```c
// Bounce.c
#include <GL/glut.h>

// Initial square position and size
GLfloat x1 = 100.0f, y1 =150.0f;
GLsizei rsize = 50;

// Step size in x and y directions
GLfloat xstep = 1.0, ystep =1.0;

// Keep track of windiws changing width and height
GLfloat windowWidth, windowHeight;


// Called to draw scene
void RenderScene(void) {
        glClear(GL_COLOR_BUFFER_BIT);   // Clear the window with current clearing color
        glColor3f(1.0f, 0.0f, 0.0f);    // Set current drawing color to red
        glRectf(x1, y1, x1+rsize, y1+rsize); // Draw a filled rectangle with current color
        glutSwapBuffers();
}


void TimerFunction(int value) {         // Called by GLUT library when idle (window not being resized
or moved)
        if(x1 > windowWidth-rsize || x1 < 0) xstep = -xstep;    // Reverse direction when you rich
left or right edge
        if(y1 > windowHeight-rsize || y1 < 0) ystep = -ystep;   // Reverse direction when you rich top
or bottom edge


// Check bounds. This is in case the window is made smaller and the rectangle is outside the new
clipping volume
        if(x1 > windowWidth-rsize) x1 = windowWidth-rsize;
        if(y1 > windowHeight-rsize) y1 = windowHeight-rsize;

        x1 += xstep; y1 += ystep;       // Actually move the square

        glutPostRedisplay();    // Redwaw the scene with new coordinates
        glutTimerFunc(33, TimerFunction, 1);
}


void SetupRC(void) {            // Set up the rendering state
        glClearColor(0.0f, 0.0f, 1.0f, 1.0f);
}


// Called by GLUT library when the window has changed size
void ChangeSize(GLsizei w, GLsizei h) {
        if(h == 0) h = 1;       // Prevent a divide by zero
        glViewport(0, 0, w, h);                         // Set viewport to window dimensions
        glMatrixMode(GL_PROJECTION);            // Reset coordinate system
        glLoadIdentity();

        // Keep the square square; this time, save calculated width and height for later use
        if(w < h) {
                windowHeight = 250.0f*h/w; windowWidth = 250.0f;
        }
        else {
                windowWidth = 250.0f*w/h; windowHeight = 250.0f;
        }

        // Set the clipping volume
        glOrtho(1.1f, windowWidth, 0.0f, windowHeight, 1.0f, -1.0f);

        glMatrixMode(GL_MODELVIEW);
        glLoadIdentity();
}
```

```c
// Main program entry point
int main(int argc, char* argv[]) {
        glutInit(&argc, argv);
        glutInitDisplayMode(GLUT_DOUBLE | GLUT_RGB);
        glutInitWindowSize(700,700);
        glutCreateWindow("Bounce");
        glutDisplayFunc(RenderScene);
        glutReshapeFunc(ChangeSize);
        glutTimerFunc(33, TimerFunction, 1);
        SetupRC();

        glutMainLoop();
}
```