```
/*A program to allow the user to draw rubber rectangles: those
that grow and shrink as the user moves the mouse.*/
//all suitable includes
struct GLintPoint {
      GLint x, y;
};
//global variables
GLintPoint   corner[2];
bool    selected = false;
int    screenWidth = 640, screenHeight = 480;
void myDisplay() {
  glClear( GL_COLOR_BUFFER_BIT );
  glMatrixMode( GL_MODELVIEW );
  glLoadIdentity();
  glColor3f( 1.0f, 1.0f, 1.0f );
  if( selected ) {
    glBegin( GL_QUADS );
    glVertex2i( corner[0].x, corner[0].y ); //draw a rectangle
    glVertex2i( corner[0].x, corner[1].y );
    glVertex2i( corner[1].x, corner[1].y );
    glVertex2i( corner[1].x, corner[0].y );
  glEnd();

}
glutSwapBuffers();

}
void myMouse( int button, int state, int x, int y ) {
  if( button == GLUT_LEFT_BUTTON && state == GLUT_DOWN ) {
    corner[0].x = x;
    corner[0].y = screenHeight - y;
    selected = true;
}
glutPostRedisplay();
}
void myPassiveMotion( int x, int y )  {
  corner[1].x = x;
  corner[1].y = screenHeight - y;
  glutPostRedisplay();
}
int main( int argc, char ** argv ) {
glutInit( &argc, argv );
// initialize window
glutInitWindowSize( screenWidth, screenHeight );
glutInitWindowPosition( 0, 0 );
glutInitDisplayMode( GLUT_RGB | GLUT_DOUBLE );
// create window
glutCreateWindow( "Rubber Rect Demo" );
// set the projection matrix
glMatrixMode( GL_PROJECTION );
glLoadIdentity();
gluOrtho2D( 0, screenWidth, 0, screenHeight );
glMatrixMode( GL_MODELVIEW );
// clear rendering surface
glClearColor(0.0f, 0.0f, 0.0f, 0.0f);  // background is black
glViewport(0, 0, screenWidth, screenHeight);
```