```cpp
#include <iostream>
#include <time.h>
#include <GL/glut.h>
#include <math.h>

void desenhaRetas();

int main(int argc, char *argv[])
{
    /* code */
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE);
    glutInitWindowSize(400, 400);
    glutInitWindowPosition(100, 100);
    glutCreateWindow("Retas");
    glutDisplayFunc(desenhaRetas);

    glutMainLoop();
    return 0;
}


void equacaoDaReta(int x1, int x2, int y1, int y2)
{
    int dx = x2 - x1;
    int dy = y2 - y1;
    float coeficienteAgunular = (float)dy/(float)dx;
    glVertex2i(x1, y1);
    for(float x = x1; x < x2; x++)
    {
        int dx2 = x - x1;
        float y = (coeficienteAgunular * (float)dx2) + (float)y1;
        glVertex2i(x, y);
    }
}

void DDA(int x1, int x2, int y1, int y2)
{
    float dx = x2 - x1;
    float dy = y2 - y1;
    float tamanho = abs(dx) > abs(dy) ? abs(dx) : abs(dy);
    float xAlt = dx/tamanho;
    float yAlt = dy/tamanho;
    glVertex2i(x1, y1);

    for(float k = 1; k < tamanho; k++)
    {
        x1 += xAlt;
        y1 += yAlt;
        glVertex2i(x1,y1);
    }

}


void Bresenham(int x1, int x2, int y1, int y2)
{
    int dx, dy, x, y, xfinal, p, const1, const2;
    dx = fabs(x1 - x2); dy = fabs(y1 - y2);
    p = 2 * dy - dx;
```

```
61      const1 = 2 * dy;
62      const2 = 2 * (dy - dx);
63      if(x1 > x2)
64      {
65          x = x2;
66          y = y2;
67          xfinal = x1;
68      }
69      else
70      {
71          x = x1;
72          y = y1;
73          xfinal = x2;
74      }
75      glVertex2i(x,y);
76      while(x < xfinal)
77      {
78          x++;
79          if(p < 0)
80          {
81              p += const1;
82          }
83          else
84          {
85              y++;
86              p += const2;
87          }
88          glVertex2i(x,y);
89      }
90
91  }
92
93  void desenhaRetas()
94  {
95    glClearColor(0,0,0,1);
96    glClear(GL_COLOR_BUFFER_BIT);
97    gluOrtho2D(0.0,50.0,0.0,50.0);
98      glPointSize(5.0f);
99
100     glBegin(GL_POINTS);
101
102             glColor3d(1, 0, 0);
103             equacaoDaReta(10, 20, 45, 35);
104
105             glColor3d(0, 1, 0);
106             DDA(30, 40, 45, 35);
107
108             glColor3d(0, 0, 1);
109             Bresenham(30, 20, 25, 15);
110
111     glEnd();
112
113     glFlush();
114 }
115
116
117
```