```
/*WAP to implement DDA algorithm in Cpp*/
//Digital Differential Analyzer Algorithm
#include <iostream>//line(x1,y1,x2,y2)
#include <cmath>
#include <graphics.h>
using namespace std;
float x_1,x_2,y_1,y_2,m;
void line_plot_m_s() // |m|<=1 , x(n+1)=x(n)+1 , y(n+1) = y(n)+m ,small slope
{
   x_1++;
   y_1=y_1+m;
   putpixel(x_1,(int)(y_1),GREEN);
}
void line_plot_m_l()// |m|>1 , y(n+1)=y(n)+1 , x(n+1) = x(n)+1/m , large slope
{
   y_1++;
   x_1=x_1+1/m;
   putpixel((int)(x_1),y_1,GREEN);
}
int main()
{
   int i;
   while(1)
   {
      cout<<"\n\n\n\t\t\t\t\t1366*768 ";
      cout<<"\n\n\n\t\t Enter line coordinates (x1,y1), (x2,y2) with in range (0,0)
to (1365,767)";
      cout<<"\n\n Enter (x1,y1)";
      cout<<"\n Enter x1: ";
      cin>>x_1;
      cout<<" Enter y1: ";
      cin>>y_1;
      cout<<"\n\n Enter (x2,y2)";
      cout<<"\n Enter x2: ";
      cin>>x_2;
      cout<<" Enter y2: ";
      cin>>y_2;
```

```
initwindow(1366,768);
for(i=0; i<=1365; i++) // creates white background
{
   line(0,i,1365,i);
}
//setcolor(GREEN);
//line(x_1+50,y_1+50,x_2+50,y_2+50);
m=(y_2-y_1)/(x_2-x_1);
putpixel(x_1,y_1,GREEN);
if (fabs(m)<=1)
{
   if (x_2<x_1)//swap
   {
      x_1=x_1+x_2;
      x_2=x_1-x_2;
      x_1=x_1-x_2;
      y_1=y_1+y_2;
      y_2=y_1-y_2;
      y_1=y_1-y_2;
   }
   while(x_1<x_2) //small slope |m|<=1
   {
      line_plot_m_s();
   }
}
else
{
   if (y_2<y_1)//swap
   {
      x_1=x_1+x_2;
      x_2=x_1-x_2;
      x_1=x_1-x_2;
      y_1=y_1+y_2;
      y_2=y_1-y_2;
      y_1=y_1-y_2;
   }
   while(y_1<y_2)//large slope  |m|>1
```

```
            {
               line_plot_m_l();
            }
         }
         getch();
         closegraph();
      }
      return 0;
}
/*
Test lines (x1,y1,x2,y2,slope)
(50,60,1200,600,0.46)
(1200,600,50,60,0.46)
(50,600,1200,70,-0.46)
(1200,70,50,600,-0.46)
(50,60,600,700,1.16)
(600,700,50,60,1.16)
(70,600,400,60,-1.63)
(400,60,70,600,-1.63)
(50,50,70,70,1)
(70,70,50,50,1)
(70,30,50,50,-1)
(50,50,70,30,-1)
(80,70,900,70,0)
(900,70,80,70,0)
(80,70,80,700,1/0)
(80,700,80,70,-1/0)
*/

/*WAP to implement DDA algorithm in Cpp*/
//Digital Differential Analyzer Algorithm
#include<GL/gl.h>
#include<GL/glu.h>
#include<GL/glut.h>
#include<iostream>
#include<math.h>
using namespace std;
```

```
void display();
void reshape(int,int);
void draw();
void takeData();
float X1,X2,Y1,Y2;
void init()
{
   glClearColor(0,0,0,1.0);
}
int main(int argc, char**argv)
{

   takeData();
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_RGB);

   glutInitWindowPosition(200,100);
   glutInitWindowSize(500,500);

   glutCreateWindow("DDA");
   glutDisplayFunc(display);
   glutReshapeFunc(reshape);
   init();

   glutMainLoop();

}

void display()
{
   glClear(GL_COLOR_BUFFER_BIT);
   glLoadIdentity();


//axis drawing
   glPointSize(1);
   glColor3f(1,1,1);
```

```
    glBegin(GL_LINES);
    glVertex2f(-250,0);
    glVertex2f(250,0);
    glVertex2f(0,-250);
    glVertex2f(0,250);

    glEnd();
    //draw
    glBegin(GL_POINTS);

    draw();

    glEnd();

    glFlush();

}

void reshape(int w, int h)
{

    glViewport(0,0,w,h);
    glMatrixMode(GL_PROJECTION);
    glLoadIdentity();

    gluOrtho2D(-250,250,-250,250);
    //gluOrtho2D(-125,125,-125,125);
    glMatrixMode(GL_MODELVIEW);

}


//function

void draw()
{
    glColor3f(1,1,1);
    float x1,x2,y1,y2,step,mx,my,dx,dy;
    x1=X1;
    x2=X2;
    y1=Y1;
    y2=Y2;
    dx=x2-x1;
    dy=y2-y1;
    if(abs(dy)<abs(dx))
       step=abs(dx);
    else
       step=abs(dy);

    mx=dx/step;
    my=dy/step;

    glVertex2f(x1,y1);
    for(int i=0; i<step; i++)
    {
       x1=x1+mx;
       y1=y1+my;

       glVertex2f(x1,y1);
    }

}


void takeData()
{

    cout<<"enter initial point: ";
    cin>>X1>>Y1;
    cout<<"enter final point: ";
    cin>>X2>>Y2;

}
```