Practical 2: -

Code:

#include<iostream>

#include<GL/glut.h>

using namespace std;

int Algo,type;

void Init()

{

glClearColor(0,0,0,0);

glColor3f(0,1,0);

gluOrtho2D(0,640,0,480);

glClear(GL\_COLOR\_BUFFER\_BIT);

}

int sign(float a){

if(a==0){

return 0;

}

if(a>0){

return 1;

}

return -1;

}

void B\_Line(int x\_1,int y\_1,int x\_2,int y\_2,int t){

float dy, dx, m , P;

dy = y\_2 - y\_1;

dx = x\_2 - x\_1;

m = dy/dx;

P = 2\*dy - dx;

int x = x\_1, y = y\_1;

cout<<"\n x1 = "<<x<<" y1 = "<<y;

if(m<1){

int cnt=1;

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

if(t == 1){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

if(t == 2){

if(i%2==0){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

}

if(t == 3){

if(cnt <= 10){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

cnt++;

if(cnt == 15){

cnt =1;

}

}

if(P<0){

x = x +1;

y =y;

P = P + 2\*dy;

}

else{

x= x+1;

y = y+1;

P = P + 2\*dy - 2\*dx;

}

}

}

else{

int cnt = 1;

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

if(t == 1){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

if(t == 2){

if(i%2==0){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

}

if(t == 3){

if(cnt <= 10){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

cnt++;

if(cnt == 15){

cnt =1;

}

}

if(P<0){

x = x;

y =y+1;

P = P + 2\*dx;

}

else{

x= x+1;

y = y+1;

P = P + 2\*dx - 2\*dy;

}

}

}

cout<<"\n xlast = "<<x<<" ylast = "<<y;

glFlush();

}

void DDA\_LINE(int x\_1,int y\_1,int x\_2,int y\_2, int t){

float dx,dy,length;

dx = x\_2-x\_1;

dy = y\_2-y\_1;

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

length = abs(dx);

}

else{

length = abs(dy);

}

float xin, yin;

xin = dx/length;

yin = dy/length;

float x,y;

x = x\_1 + 0.5 \* sign(xin);

y = y\_1 + 0.5 \* sign(yin);

int i=0;

int cnt =1;

while(i<=length){

if(t == 1){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

if(t == 2){

if(i%2==0){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

}

if(t == 3){

if(cnt <= 10){

glBegin(GL\_POINTS);

glVertex2i(x,y);

glEnd();

}

cnt++;

if(cnt == 15){

cnt =1;

}

}

x = x + xin;

y = y + yin;

i++ ;

}

glFlush();

}

void display()

{

DDA\_LINE(0,240,640,240,1);

B\_Line(320,0,320,640,1);

glFlush();

}

void mymouse(int b,int s, int x, int y)

{

static int x\_s,y\_s,x\_e,y\_e,pt=0;

if(b==GLUT\_LEFT\_BUTTON && s==GLUT\_DOWN)

{

if(pt==0)

{

x\_s =x;

y\_s =480 - y;

pt++;

glBegin(GL\_POINTS);

glVertex2i(x\_s,y\_s);

glEnd();

}

else

{

x\_e=x;

y\_e=480-y;

cout<<"\n x\_1\_click "<<x\_s<<" y\_1\_click "<<y\_s;

cout<<"\n x\_2\_click "<<x\_e<<" y\_2\_click "<<y\_e<<"\n";

glBegin(GL\_POINTS);

glVertex2i(x\_e,y\_e);

glEnd();

if(Algo == 1){

DDA\_LINE(x\_s,y\_s,x\_e,y\_e,type);

}

if(Algo == 2){

B\_Line(x\_s,y\_s,x\_e,y\_e,type);

}

}

}

else if(b==GLUT\_RIGHT\_BUTTON && s==GLUT\_DOWN)

{

pt=0;

}

glFlush();

}

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

{

cout<<"\n Select the Algorithm \n 1. DDA \n 2. Bresenham's \n";

cin>>Algo;

cout<<"Select the Line Type \n 1. Simple Line \n 2. Dotted Line\n 3. Dashed Line \n";

cin>>type;

if((Algo == 1 || Algo == 2 )&&(type==1 || type==2 || type==3)){

}

else{

cout<<"\n Option enter are wrong \n";

return 0;

}

glutInit(&argc,argv);

glutInitDisplayMode(GLUT\_SINGLE|GLUT\_RGB);

glutInitWindowPosition(100,100);

glutInitWindowSize(640,480);

glutCreateWindow("DDA-Line");

Init();

glutDisplayFunc(display);

glutMouseFunc(mymouse);

glutMainLoop();

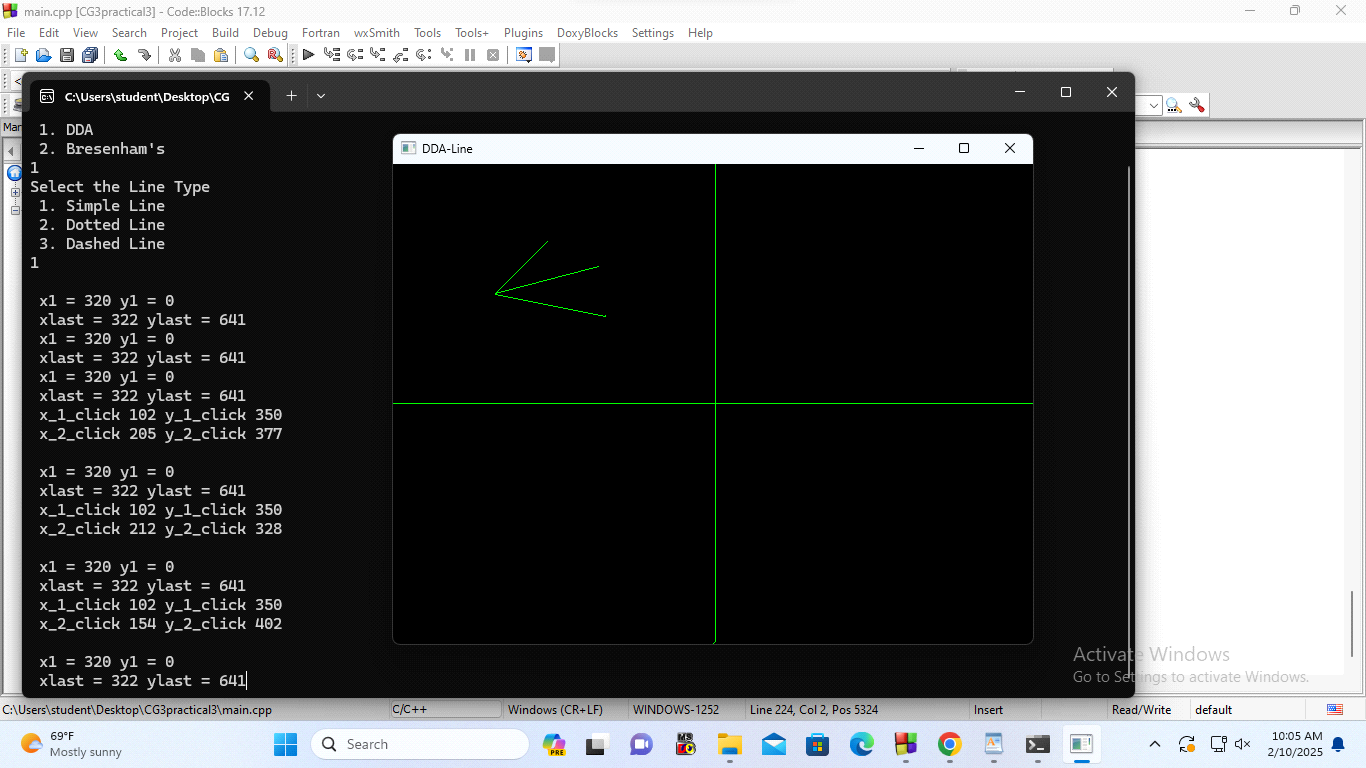
return 0;

}

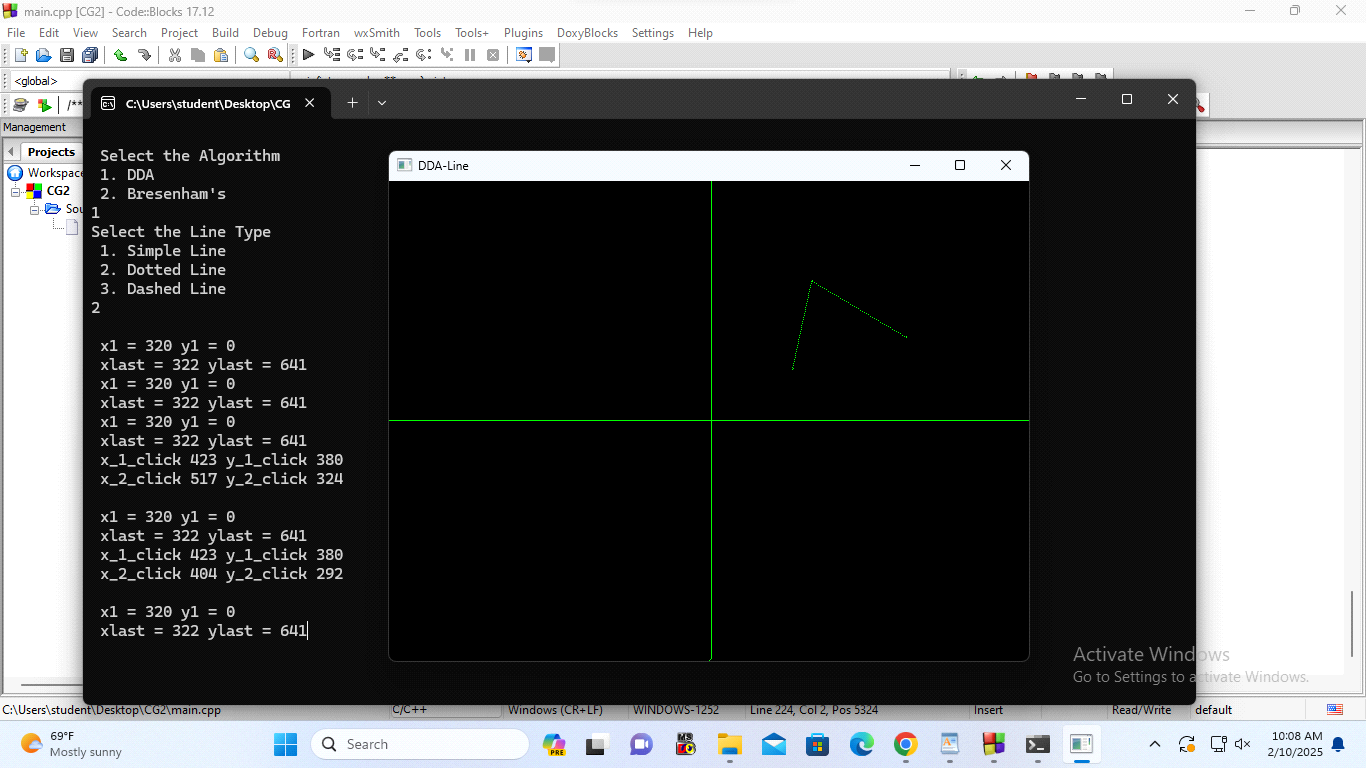
Output:

1.

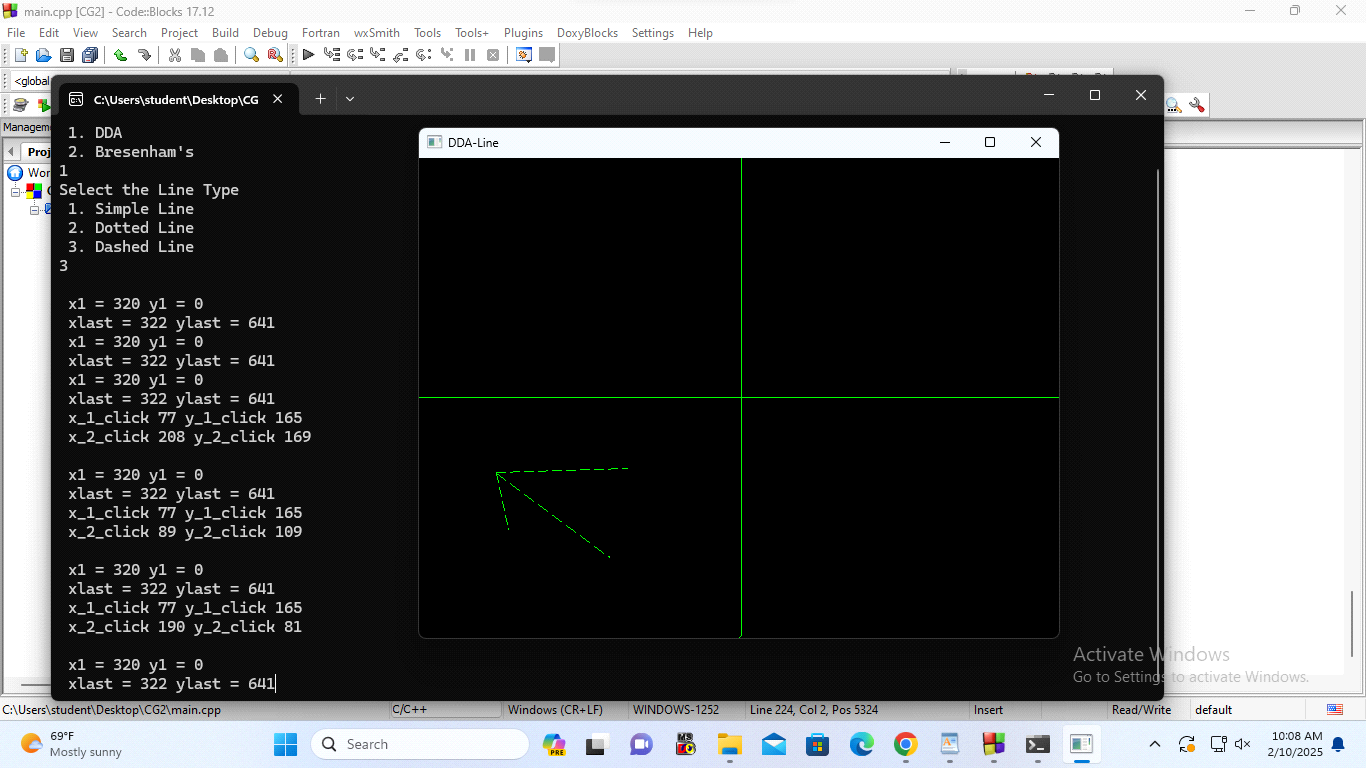
DDA (Simple Line):



DDA (Dotted Line):

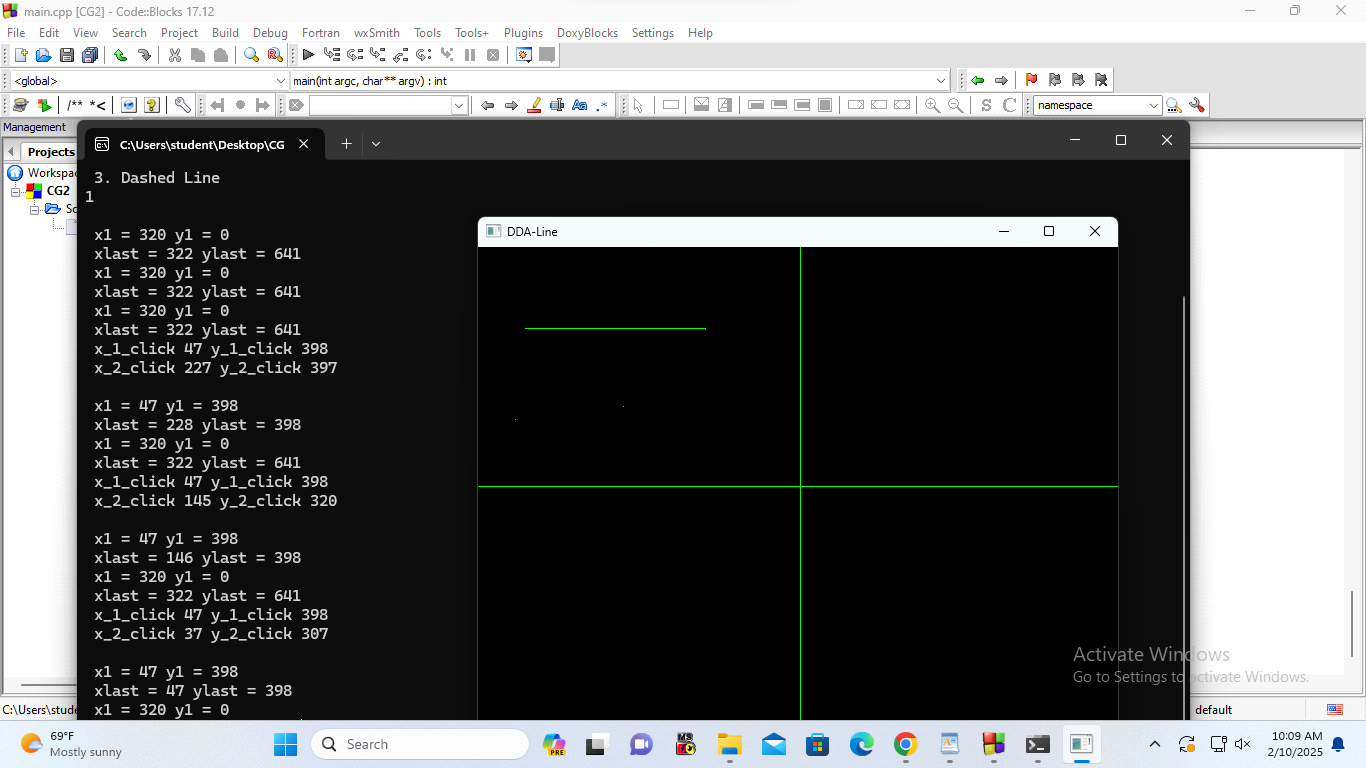


DDA (Dashed Line):

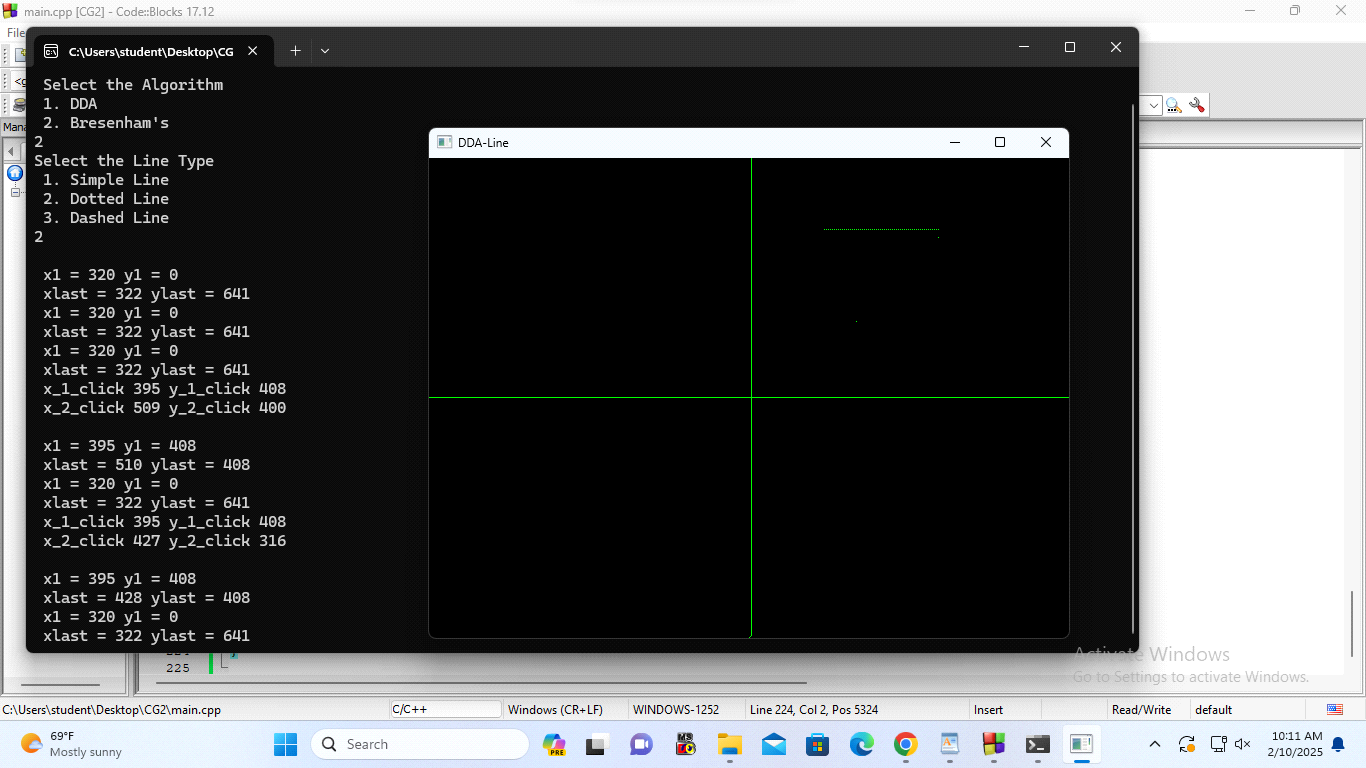


2.

Bresenham's (Simple Line):



Bresenham's (Dotted Line):



Bresenhams (Dashed Line):

