**SSN COLLEGE OF ENGINEERING**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**UCS1712 – GRAPHICS AND MULTIMEDIA LAB**

**EX NO: 2–** Drawing 2D Primitives –Line – DDA Algorithm

**Name: Nachammai Devi Pooja S**

**RollNo:185001096**

**Date:04/08/2020**

**Aim:**

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

Case 1: +ve slope Left to Right line

Case 2: +ve slope Right to Left line

Case 3: -ve slope Left to Right line

Case 4: -ve slope Right to Left line

Each case has two subdivisions

1. |m|<= 1 (ii) |m|>1

**Algorithm:**

* Step 1: Declare x1,y1,x2,y2,dx,dy,x,y as integer variables.
* Step 2: Enter value of x1,y1,x2,y2.
* Step 3: calculate dx , dy

dx = X1 - X0;

dy = Y1 - Y0;

* Step 4:Depending upon absolute value of dx & dy,choose number of steps to put pixel.

steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy)

steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

* Step 5:calculate increment in x & y for each steps

Xinc = dx / (float) steps;

Yinc = dy / (float) steps;

* Step 6:Set pixel for each step

X = X0;

Y = Y0;

* Step 7 : x = x + xinc y = y + yinc Set pixels (Round (x), Round (y))
* Step 8: Repeat step 8 until x = x2

**Code:**

#include <gl/glut.h>

#include <stdio.h>

#include <iostream>

using namespace std;

const float PI = 3.14;

void drawLine(int x0, int y0, int x1, int y1) {

glBegin(GL\_POINTS);

glColor3f(1.0, 1.0, 1.0);

int tempx, tempy;

if (x1 < x0) {

tempx = x0;

tempy = y0;

y0 = y1;

x0 = x1;

x1 = tempx;

y1 = tempy;

}

double m = (double)(y1 - y0) / (x1 - x0);

double y = (double)y0;

double x = (double)x0;

if (m < 1) {

while (x <= x1) {

glColor3d(1, 0, 0);

if (-m > 1) {

glColor3d(0, 0, 1);

}

glVertex2d(x, floor(y));

//printf("%f %f\n", floor(y), x);

y = y + m;

x++;

}

}

else {

double m1 = 1 / m;

while (y <= y1) {

glColor3d(0, 0, 1);

glVertex2d(floor(x), y);

y++;

x = x + m1;

}

}

glEnd();

}

void init(void) {

glClearColor(0.0, 0.0, 0.0, 0.0);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

gluOrtho2D(-1000, 1000, -1000, 1000);

}

void display(void)

{

glClear(GL\_COLOR\_BUFFER\_BIT);

printf("\nEnter x1 y1:");

int x1, y1, x2, y2;

cin >> x1 >> y1;

printf("\nEnter x2 y2:");

cin >> x2 >> y2;

drawLine(x1, y1, x2, y2);

glutSwapBuffers();

}

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

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_DOUBLE | GLUT\_RGB);

glutInitWindowSize(500, 500);

glutInitWindowPosition(50, 50);

glutCreateWindow("DDA Line Drawing!");

init();

glutDisplayFunc(display);

glutMainLoop();

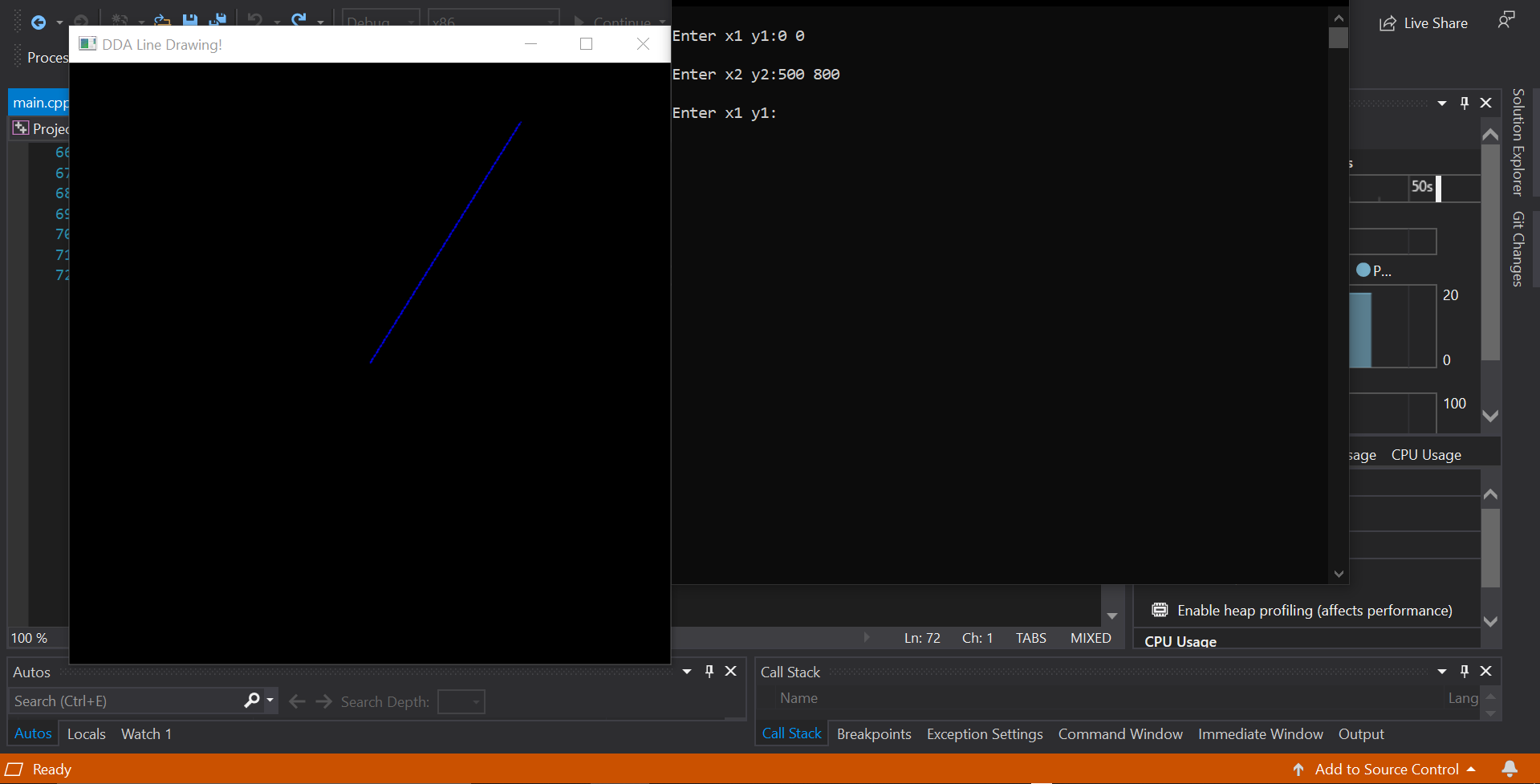
return 0;

}

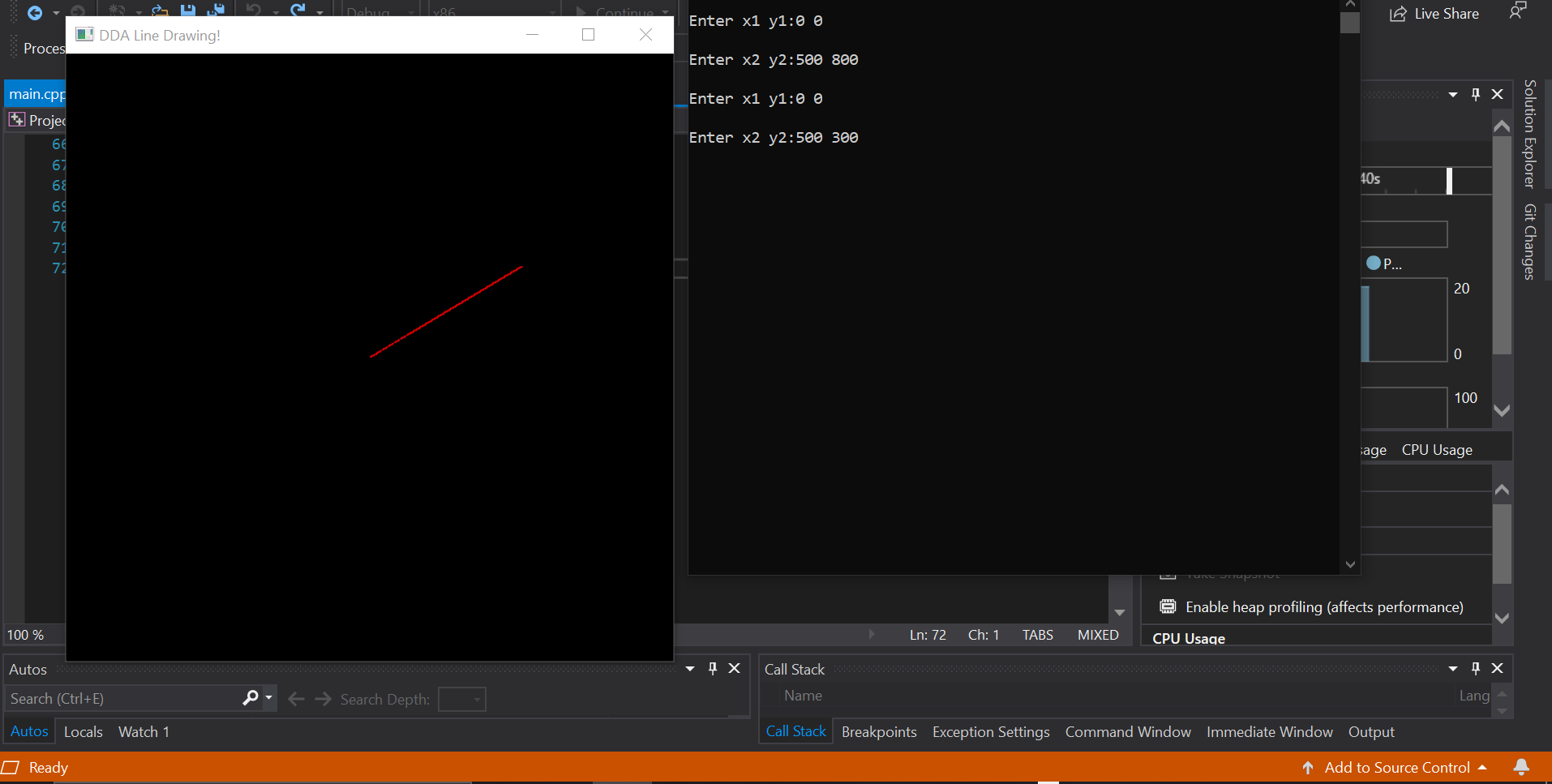
**OUTPUT:**

**Left to Right:**

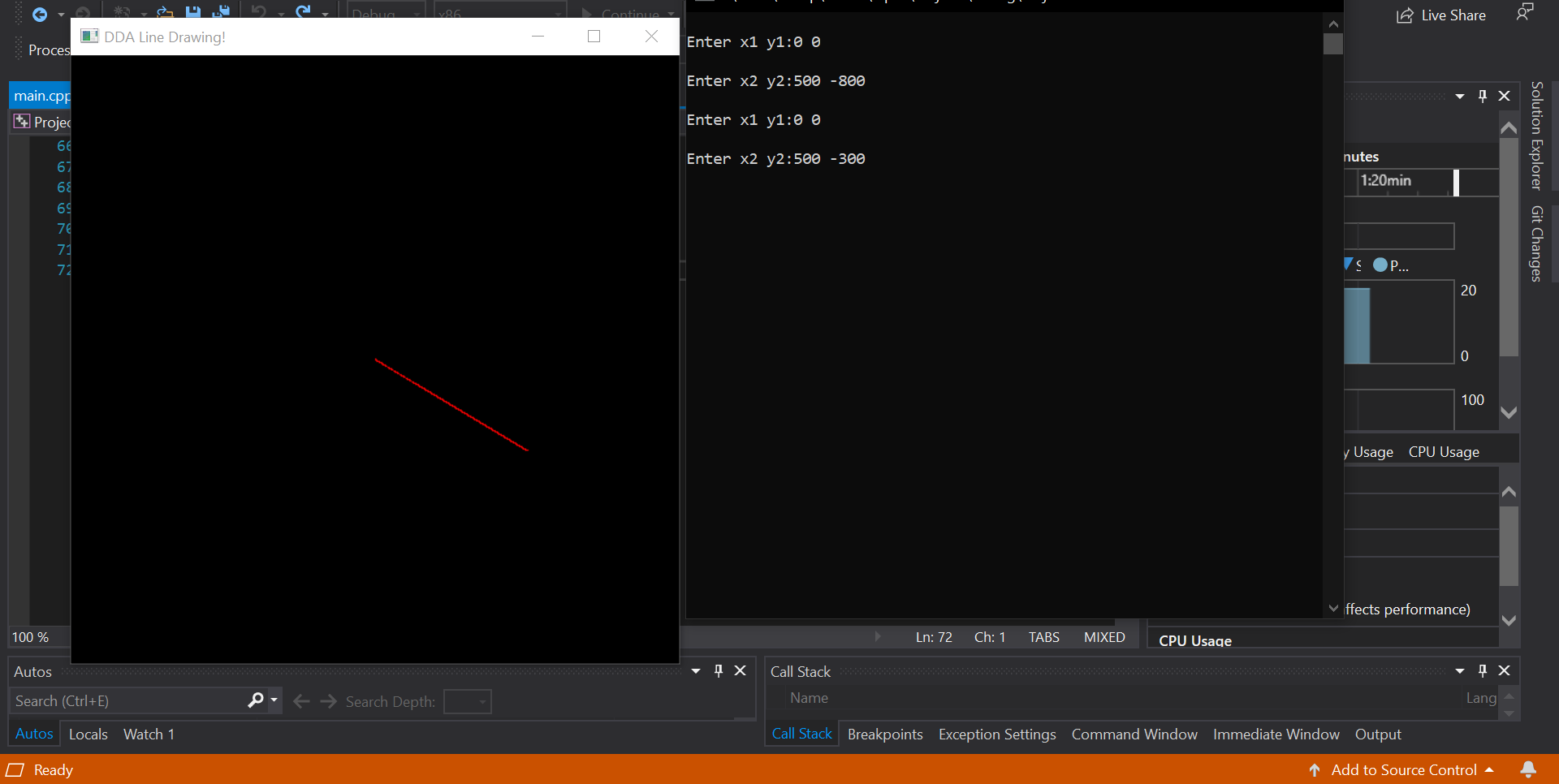
* **+ve slope: m>1**



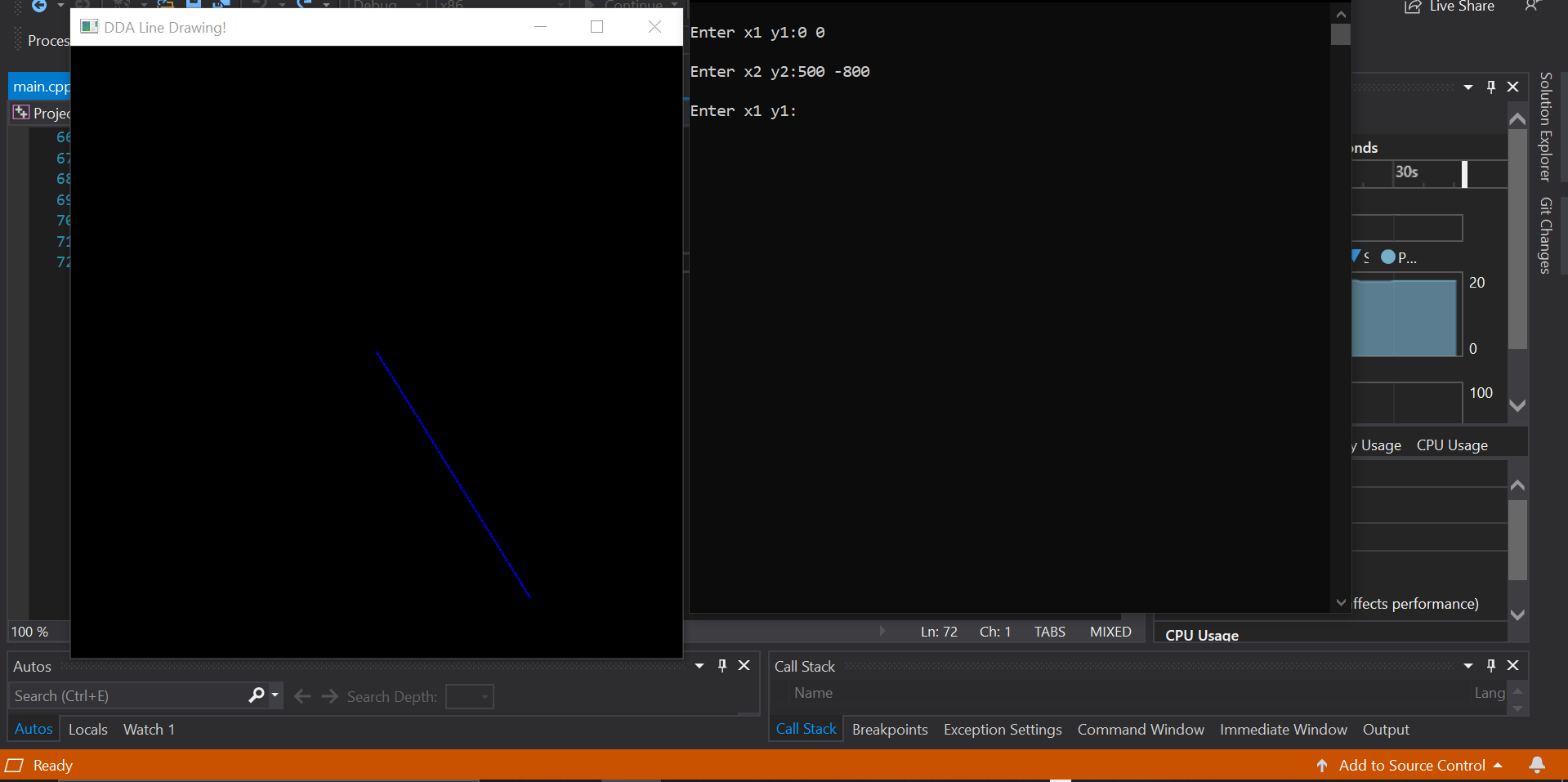
* **+ve slope: m<1**



* **-ve slope: m>1**

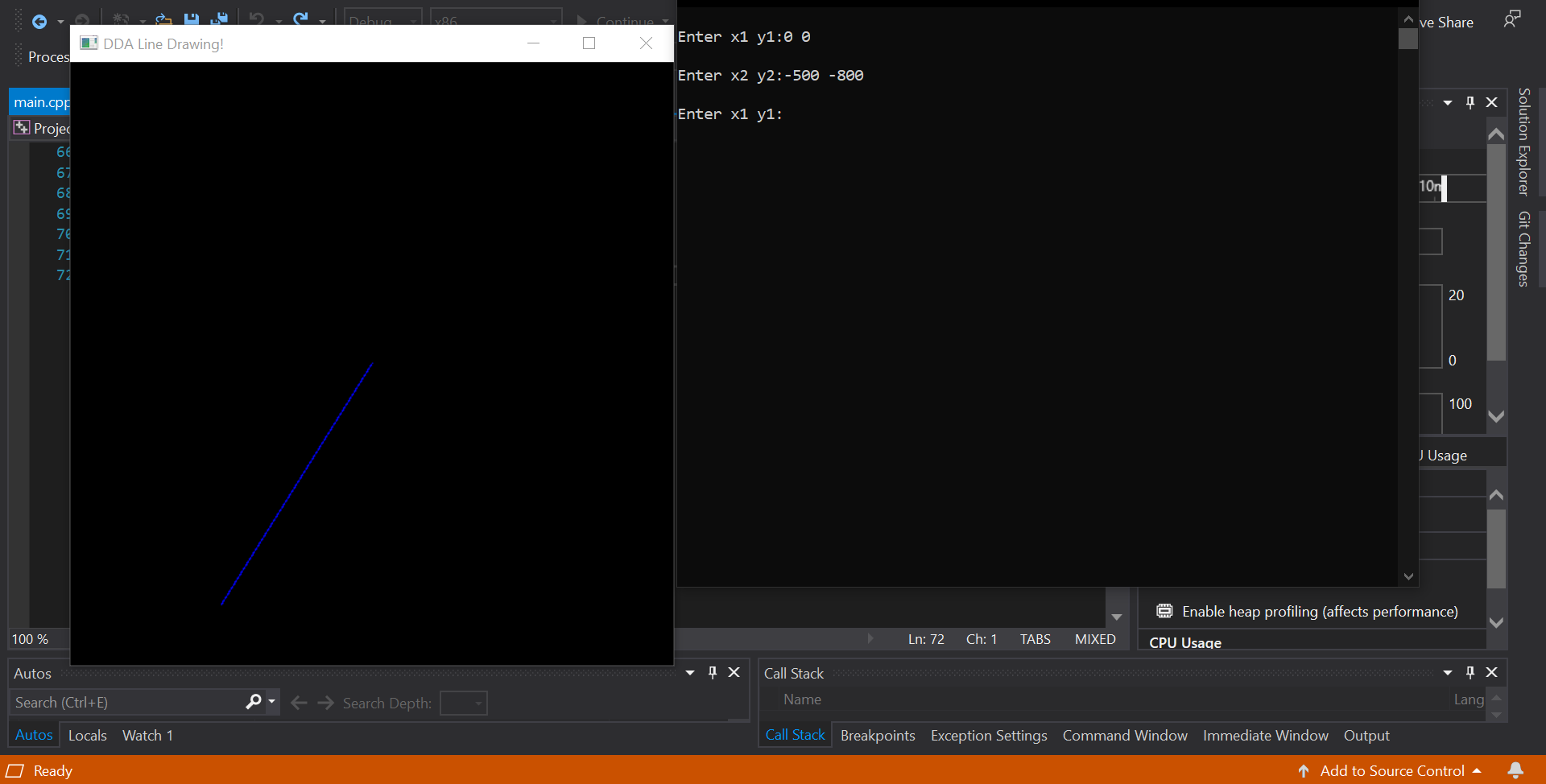


* **-ve slope: m<1**

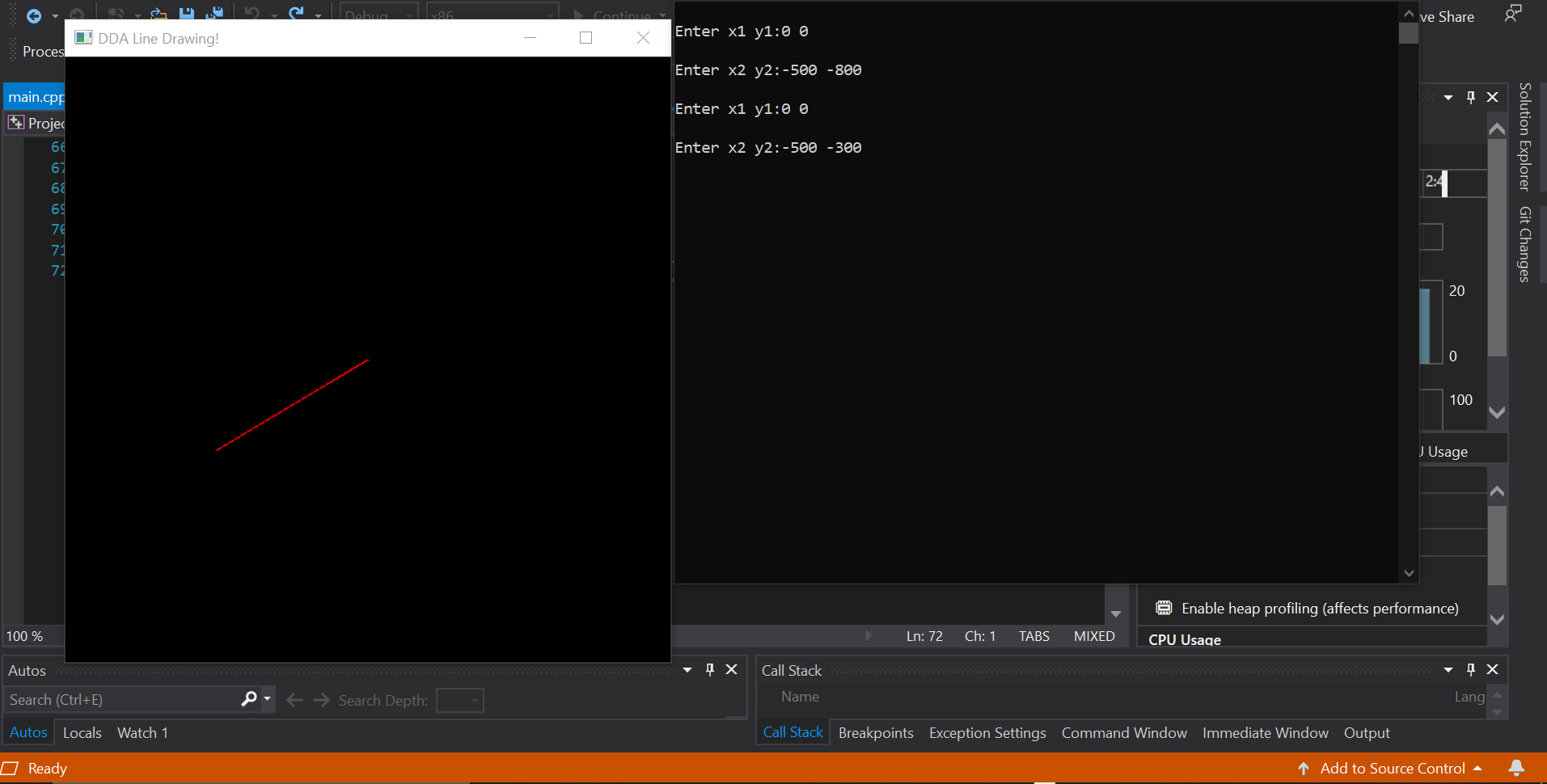


**Right to Left:**

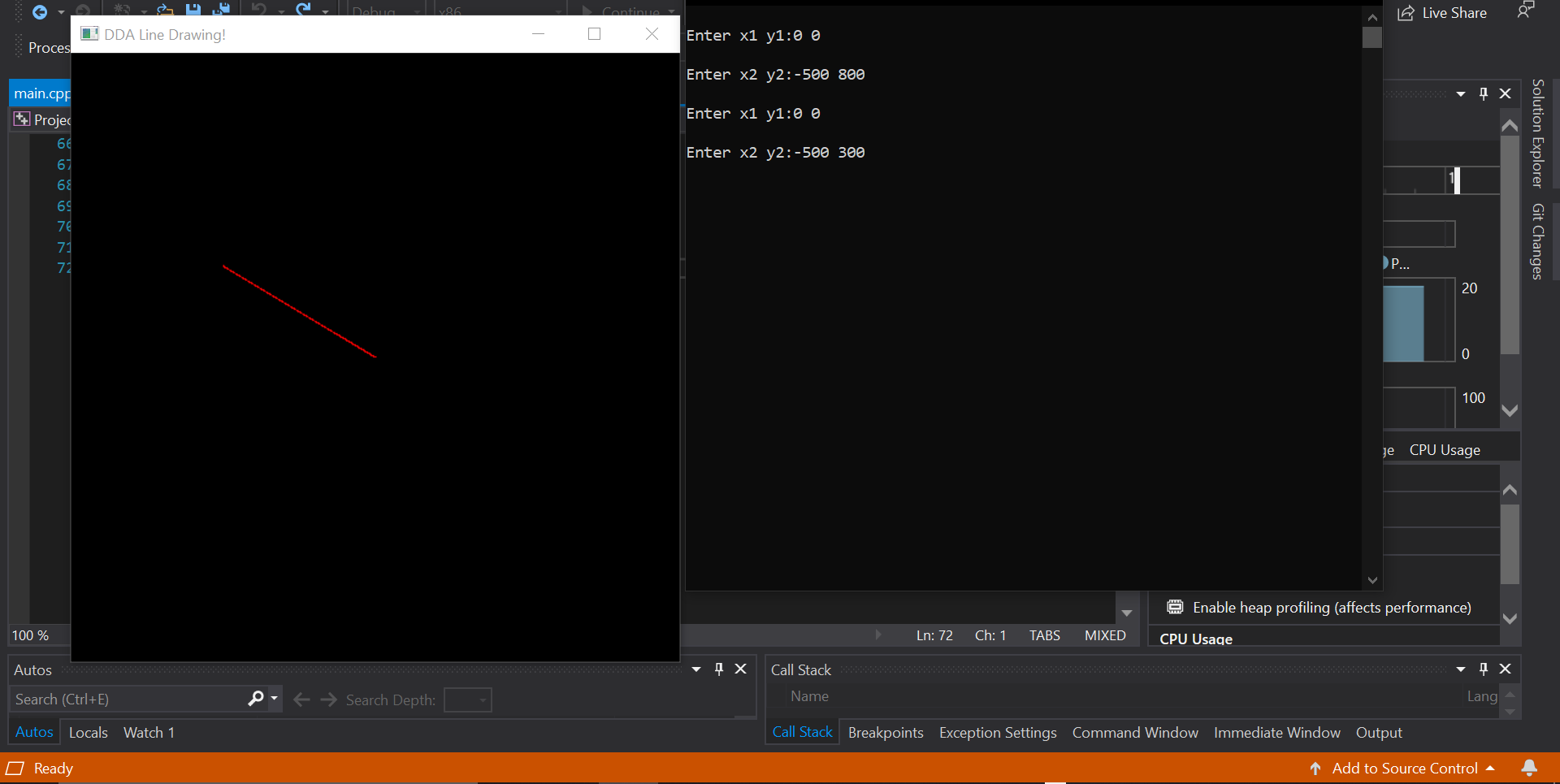
* **+ve slope: m>1**



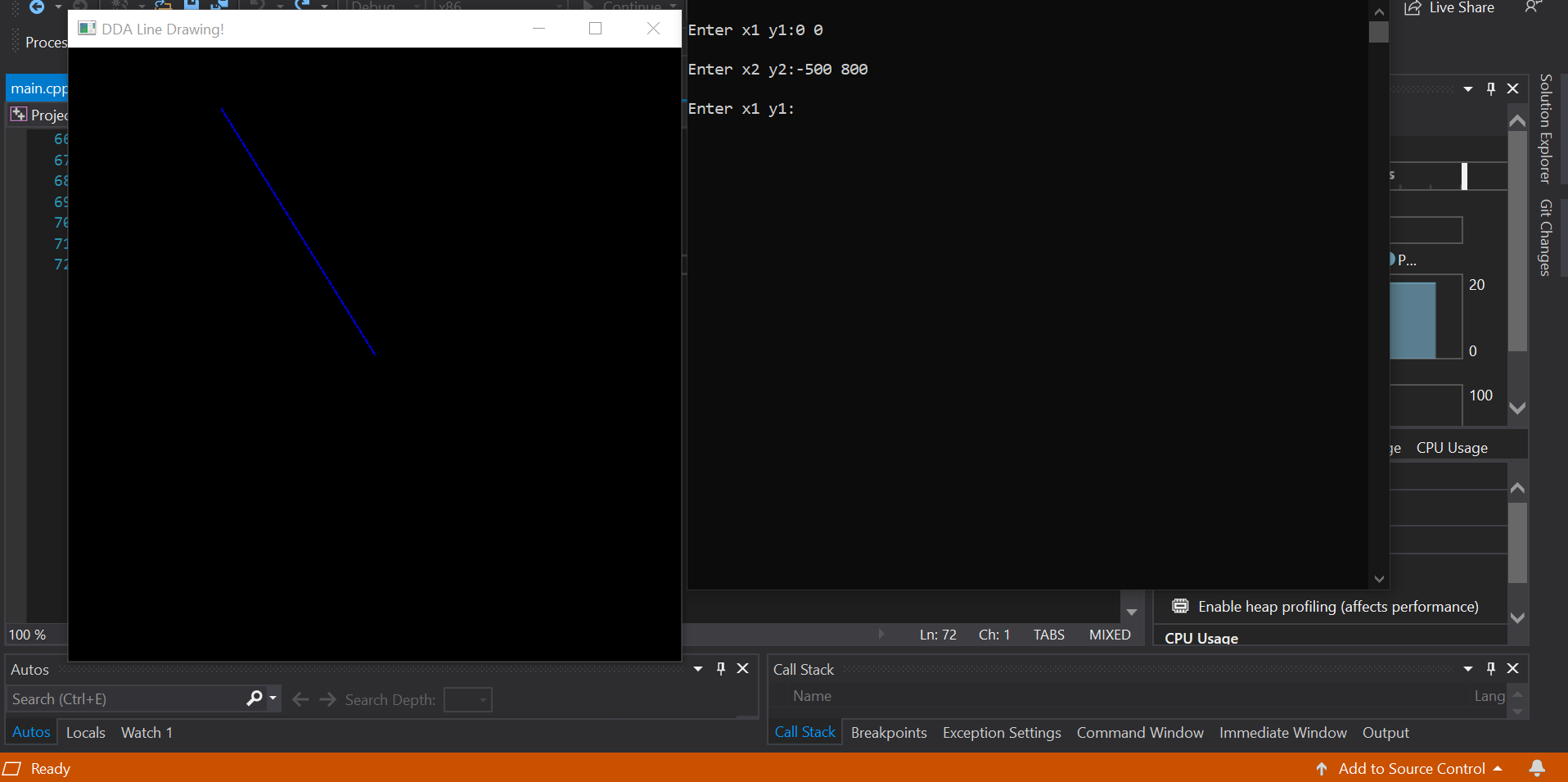
* **+ve slope: m<1**



* -**ve slope: m>1**



* **-ve slope: m<1**



**Practice question**

**2b)**

**AIM:**

To Replicate the pattern

**CODE:**

#include<stdlib.h>

#include<gl/glut.h>

#include<math.h>

#include<stdio.h>

#include<time.h>

void myInit() {

glClearColor(1.0, 1.0, 2.0, 1.0);

glColor3f(0.23, 0.86, 0.24);

glMatrixMode(GL\_PROJECTION);

glLoadIdentity();

glPointSize(1);

gluOrtho2D(0.0, 500.0, 0.0, 500.0);

}

int round(float num) {

return floor(num + 0.5);

}

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

float x = x1, y = y1;

int dx = x2 - x1;

int dy = y2 - y1;

int steps = abs(dx) > abs(dy) ? abs(dx) : abs(dy);

float xi = dx / (float)steps;

float yi = dy / (float)steps;

glBegin(GL\_POINTS);

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

glVertex2d(round(x), round(y));

x += xi;

y += yi;

}

glEnd();

}

void myDisplay() {

glClear(GL\_COLOR\_BUFFER\_BIT);

srand((unsigned)time(NULL));

int x1, y1, x2, y2, dis;

for (int i = 0; i < 200; ++i) {

x1 = abs(rand() % 500 + 1);

y1 = abs(rand() % 500 + 1);

dis = abs(rand() % 130 + 10);

x2 = int(sqrt((dis \* dis) / 2)) + x1;

y2 = int(sqrt((dis \* dis) / 2)) + y1;

lineDDA(x1, y1, x2, y2);

}

glFlush();

}

int main(int argc, char\* argv[]) {

glutInit(&argc, argv);

glutInitDisplayMode(GLUT\_SINGLE | GLUT\_RGB);

glutInitWindowSize(500, 500);

glutCreateWindow("Line Exercise");

glutDisplayFunc(myDisplay);

myInit();

glutMainLoop();

return 1;

**OUTPUT:**

****

**Result:**

The DDA Line Algorithm is compiled and got the output successfully.