Yin = dy / (float) steps;

float x = x1;

float y = y1;

for(int i=0; i<=steps; i++){

if(i % 6 == 0){

displaydotted(x, y);

}

x = x + Xin;

y = y + Yin;

}

glFlush();

}

void DashLine(int x1, int y1, int x2, int y2){

float dx = x2 - x1;

float dy = y2 - y1;

int steps;

if(abs(dx) >= abs(dy)){

steps = abs(dx);

}

else{

steps = abs(dy);

}

float Xin = dx / (float) steps;

float Yin = dy / (float) steps;

float x = x1;

float y = y1;

for(int i=0; i<=steps; i++){

if(i % 8 > 4){

displaydashed(x, y);

}

x = x + Xin;

y = y + Yin;

}

glFlush();

}

void SolidLine(int x1, int y1, int x2, int y2){

float dx = x2 - x1;

float dy = y2 - y1;

int steps;

if(abs(dx) >= abs(dy)){

steps = abs(dx);

}

else{

steps = abs(dy);

}

float Xin = dx / (float) steps;

float Yin = dy / (float) steps;

float x = x1;

float y = y1;

for(int i=0; i<=steps; i++){

displaydashed(x, y);

x = x + Xin;

y = y + Yin;

}

glFlush();

}

void mouse(int button, int state, int x, int y)

{

static int x1, y1, pt = 0;

if (button == GLUT\_LEFT\_BUTTON && state == GLUT\_DOWN)

{

if (pt == 0)

{

x1 = x;

y1 = y;

pt = pt + 1;

}

else

{

if (ch == 1)

{

SimpleLine(x1, y1, x, y);

}

else if (ch == 2)

{

DottedLine(x1, y1, x, y);

}

else if (ch == 3)

{

DashLine(x1, y1, x, y);

}

else if(ch == 4)

{

SolidLine(x1, y1, x, y);

}

x1 = x;

y1 = y;

}

}

else if (button == GLUT\_RIGHT\_BUTTON && state == GLUT\_DOWN){

pt = 0;

}

glFlush();

}

void keyboard(unsigned char key, int x, int y)

{

switch (key)

{

case 's':

{

ch = 1;

cout << "Simple Line is opted" << endl;

glutMouseFunc(mouse);

break;

}

case 'd':

{

ch = 2;

cout << "Dotted Line is opted" << endl;

glutMouseFunc(mouse);

break;

}

case 'D':

{

ch = 3;

cout << "Dashed Line is opted" << endl;

glutMouseFunc(mouse);

break;

}

case 'S':

{

ch = 4;

cout << "Solid Line is opted" << endl;

glutMouseFunc(mouse);

break;

}

}

glutPostRedisplay();

}

void initialize()

{

glClearColor(1.0, 1.0, 1.0, 1.0);

glClear(GL\_COLOR\_BUFFER\_BIT);

gluOrtho2D(0, 600, 600, 0);

}

void initialaxis(){

glColor3f(0,0,0);

glLineWidth(2);

glBegin(GL\_LINES);

glVertex2i(300, 0);

glVertex2i(300, 600);

glVertex2i(0, 300);

glVertex2i(600, 300);

glEnd();

glFlush();

glutKeyboardFunc(keyboard);

}

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

{

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE);

glutInitWindowSize(600, 600);

glutInitWindowPosition(800, 100);

glutCreateWindow("DDA Algorithm");

initialize();

cout << "Choose your Line type: " << endl;

cout << "--------------------------------------------" << endl;

cout << "s => simple" << endl;

cout << "d => dotted" << endl;

cout << "D => Dashed" << endl;

cout << "S => Solid" << endl;

cout << "--------------------------------------------" << endl;

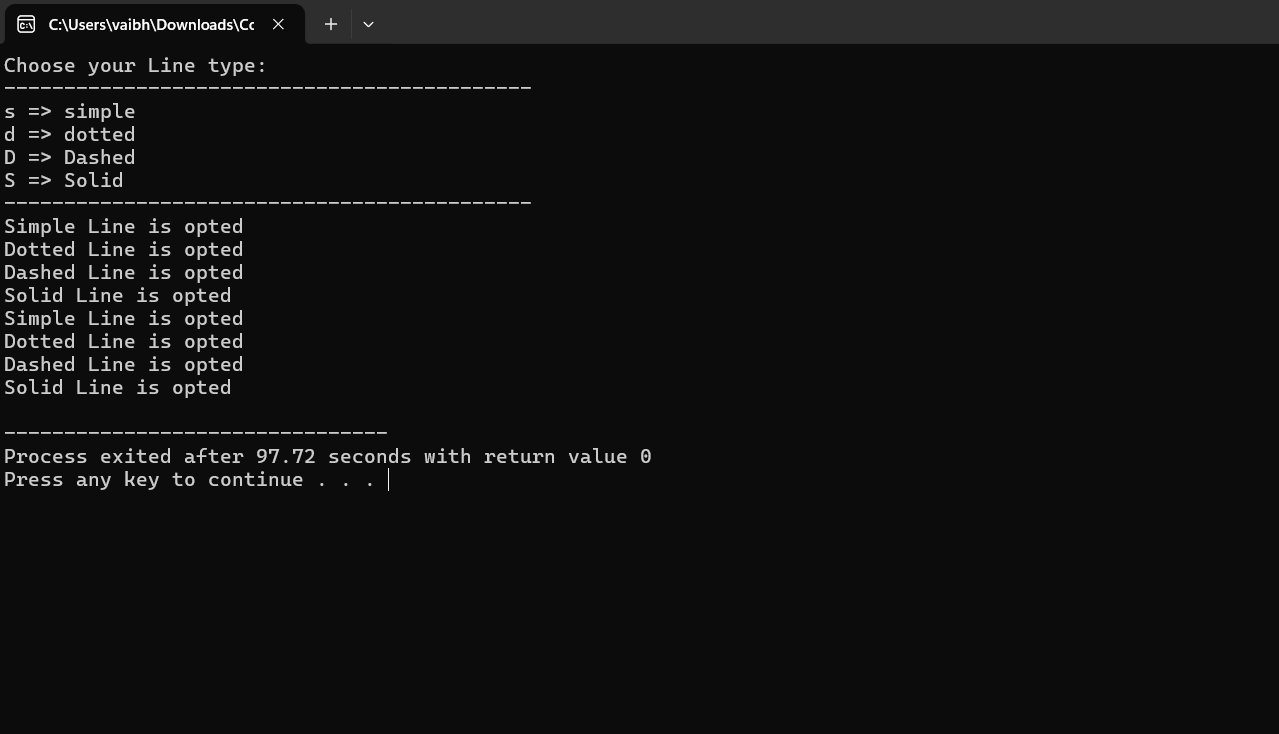
glutDisplayFunc(initialaxis);

glutMainLoop();

return 0;

}

**OUTPUT:**

****

