```
#include
<GL/glut.h>
              #include <iostream>
              #define zero 0.0
              #define one 1.0
              using namespace std;
              int a, b, c, d, type;
              void drawpixel(int x,int y,int type){
                  glColor3f(one,one,one);
                  glBegin(GL_POINTS);
                  glVertex2i(x,y);
                  glEnd();
              }
              void BresenhamLine(int x1, int y1, int x2, int y2, int type) {
                  int dx, dy, i, e;
                     int incx, incy;
                     int x,y;
                  glColor3f(one,one,one);
                     if(type==4){
                      glPointSize(10.0f);
                     }else{
                      glPointSize(1.0f);
                     }
                     glBegin(GL_POINTS);
                     glVertex2i(x, y);
                     dx = x2-x1;
                     dy = y2-y1;
                     if (dx < 0) dx = -dx;
                     if (dy < 0) dy = -dy;
                     incx = 1;
                     if (x2 < x1) incx = -1;
                     incy = 1;
                     if (y2 < y1) incy = -1;
                     x = x1; y = y1;
                     if (dx > dy) {
                            glVertex2i(x, y);
                            e = 2 * dy-dx;
                            int j=0;
                            for (i=0; i<dx; i++) {
                                   if (e >= 0) {
                                          y += incy;
                                          e += 2*(dy-dx);
                                   }
```

```
                    else
                           e += 2*dy;
                    x += incx;
                     if (type == 4 || type == 1) {
          glVertex2i((int)x, (int)y);
          }
          if (j % 2 == 0 && type == 2) {
              glVertex2i((int)x, (int)y);
          }
          if (j < 5 && type == 3) {
              glVertex2i((int)x, (int)y);
          }
          j = (j + 1) % 10;
             }
     } else {
             e = 2*dx-dy;
             int j=0;
             for (i=0; i<dy; i++) {
                    if (e >= 0) {
                           x += incx;
                           e += 2*(dx-dy);;
                    }
                    else
                           e += 2*dx;
                    y += incy;
                     if (type == 4 || type == 1) {
          glVertex2i((int)x, (int)y);
          }
          if (j % 2 == 0 && type == 2) {
              glVertex2i((int)x, (int)y);
          }
          if (j < 5 && type == 3) {
              glVertex2i((int)x, (int)y);
          }
          j = (j + 1) % 10;
             }
     }
     glEnd();
}
void display() {
    glClear(GL_COLOR_BUFFER_BIT);
    glColor3f(zero, zero, zero);
    BresenhamLine(-350, 0, 350, 0, 1);
    BresenhamLine(0, 350, 0, -350, 1);
    glFlush();
}
void init() {
    glClearColor(zero, zero, zero, zero);
    gluOrtho2D(-350, 350, -350, 350);
}
int oldx,oldy,newx,newy,cnt=0;
void mouse(int button,int status,int x,int y){
    if(status==GLUT_DOWN && button==GLUT_LEFT_BUTTON){
        int viewport[4];
      glGetIntegerv(GL_VIEWPORT, viewport);
      int winWidth = viewport[2];
      int winHeight = viewport[3];
      int xi = x- winWidth / 2;
      int yi = winHeight/2-y;
      cout << xi << "\t" << yi << "\n";
      cnt = (cnt + 1) % 2;
      if (cnt == 1)
      {
        oldx = xi;
        oldy = yi;
        cout << "a" << endl;
      }
      if (cnt == 0)
      {
        newx = xi;
        newy = yi;
        cout << "b" << endl;
      }
      glPointSize(5.0f);
      glColor3f(1.0, 0.0, 0.0);
      glBegin(GL_POINTS);
      glVertex2i(xi, yi);
      glEnd();
      glFlush();
    }
}
void menu(int x){
    BresenhamLine(oldx,oldy,newx,newy,x);
}
int main(int argc, char** argv) {
    glutInit(&argc, argv);
    glutInitDisplayMode(GLUT_SINGLE | GLUT_RGB);
    glutInitWindowSize(700, 700);
    glutInitWindowPosition(50, 50);
    glutCreateWindow("BRESENHAM LINE DRAWING: ");
    glutCreateMenu(menu);
    init();
    glutMouseFunc(mouse);
    glutAddMenuEntry("SIMPLE LINE ", 1);
    glutAddMenuEntry("DOTTED LINE ", 2);
    glutAddMenuEntry("DASHED LINE ", 3);
    glutAddMenuEntry("SOLID LINE ",4);
    glutDisplayFunc(display);
    glutAttachMenu(GLUT_RIGHT_BUTTON);
    glutMainLoop();
    return 0;
}
```