/* **Experiment No. 2**

Problem Statement : Implement DDA and Bresenham line drawing algorithm to draw:

1.Center Line 2.Dotted Line. 3.Dashed Line 4.Simple Line

Roll No. : ITSA16

Name : KRUSHNA SUDHIR SHINDE

*/

A] DDA Line Drawing Algorithm Program Code :

```
#include<GL/gl.h>
#include<GL/glu.h>
#include<GL/glut.h>
#include<iostream>
using namespace std;

float x1,x2,Y1,y2;
int ch;
void init(){
   glClearColor(1,1,1,1);
   glColor3f(0.3,2.0,1.0);
   gluOrtho2D(-640,640,-480,480);
}
void Display(){
   float dy,dx,step,x,y,Xin,Yin;
   glClear(GL_COLOR_BUFFER_BIT);
   glPointSize(3);
   glLineWidth(2);

   dx=x2-x1;
   dy=y2-Y1;
   if(abs(dx)>abs(dy))
   {
      step=abs(dx);
   }
   else{
      step=abs(dy);
   }
   Xin=dx/step;
   Yin=dy/step;
   x=x1;
   y=Y1;
   glBegin(GL_POINTS);
   glVertex2i(x,y);
   glEnd);
```

```
switch(ch)
{
   int i;
   case 1:
      {
         for(i=1;i<=step;i++){ //dash line
            x=x+Xin;
            y=y+Yin;
            if(i%16<=8)
            {
               glBegin(GL_POINTS);
               glVertex2i(x,y);
               glEnd();
            }
         }
         break;
      }
   case 2:
      {
         for(i=1;i<=step;i++){ //dotted line
            x=x+Xin;
            y=y+Yin;
            if(i%8<=0)
            {
               glBegin(GL_POINTS);
               glVertex2i(x,y);
               glEnd();
            }
         }
         break;
      }
   case 3:
      {
          for(i=1;i<=step;i++){ //centreline
            x=x+Xin;
            y=y+Yin;

            int cycle = i % 35;
            if ((cycle < 10) || (cycle >= 20 && cycle < 25) || (cycle >= 30 && cycle < 40)) {
                  glBegin(GL_POINTS);
               glVertex2i(x,y);
               glEnd();
            }

         }
```

```
            break;
          }
           case 4:
            for(i=1;i<=step;i++){ // simple Line
                    x=x+Xin;
                    y=y+Yin;

                    glBegin(GL_POINTS);
                    glVertex2i(x,y);
                    glEnd();
            }
           break;

          default :
          cout<<"Wrong Choice !!!";
    }
glBegin(GL_LINES);
glVertex2i(-640,0);
glVertex2i(640,0);
glVertex2i(0,-480);
glVertex2i(0,480);
glEnd();
glFlush();
}
int main(int argc,char **argv)
{
   cout<<"Enter x1 and y1"<<endl;//Accept end point coordinates of line
   cin>>x1>>Y1;
   cout<<"Enter x2 and y2"<<endl;
   cin>>x2>>y2;

   cout<<"1.Dashed line\n2.Dotted line\n3.Center line\n4.Simple Line"<<endl;
   cout<<"Enter your choice:";
   cin>>ch;
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
   glutInitWindowPosition(0,0);
   glutInitWindowSize(640,480);

   glutCreateWindow("DDA");
   init();
   glutDisplayFunc(Display);
   glutMainLoop();
   return 0;
}
```

B] Bresenham’s Line Drawing Algorithm Program Code :

```
#include<GL/gl.h>
#include<GL/glu.h>
#include<GL/glut.h>
#include<iostream>
using namespace std;
float x1,x2,Y1,y2;
int ch;
int sign(int a){
   if(a>0){
      return 1;
   }else if(a<0){
      return -1;
   }else{
      return 0;
   }
}
void init(){
   glClearColor(1,1,1,1);
   glColor3f(0.3,2.0,1.0);
   gluOrtho2D(-640,640,-480,480);
}
void Display(){
   float dy,dx,step,x,y, G, s1, s2;
   glClear(GL_COLOR_BUFFER_BIT);
   glPointSize(3);
   glLineWidth(2);
   dx=abs(x2-x1);
   dy=abs(y2-Y1)
   if(dx > dy)
   {
      step=dx;
   }
   else{
      step=dy;
   }
   s1=sign(x2-x1);
   s2=sign(y2-Y1);
   G =(2*dy)-dx;
   x=x1;
   y=Y1;
   glBegin(GL_POINTS);
   glVertex2i(x,y);
   glEnd();
   switch(ch)
   {
int i;
 case 1:
```

```
            for(i=1;i<=step;i++){ //dash line
                while(G>= 0){
                    y=y+s2;
                    G=G-(2*dx);
                }
                x=x+s1;
                G=G+(2*dy);
                if(i%16<=8)
                {
                    glBegin(GL_POINTS);
                    glVertex2i(x,y);
                    glEnd();
                }
            }
            break;
        }
    case 2:
        {
            for(i=1;i<=step;i++){ //dotted line
                while(G>= 0){
                    y=y+s2;
                    G=G-(2*dx);
                }
                x=x+s1;
                G=G+(2*dy);
                if(i%8<=0)
                {
                    glBegin(GL_POINTS);
                    glVertex2i(x,y);
                    glEnd();
                }
            }
            break;
        }
    case 3:
        {
             for(i=1;i<=step;i++){ //centreline
                while(G>= 0){
                    y=y+s2;
                    G=G-(2*dx);
                }
                x=x+s1;
                G=G+(2*dy);
                int cycle = i % 35;
                if ((cycle < 10) || (cycle >= 20 && cycle < 25) || (cycle >= 30 && cycle < 40)) {
                        glBegin(GL_POINTS);
                    glVertex2i(x,y);
                    glEnd();
                }
              }
              Break;
           case 4:
           for(i=1;i<=step;i++){ // simple Line
                     while(G>= 0){
                      y=y+s2;
                     G=G-(2*dx);
                }
                     x=x+s1;
                     G=G+(2*dy);
                     glBegin(GL_POINTS);
                     glVertex2i(x,y);
                     glEnd();
            }
            break;
          default :
          cout<<"Wrong Choice !!!";
   }
glBegin(GL_LINES);
glVertex2i(-640,0);
glVertex2i(640,0);
glVertex2i(0,-480);
glVertex2i(0,480);
glEnd();
glFlush();
}
int main(int argc,char **argv)
{
   cout<<"Enter x1 and y1"<<endl;
   cin>>x1>>Y1;
   cout<<"Enter x2 and y2"<<endl;
   cin>>x2>>y2;
   cout<<"1.Dashed line\n2.Dotted line\n3.Center line\n4.Simple Line"<<endl;
   cout<<"Enter your choice:";
   cin>>ch;
   glutInit(&argc,argv);
   glutInitDisplayMode(GLUT_RGB | GLUT_SINGLE);
   glutInitWindowPosition(0,0);
   glutInitWindowSize(640,480);
   glutCreateWindow("Bresenham's");
   init();
   glutDisplayFunc(Display);
   glutMainLoop();
   return 0;  }
```