Write a shader program with the source codes of the vertex shader and the fragment shader hard-coded as strings in your OpenGL application (so no need to read any shader file). Your program displays a red pentagon.

Source Code:
```
int readVertexShader( GLchar **shader )
{
    // Allocate memory to hold the source of our shaders.
    char str[] = "
        attribute vec3 temp;
        varying vec3 color;
        void main(void){
            color = temp;
            gl_Position = gl_ModelViewProjectionMatrix*gl_Vertex;
        }
    ";
    int len = strlen ( str );
    *shader = (GLchar *) malloc( len + 1);
    strcpy ( *shader, str );
    (*shader)[len] = '\0';
    return 1;
}

int readFragmentShader( GLchar **shader )
{
    // Allocate memory to hold the source of our shaders.
    char str[] = "
        varying vec3 color;
        void main(void){
            gl_FragColor = vec4( color, 1);
        }
    ";
    int len = strlen ( str );
    *shader = (GLchar *) malloc( len + 1);
    strcpy ( *shader, str );
    (*shader)[len] = '\0';
    return 1;
}
```
return 1;
}

int init(void)
{

    const char *version;
    GLchar *VertexShaderSource, *FragmentShaderSource;
    int loadstatus = 0;

    version = (const char *) glGetString(GL_VERSION);
    if (version[0] != '2' || version[1] != '.') {
        printf("This program requires OpenGL 2.x, found %s\n", version);
        exit(1);
    }
    readVertexShader(&VertexShaderSource);
    readFragmentShader(&FragmentShaderSource);
    loadstatus = installShaders(VertexShaderSource, FragmentShaderSource);

    return loadstatus;
}

void display(void)
{
    GLfloat vec[4];

    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);
    glClearColor( 1.0, 1.0, 1.0, 0.0 ); //get white background color
    glColor3f ( 1, 0, 1 ); //purple if shader incorrect
    int loc;
    loc = glGetUniformLocation(programObject, "temp");

    glBegin ( GL_POLYGON );
    glVertex3f(loc,1,0,0);
    glVertex3f(-1, -2, 1);
    glVertex3f(loc,1,0,0);
    glVertex3f( 1, -2, 1);
    glVertex3f(loc,1,0,0);
    glVertex3f( 2, 0, 1);
    glVertex3f(loc,1,0,0);
    glVertex3f( 0, 2, 1);
    glVertex3f(loc,1,0,0);
    glVertex3f(-2, 0, 1);
    glEnd();
    glutSwapBuffers();
    glFlush();
}
Summary:

I have completed all parts of this assignment and I am going to give myself 20 points. For this lab, I added 2 new functions that take in the shader files as a string. Then within the init function, both shader read functions are called and the pentagon is made within the display function.