LAB -2

Implement DDA and Bresenham line drawing algorithm to draw: i) Simple Line ii) Dotted Line iii) Dashed Line iv) Solid line; using mouse interface Divide the screen in four quadrants with center as (0, 0). The line should work for all the slopes positive as well as negative.

Source Code :

```
#include <GL/glut.h>
#include <stdlib.h>
#include <stdio.h>

void displayPoint(int x, int y)
{
  glColor3f(0, 1, 0);
  glPointSize(5);
  glBegin(GL_POINTS);
  glVertex2i(x, y);
  glEnd();
}

float x01, x2, y01, y2;
int ch;

void SimpleLine(float x1, float y1, float x2, float y2)
{
  float step;

  float dx = x2 - x1;
  float dy = y2 - y1;

  if (abs(dx) > abs(dy))
  {
    step = abs(dx);
  }
  else
    step = abs(dy);

  float Xinc = dx / (float)step;
  float Yinc = dy / (float)step;
  float x = x1;
  float y = y1;

  for (int i = 0; i<= step; i++)
  {
    displayPoint(x, y);
    x = x + Xinc;
```

```
    y = y + Yinc;
  }
  glFlush();
}

void DottedLine(float x1, float y1, float x2, float y2)
{
  float step;

  float dx = x2 - x1;
  float dy = y2 - y1;

  if (abs(dx) > abs(dy))
  {
    step = abs(dx);
  }
  else
    step = abs(dy);

  float Xinc = dx / (float)step;
  float Yinc = dy / (float)step;
  float x = x1;
  float y = y1;
  displayPoint(x, y);

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

    x = x + Xinc;
    y = y + Yinc;
    if (i % 3 == 0)
    {
      displayPoint(x, y);
    }
  }
  glFlush();
}

void DashedLine(float x1, float y1, float x2, float y2)
{
  float step;

  float dx = x2 - x1;
  float dy = y2 - y1;

  if (abs(dx) > abs(dy))
  {
    step = abs(dx);
  }
  else
```

```
    step = abs(dy);

  float Xinc = dx / (float)step;
  float Yinc = dy / (float)step;
  float x = x1;
  float y = y1;
  displayPoint(x, y);

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

    x = x + Xinc;
    y = y + Yinc;
    if (i % 7 == 0)
    {
      displayPoint(x, y);
    }
  }
  glFlush();
}

void myMouse(int button, int state, int x, int y)
{
  static int xst, yst, pt = 0;
  if (button == GLUT_LEFT_BUTTON && state == GLUT_DOWN)
  {
    if (pt == 0)
    {
      xst = x;
      yst = y;
      x01 = xst;
      y01 = yst;
      pt = pt + 1;
    }
    else
    {
      x2 = x;
      y2 = y;
      if (ch == 1)
      {
        SimpleLine(xst, yst, x, y);
      }
      else if (ch == 2)
      {
        DottedLine(xst, yst, x, y);
      }
      else if (ch == 3)
      {
        DashedLine(xst, yst, x, y);
      }
```

```
        xst = x;
        yst = y;
      }
    }
    else if (button == GLUT_RIGHT_BUTTON && state == GLUT_DOWN)
      pt = 0;
    //Clear Screen
    glFlush();
}
void keyboard(unsigned char key, int x, int y)
{
    switch (key)
    {
    case 's':
    {
      ch = 1;
      glutMouseFunc(myMouse);
      break;
    }
    case 'd':
    {
      ch = 2;
      glutMouseFunc(myMouse);
      break;
    }
    case 'D':
    {
      ch = 3;
      glutMouseFunc(myMouse);
      break;
    }
    }
    glutPostRedisplay();
}
void initialize(void)
{
    glClearColor(1.0, 1.0, 1.0, 1.0);
    glClear(GL_COLOR_BUFFER_BIT);
    // gluOrtho2D(l,r,b,t)
    gluOrtho2D(0, 600, 600, 0);
}
void primitives(void)
{
    //glClearColor(1.0, 1.0, 1.0, 1.0);
    //glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(1, 0, 0);
    SimpleLine(0, 300, 600, 300);
    SimpleLine(300, 0, 300, 600);
    glutKeyboardFunc(keyboard);
}
```

```
int main(int argc, char **argv)
{
  glutInit(&argc, argv);
  glutInitDisplayMode(GLUT_SINGLE);
  glutInitWindowPosition(0, 0);
  glutInitWindowSize(600, 600);
  glutCreateWindow("OpenGL - DDA Algo");
  initialize();
  printf("--------------------");
  printf("\ns. Simple Line");
  printf("\nd. Dotted Line");
  printf("\nD. Dashed Line");
  printf("\n--------------------\n");
  glutDisplayFunc(primitives);
  glutMainLoop();
  return 0;
}
```

Output :