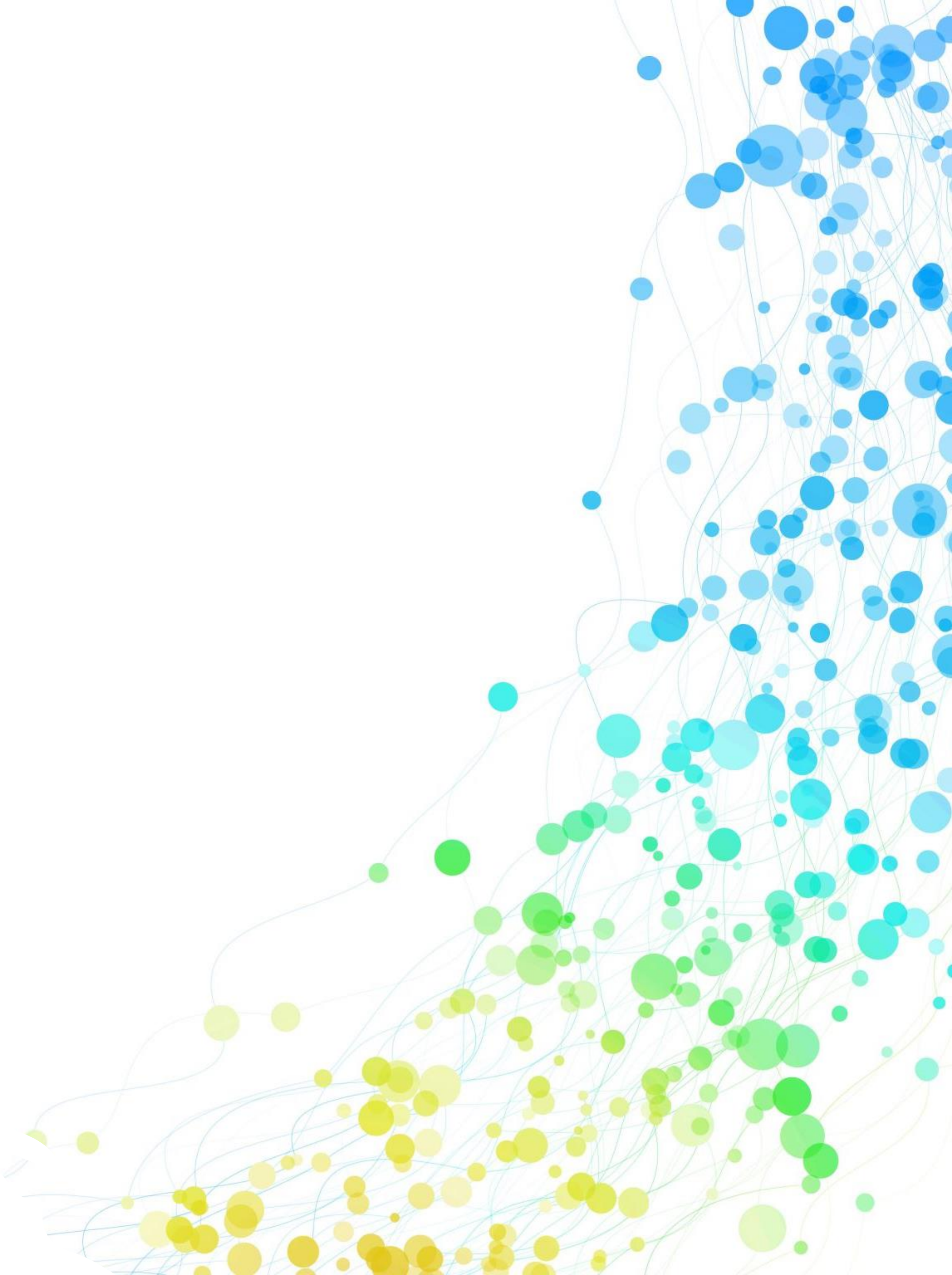
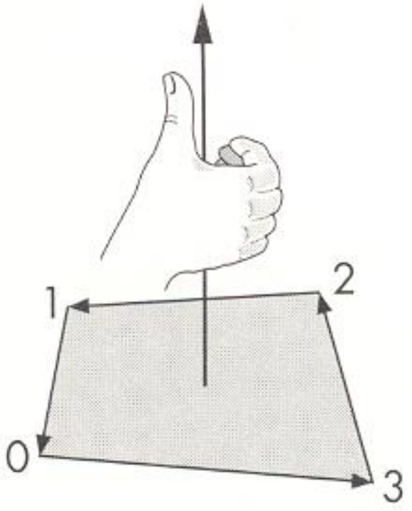
Geometric Objects and Transformations

6TH WEEK, 2022



Modeling a Cube

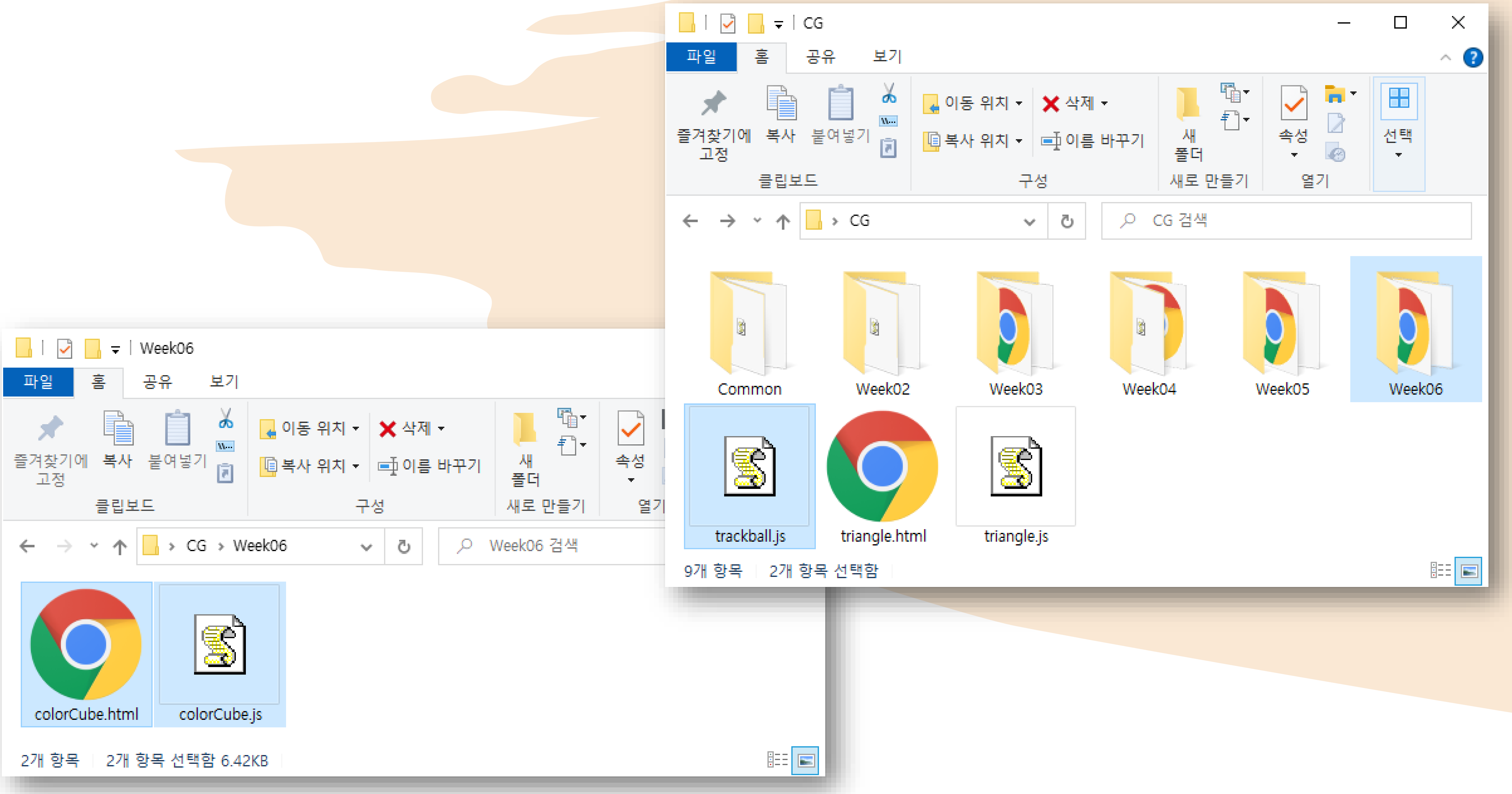


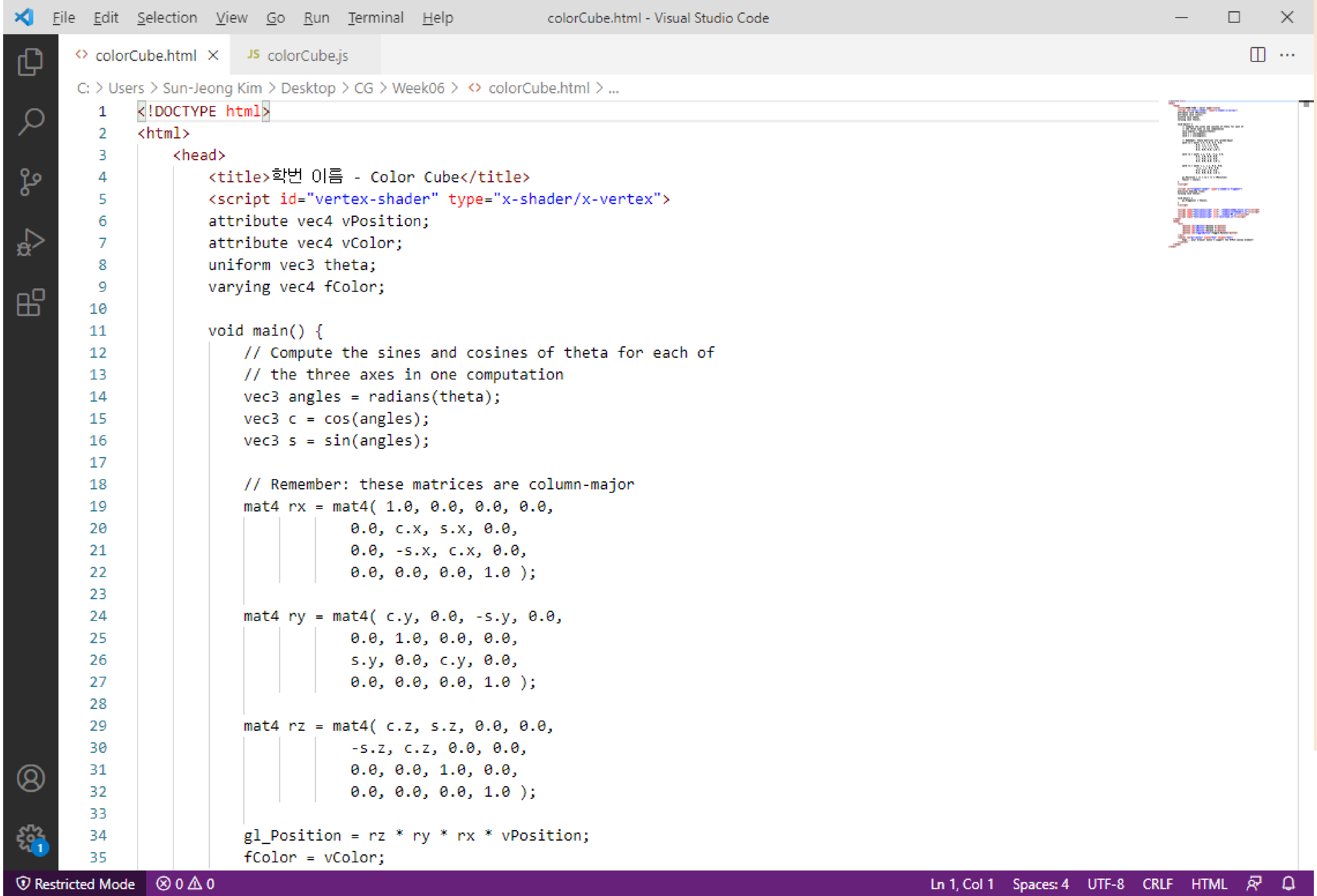
- Surface-based model
  - Outward-pointing face
    - Right-hand rule: counterclockwise order
- Data structure
  - Geometry: location of vertices

```
float vertices[8][3]= { {-1.0, -1.0, -1.0}, {1.0, -1.0, -1.0},
        {1.0, 1.0, -1.0}, {-1.0, 1.0, -1.0}, {-1.0, -1.0, 1.0},
        {1.0, -1.0, 1.0}, {1.0, 1.0, 1.0}, {-1.0, 1.0, 1.0}};
```

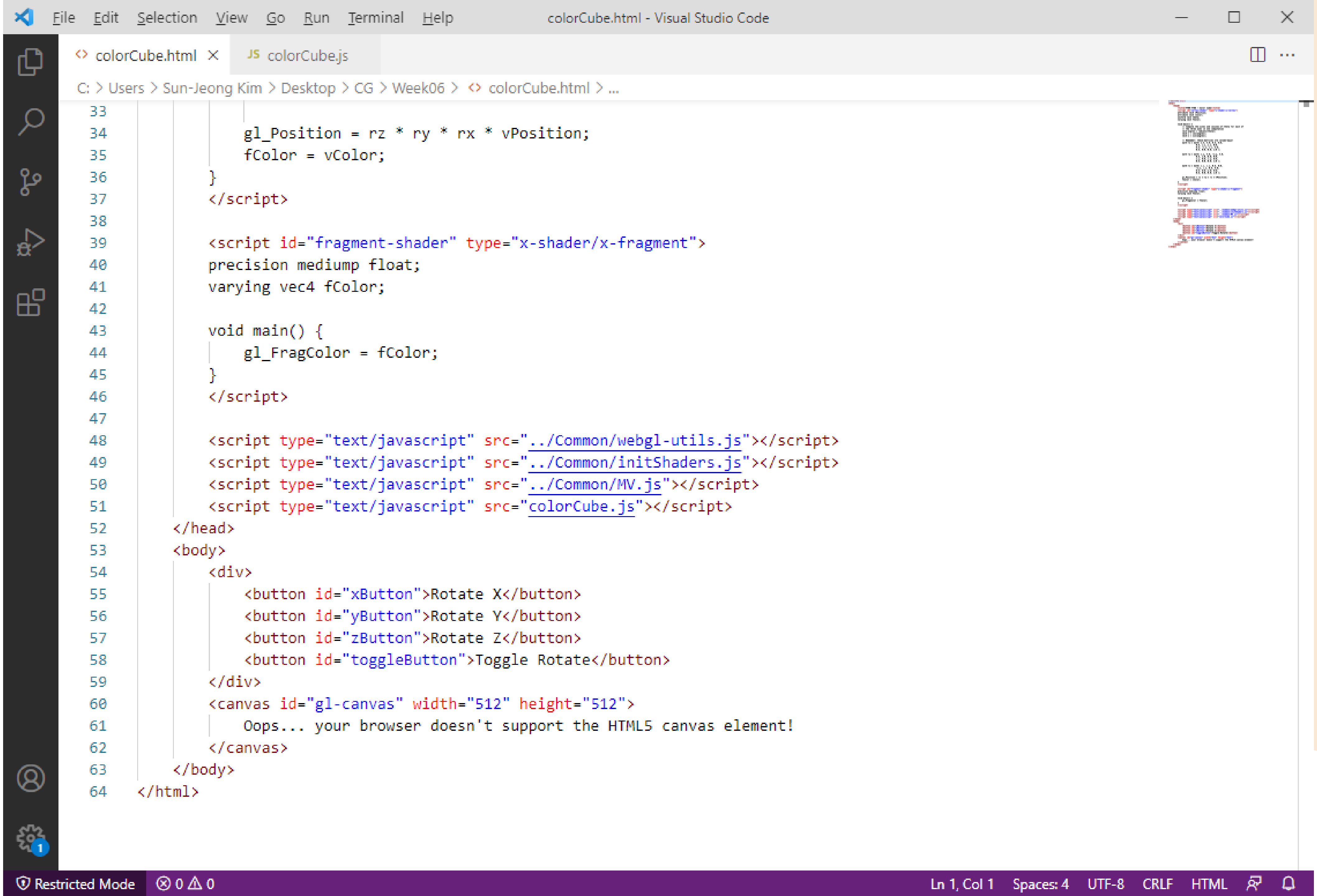
  - Topology: connectivity

```
var indices = [ 0, 3, 2, 0, 2, 1 ];
for ( var i = 0; i < indices.length; ++i )
        points.push( vertices[indices[i]] );

gl.drawArrays( gl.TRIANGLES, 0, NumVertices );
```

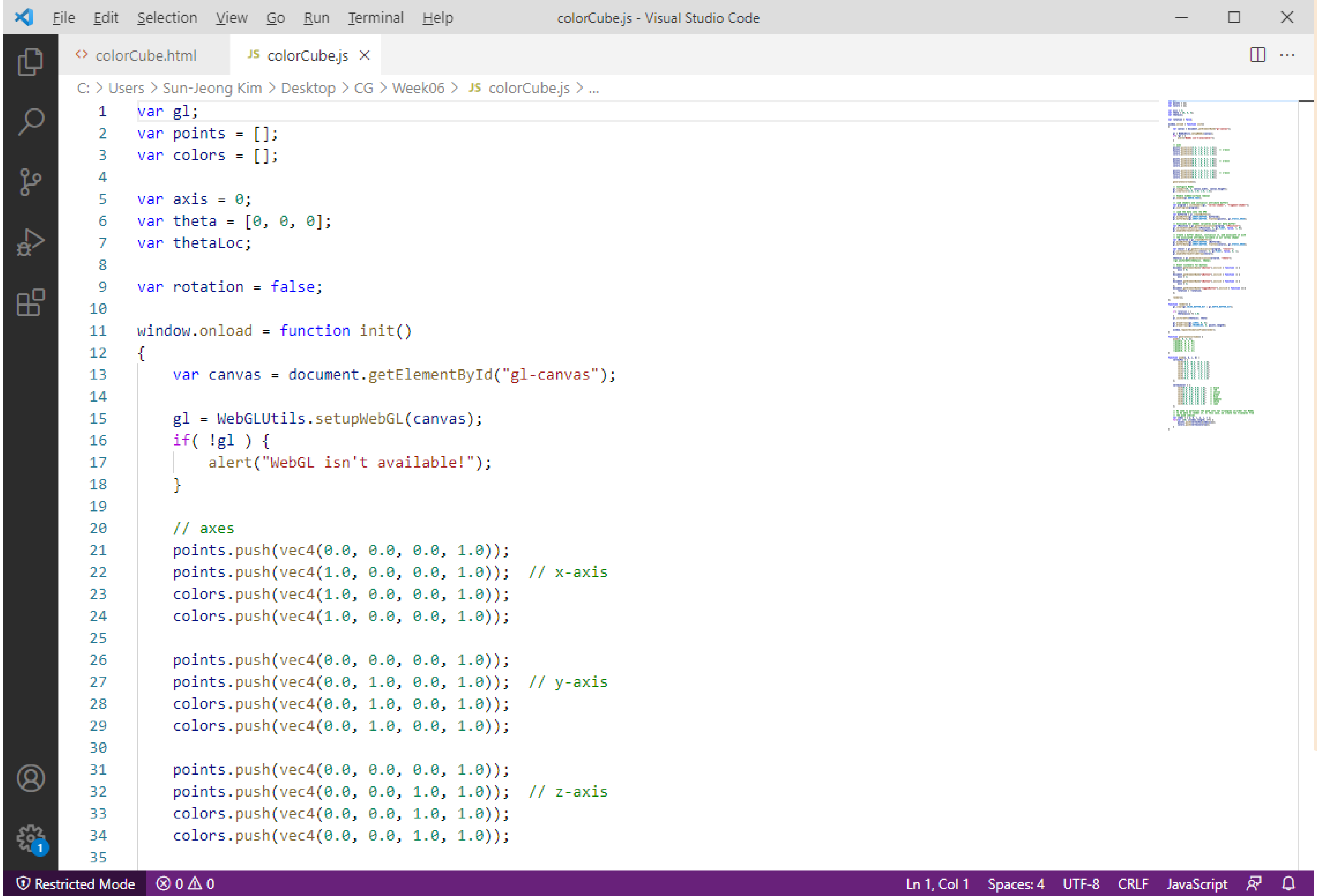

CG
파일
홈
공유
보기
즐겨찾기에 고정
복사
붙여넣기
클립보드
이동 위치
복사 위치
삭제
이름 바꾸기
구성
새 폴더
새로 만들기
속성
열기
선택
CG
CG 검색
Common
Week02
Week03
Week04
Week05
Week06
trackball.js
triangle.html
triangle.js
9개 항목
2개 항목 선택함
Week06
파일
홈
공유
보기
즐겨찾기에 고정
복사
붙여넣기
클립보드
이동 위치
복사 위치
삭제
이름 바꾸기
구성
새 폴더
새로 만들기
속성
열기
CG
Week06
Week06 검색
colorCube.html
colorCube.js
2개 항목
2개 항목 선택함 6.42KB



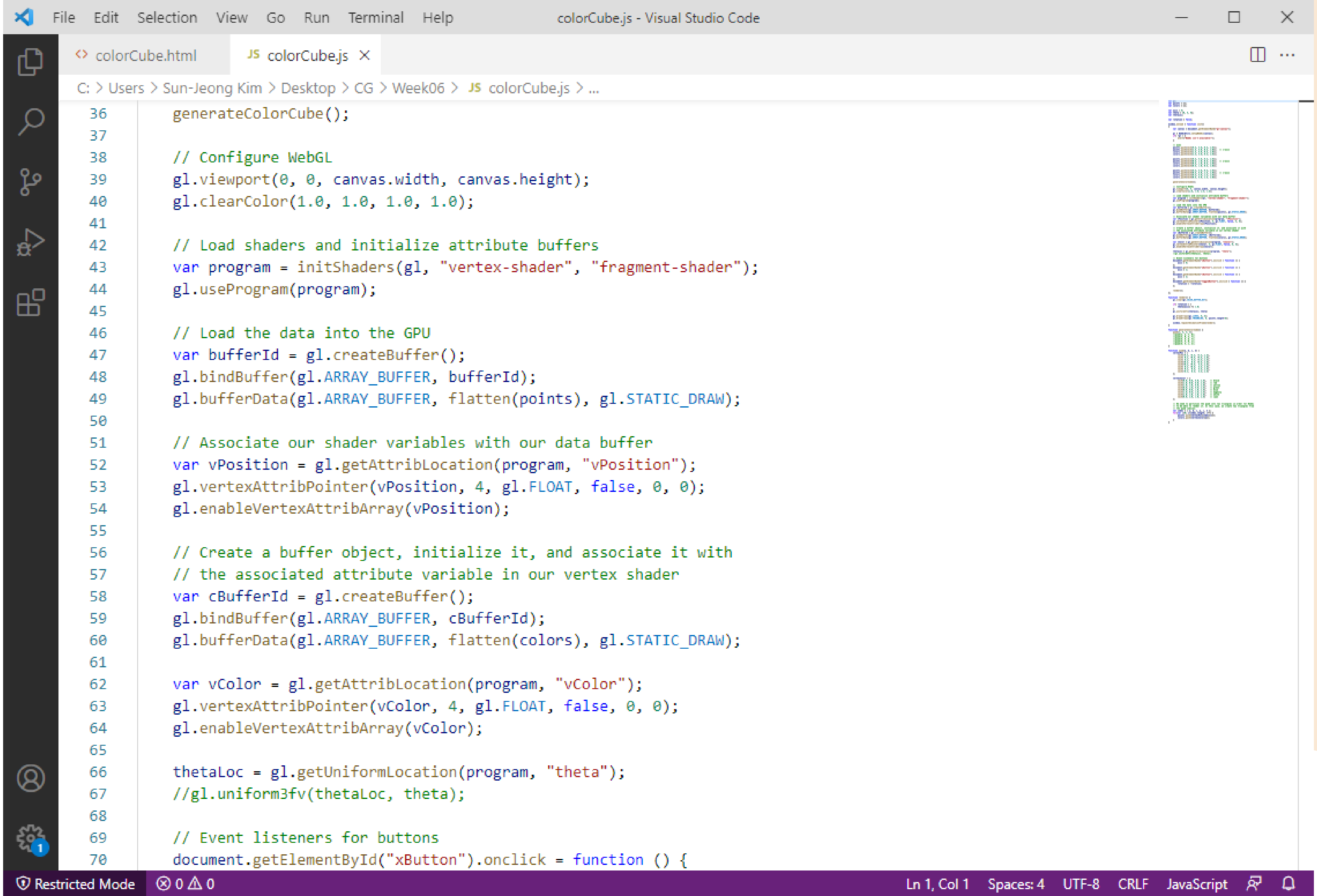
```
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Color Cube</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform vec3 theta;
        varying vec4 fColor;

        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, c.x, s.x, 0.0,
                        0.0, -s.x, c.x, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        s.y, 0.0, c.y, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                        -s.z, c.z, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            gl_Position = rz * ry * rx * vPosition;
            fColor = vColor;
```


File Edit Selection View Go Run Terminal Help
colorCube.html - Visual Studio Code
colorCube.html
colorCube.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > colorCube.html > ...
Restricted Mode
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
HTML
1

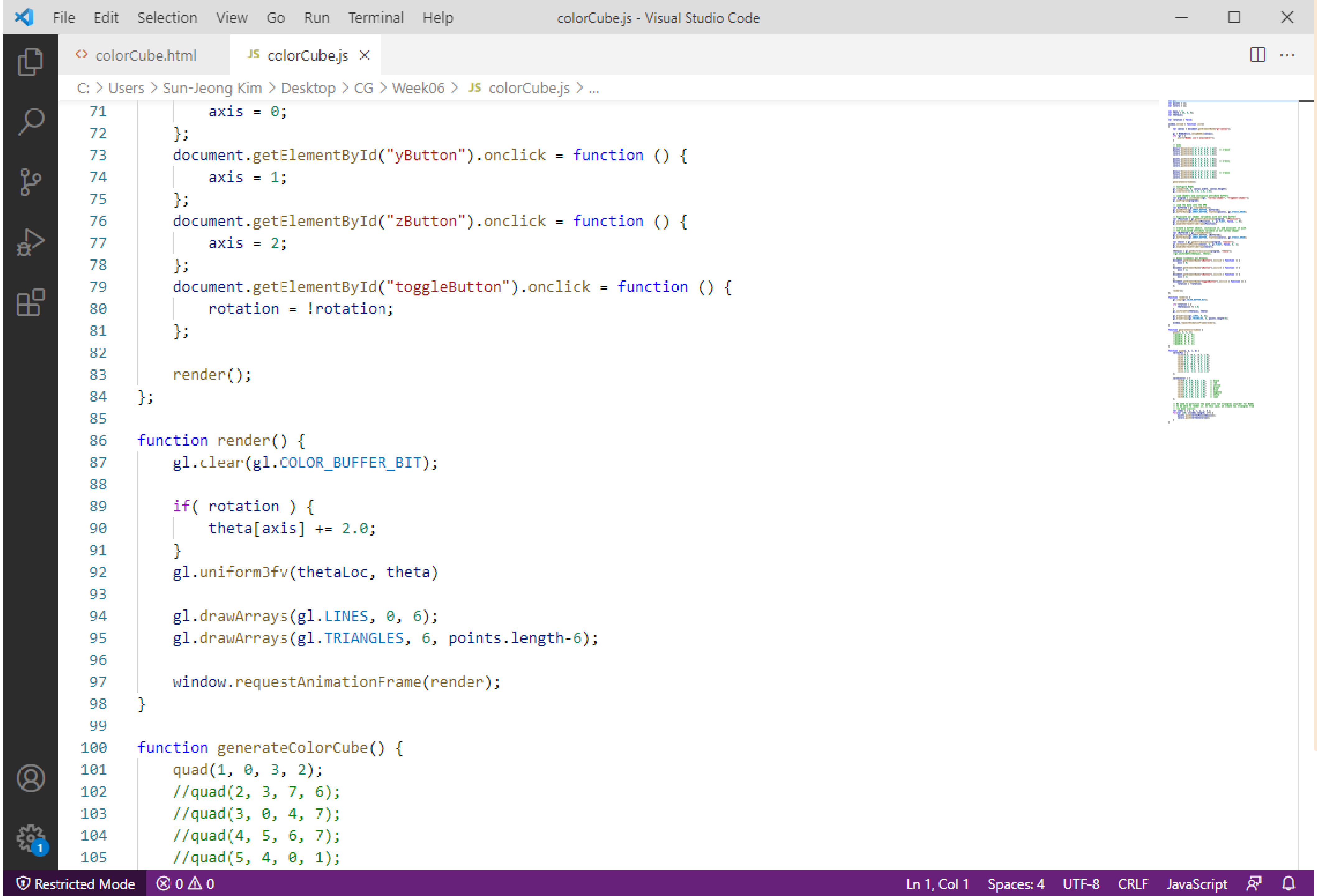
```

            gl_Position = rz * ry * rx * vPosition;
            fColor = vColor;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="../Common/webgl-utils.js"></script>
        <script type="text/javascript" src="../Common/initShaders.js"></script>
        <script type="text/javascript" src="../Common/MV.js"></script>
        <script type="text/javascript" src="colorCube.js"></script>
    </head>
    <body>
        <div>
            <button id="xButton">Rotate X</button>
            <button id="yButton">Rotate Y</button>
            <button id="zButton">Rotate Z</button>
            <button id="toggleButton">Toggle Rotate</button>
        </div>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
    </body>
</html>
```


File Edit Selection View Go Run Terminal Help
colorCube.js - Visual Studio Code
colorCube.html
colorCube.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > colorCube.js > ...
Restricted Mode
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript

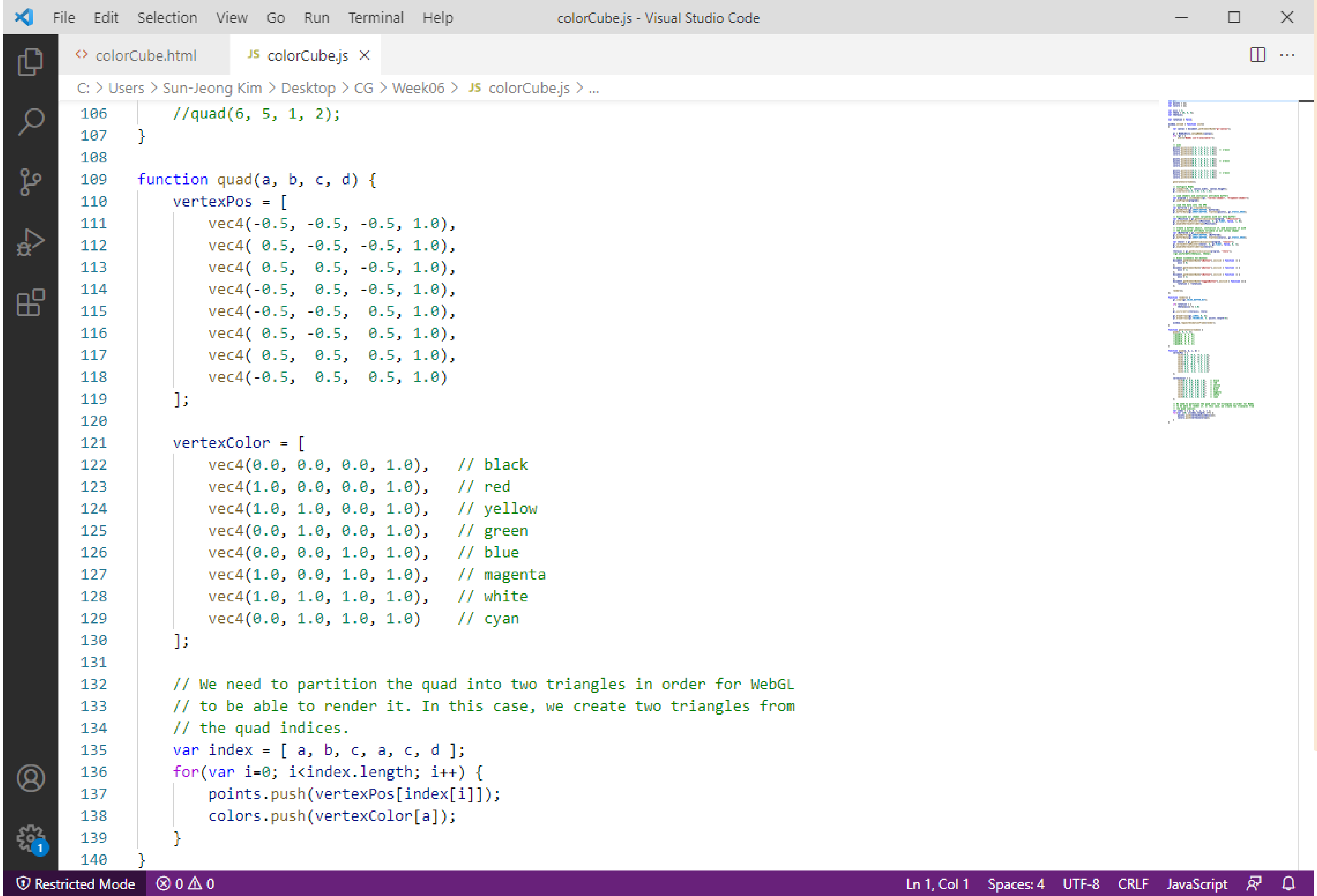
```javascript
var gl;
var points = [];
var colors = [];

var axis = 0;
var theta = [0, 0, 0];
var thetaLoc;

var rotation = false;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    // axes
    points.push(vec4(0.0, 0.0, 0.0, 1.0));
    points.push(vec4(1.0, 0.0, 0.0, 1.0));  // x-axis
    colors.push(vec4(1.0, 0.0, 0.0, 1.0));
    colors.push(vec4(1.0, 0.0, 0.0, 1.0));

    points.push(vec4(0.0, 0.0, 0.0, 1.0));
    points.push(vec4(0.0, 1.0, 0.0, 1.0));  // y-axis
    colors.push(vec4(0.0, 1.0, 0.0, 1.0));
    colors.push(vec4(0.0, 1.0, 0.0, 1.0));

    points.push(vec4(0.0, 0.0, 0.0, 1.0));
    points.push(vec4(0.0, 0.0, 1.0, 1.0));  // z-axis
    colors.push(vec4(0.0, 0.0, 1.0, 1.0));
    colors.push(vec4(0.0, 0.0, 1.0, 1.0));

```


File Edit Selection View Go Run Terminal Help
colorCube.js - Visual Studio Code
colorCube.html
colorCube.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > colorCube.js > ...
Restricted Mode
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript

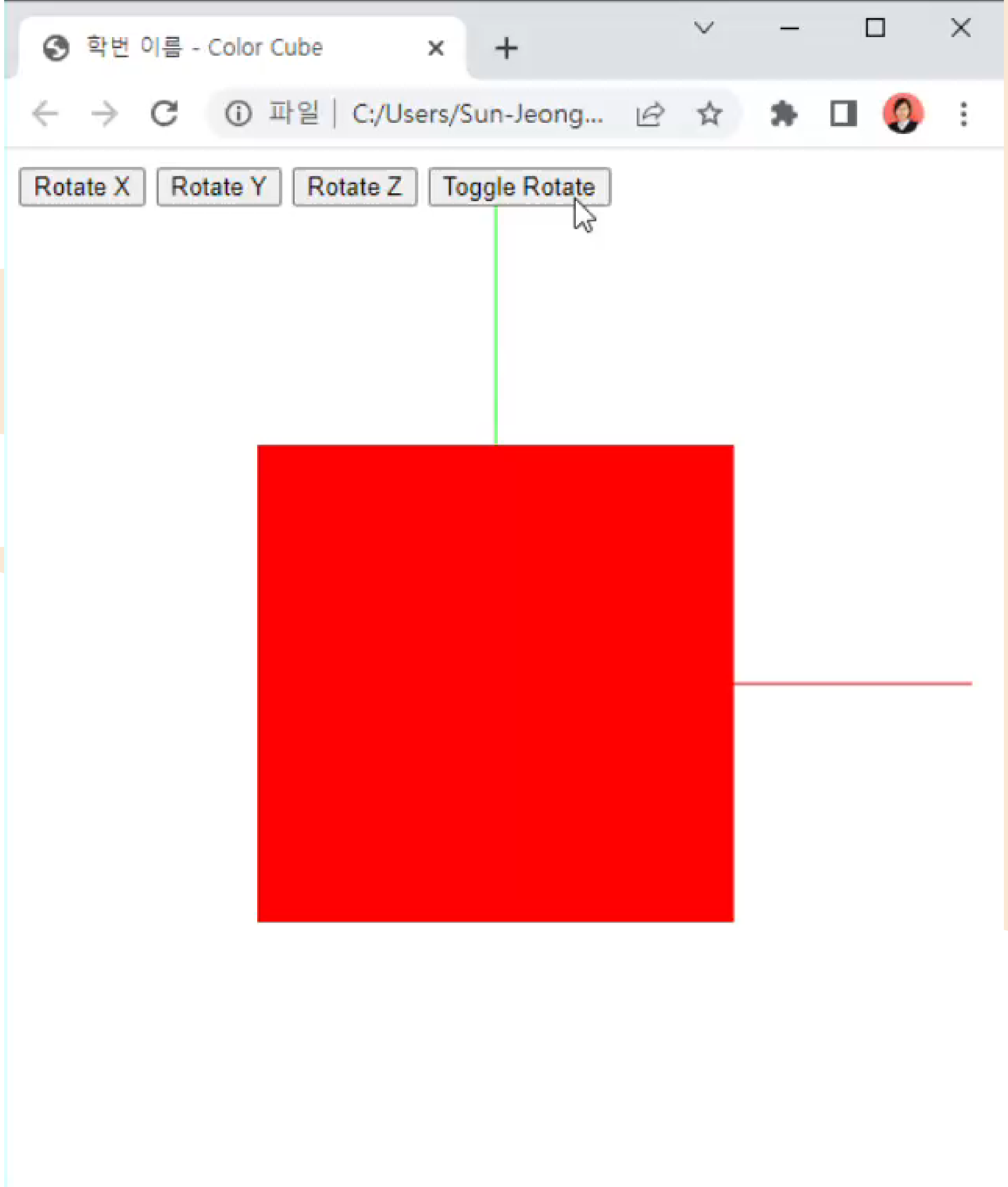
```
    generateColorCube();

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(1.0, 1.0, 1.0, 1.0);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    thetaLoc = gl.getUniformLocation(program, "theta");
    //gl.uniform3fv(thetaLoc, theta);

    // Event listeners for buttons
    document.getElementById("xButton").onclick = function () {
```



```
        axis = 0;
    };
    document.getElementById("yButton").onclick = function () {
        axis = 1;
    };
    document.getElementById("zButton").onclick = function () {
        axis = 2;
    };
    document.getElementById("toggleButton").onclick = function () {
        rotation = !rotation;
    };

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT);

    if( rotation ) {
        theta[axis] += 2.0;
    }
    gl.uniform3fv(thetaLoc, theta)

    gl.drawArrays(gl.LINES, 0, 6);
    gl.drawArrays(gl.TRIANGLES, 6, points.length-6);

    window.requestAnimationFrame(render);
}

function generateColorCube() {
    quad(1, 0, 3, 2);
    //quad(2, 3, 7, 6);
    //quad(3, 0, 4, 7);
    //quad(4, 5, 6, 7);
    //quad(5, 4, 0, 1);
```

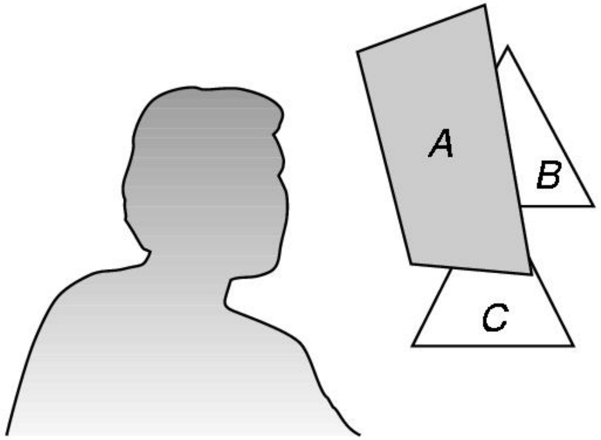

File Edit Selection View Go Run Terminal Help
colorCube.js - Visual Studio Code
colorCube.html
colorCube.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > colorCube.js > ...
Restricted Mode
0 0
Ln 1, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript

```javascript
    //quad(6, 5, 1, 2);
}

function quad(a, b, c, d) {
    vertexPos = [
        vec4(-0.5, -0.5, -0.5, 1.0),
        vec4( 0.5, -0.5, -0.5, 1.0),
        vec4( 0.5,  0.5, -0.5, 1.0),
        vec4(-0.5,  0.5, -0.5, 1.0),
        vec4(-0.5, -0.5,  0.5, 1.0),
        vec4( 0.5, -0.5,  0.5, 1.0),
        vec4( 0.5,  0.5,  0.5, 1.0),
        vec4(-0.5,  0.5,  0.5, 1.0)
    ];

    vertexColor = [
        vec4(0.0, 0.0, 0.0, 1.0),    // black
        vec4(1.0, 0.0, 0.0, 1.0),    // red
        vec4(1.0, 1.0, 0.0, 1.0),    // yellow
        vec4(0.0, 1.0, 0.0, 1.0),    // green
        vec4(0.0, 0.0, 1.0, 1.0),    // blue
        vec4(1.0, 0.0, 1.0, 1.0),    // magenta
        vec4(1.0, 1.0, 1.0, 1.0),    // white
        vec4(0.0, 1.0, 1.0, 1.0)     // cyan
    ];

    // We need to partition the quad into two triangles in order for WebGL
    // to be able to render it. In this case, we create two triangles from
    // the quad indices.
    var index = [ a, b, c, a, c, d ];
    for(var i=0; i<index.length; i++) {
        points.push(vertexPos[index[i]]);
        colors.push(vertexColor[a]);
    }
}
```


학번 이름 - Color Cube
파일 | C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

Hidden-Surface Removal

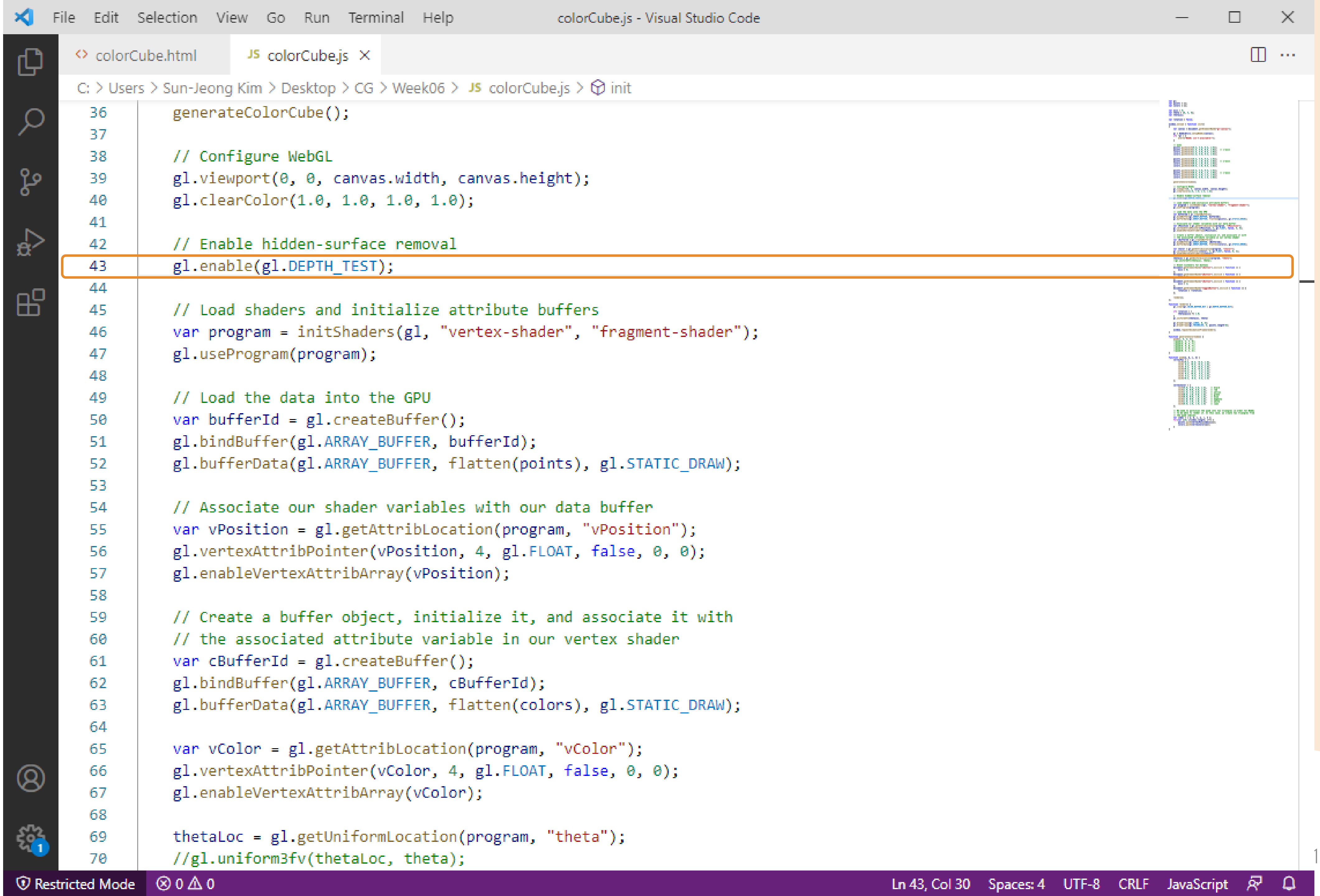
- To see only those surfaces in front of other surfaces
- Visible-surface algorithms or hidden-surface-removal algorithm
  - Algorithms for ordering objects
  - OpenGL uses the z-buffer algorithm that saves depth information



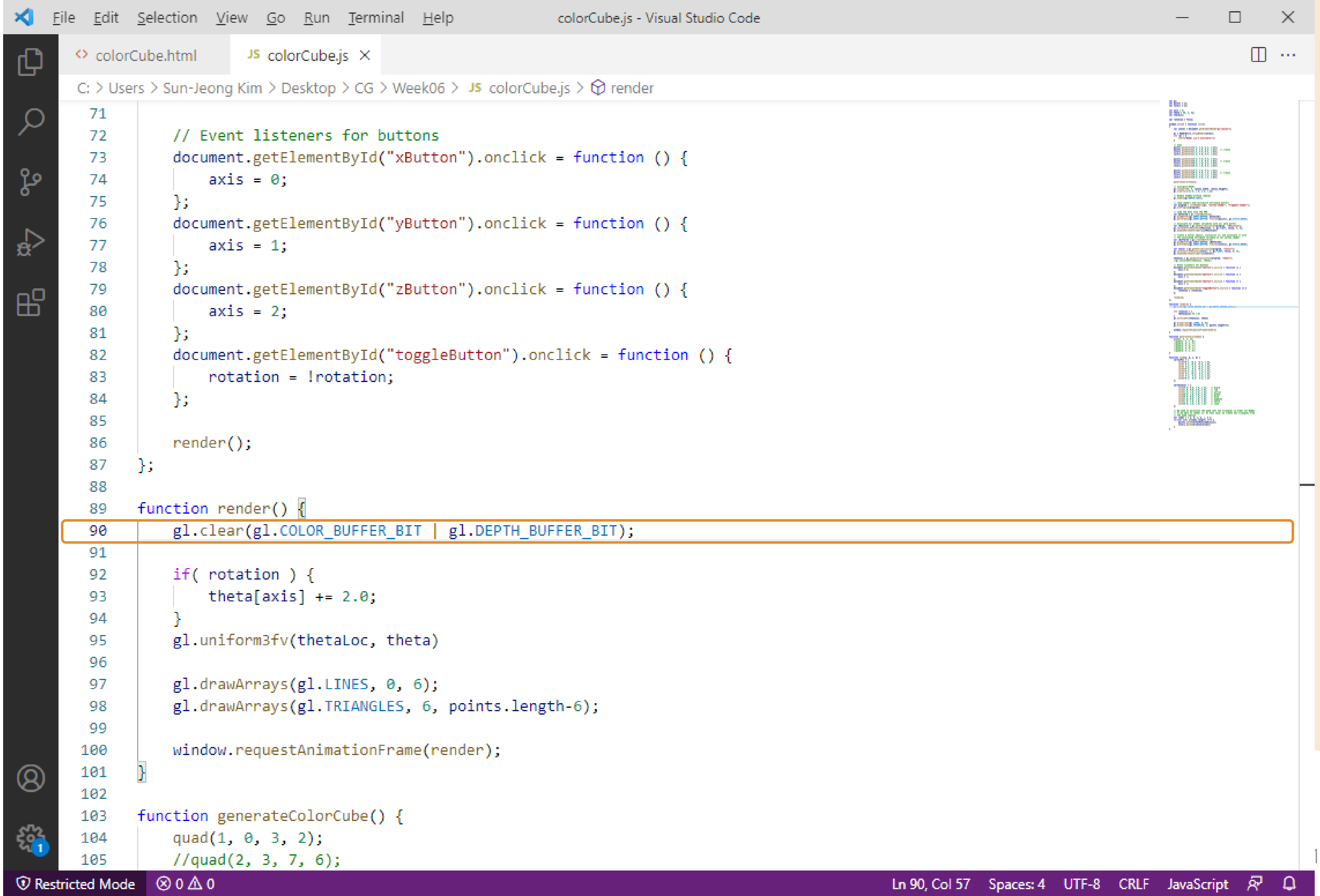
Hidden-Surface Problem

Using the Z-buffer Algorithm

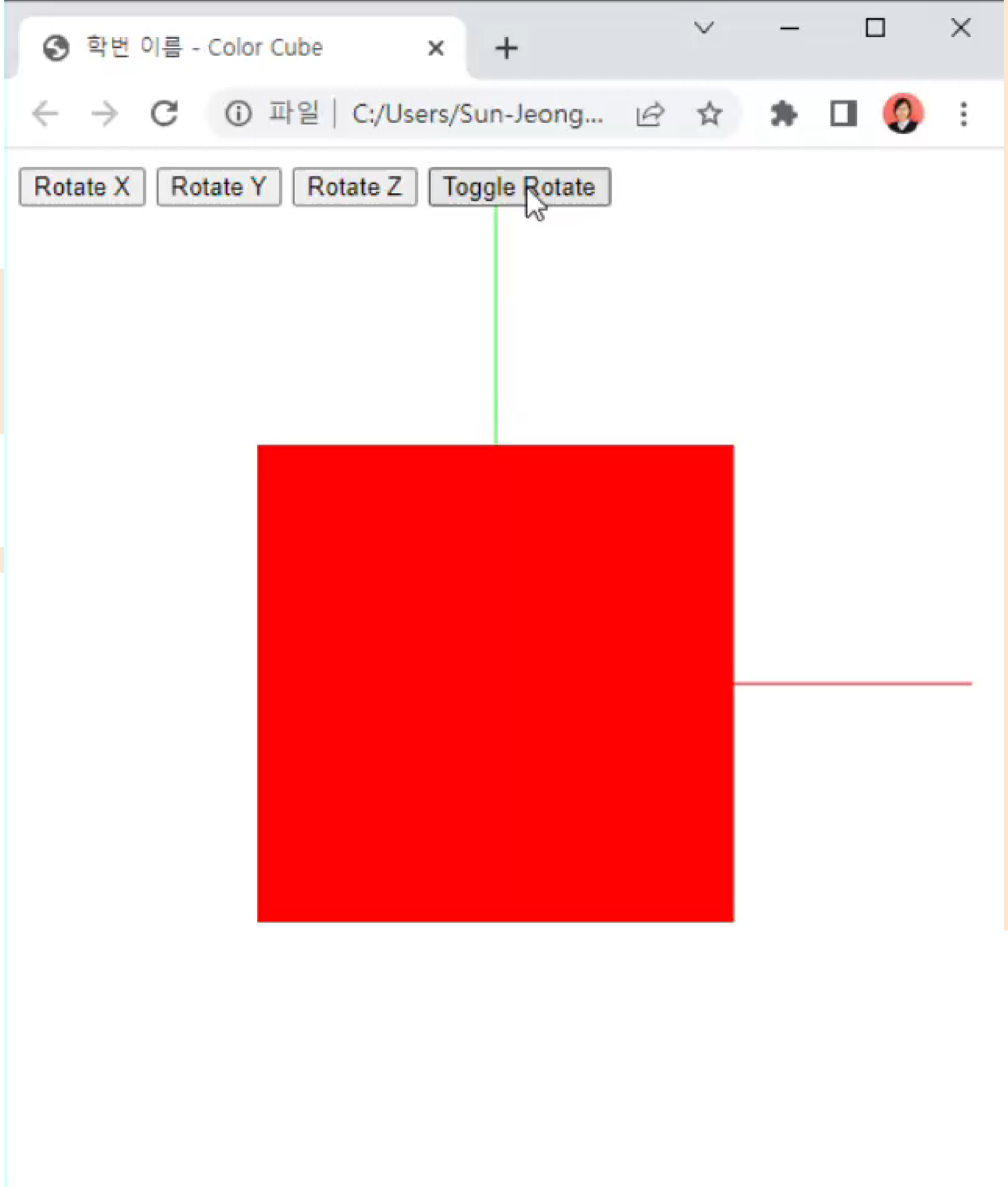
- The algorithm uses an extra buffer, the z-buffer, to store depth information as geometry travels down the pipeline
- Depth buffer is required to available in WebGL
- It must be
  - Enabled
    - `gl.enable( gl.DEPTH_TEST );`
  - Cleared in for each render
    - `gl.clear( gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT );`

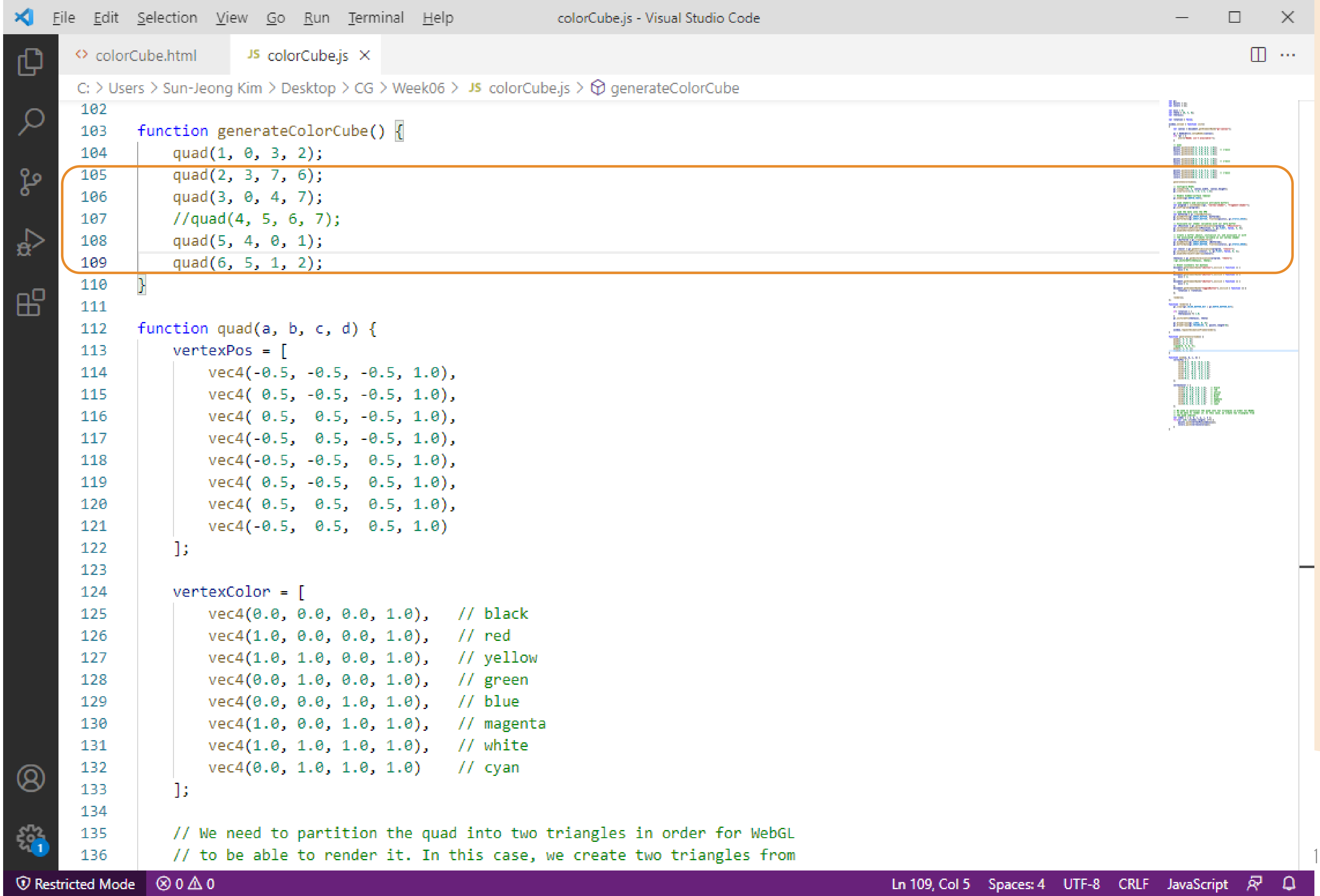


```
    generateColorCube();

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(1.0, 1.0, 1.0, 1.0);

    // Enable hidden-surface removal
    gl.enable(gl.DEPTH_TEST);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    thetaLoc = gl.getUniformLocation(program, "theta");
    //gl.uniform3fv(thetaLoc, theta);
```

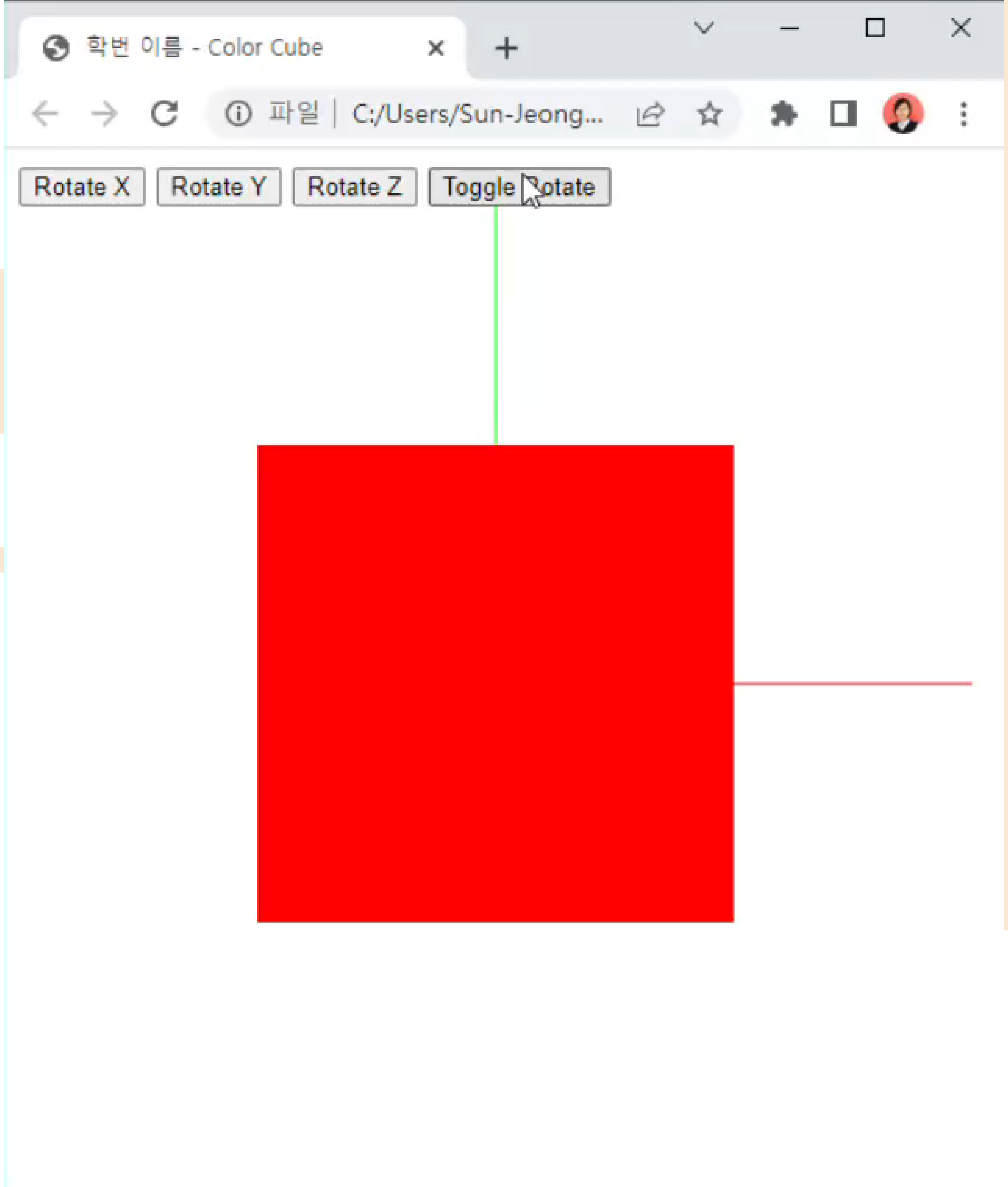

File Edit Selection View Go Run Terminal Help
colorCube.js - Visual Studio Code
colorCube.html
colorCube.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > colorCube.js > render
Restricted Mode
0 0
Ln 90, Col 57
Spaces: 4
UTF-8
CRLF
JavaScript

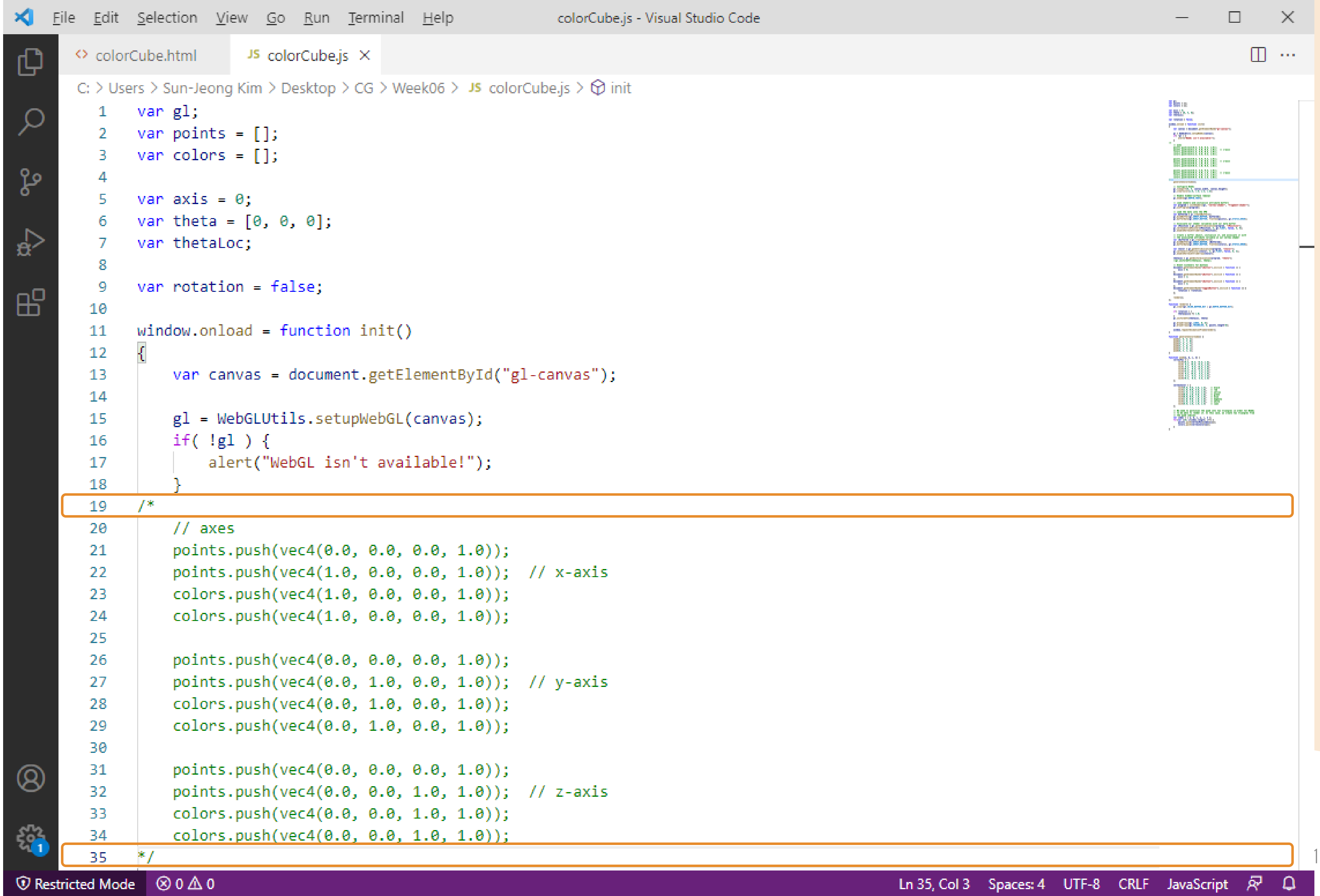
```javascript

    // Event listeners for buttons
    document.getElementById("xButton").onclick = function () {
        axis = 0;
    };
    document.getElementById("yButton").onclick = function () {
        axis = 1;
    };
    document.getElementById("zButton").onclick = function () {
        axis = 2;
    };
    document.getElementById("toggleButton").onclick = function () {
        rotation = !rotation;
    };

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    if( rotation ) {
        theta[axis] += 2.0;
    }
    gl.uniform3fv(thetaLoc, theta)

    gl.drawArrays(gl.LINES, 0, 6);
    gl.drawArrays(gl.TRIANGLES, 6, points.length-6);

    window.requestAnimationFrame(render);
}

function generateColorCube() {
    quad(1, 0, 3, 2);
    //quad(2, 3, 7, 6);
```


학번 이름 - Color Cube
파일 | C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

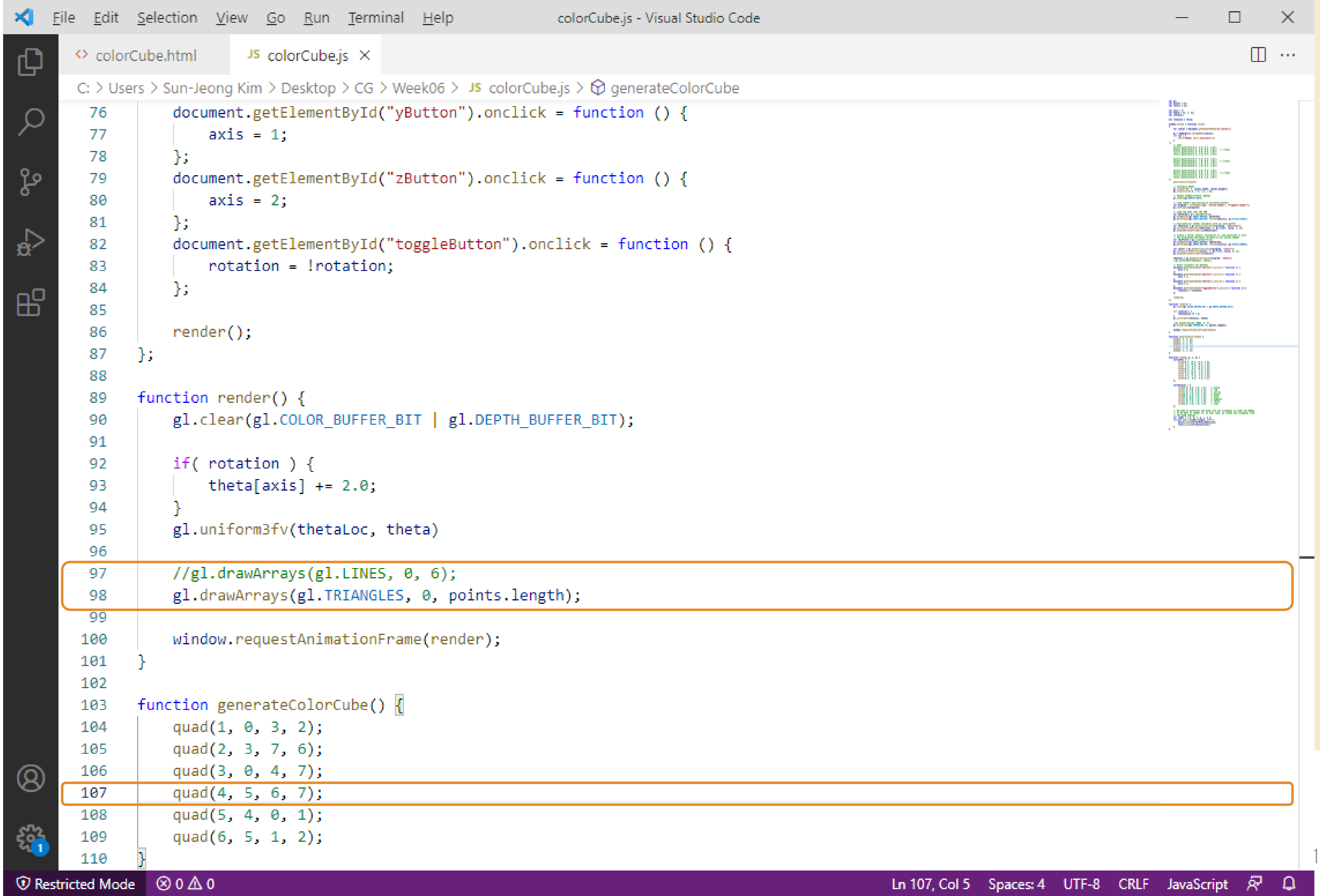


```javascript

function generateColorCube() {
    quad(1, 0, 3, 2);
    quad(2, 3, 7, 6);
    quad(3, 0, 4, 7);
    //quad(4, 5, 6, 7);
    quad(5, 4, 0, 1);
    quad(6, 5, 1, 2);
}

function quad(a, b, c, d) {
    vertexPos = [
        vec4(-0.5, -0.5, -0.5, 1.0),
        vec4( 0.5, -0.5, -0.5, 1.0),
        vec4( 0.5,  0.5, -0.5, 1.0),
        vec4(-0.5,  0.5, -0.5, 1.0),
        vec4(-0.5, -0.5,  0.5, 1.0),
        vec4( 0.5, -0.5,  0.5, 1.0),
        vec4( 0.5,  0.5,  0.5, 1.0),
        vec4(-0.5,  0.5,  0.5, 1.0)
    ];

    vertexColor = [
        vec4(0.0, 0.0, 0.0, 1.0),    // black
        vec4(1.0, 0.0, 0.0, 1.0),    // red
        vec4(1.0, 1.0, 0.0, 1.0),    // yellow
        vec4(0.0, 1.0, 0.0, 1.0),    // green
        vec4(0.0, 0.0, 1.0, 1.0),    // blue
        vec4(1.0, 0.0, 1.0, 1.0),    // magenta
        vec4(1.0, 1.0, 1.0, 1.0),    // white
        vec4(0.0, 1.0, 1.0, 1.0)     // cyan
    ];

    // We need to partition the quad into two triangles in order for WebGL
    // to be able to render it. In this case, we create two triangles from
```

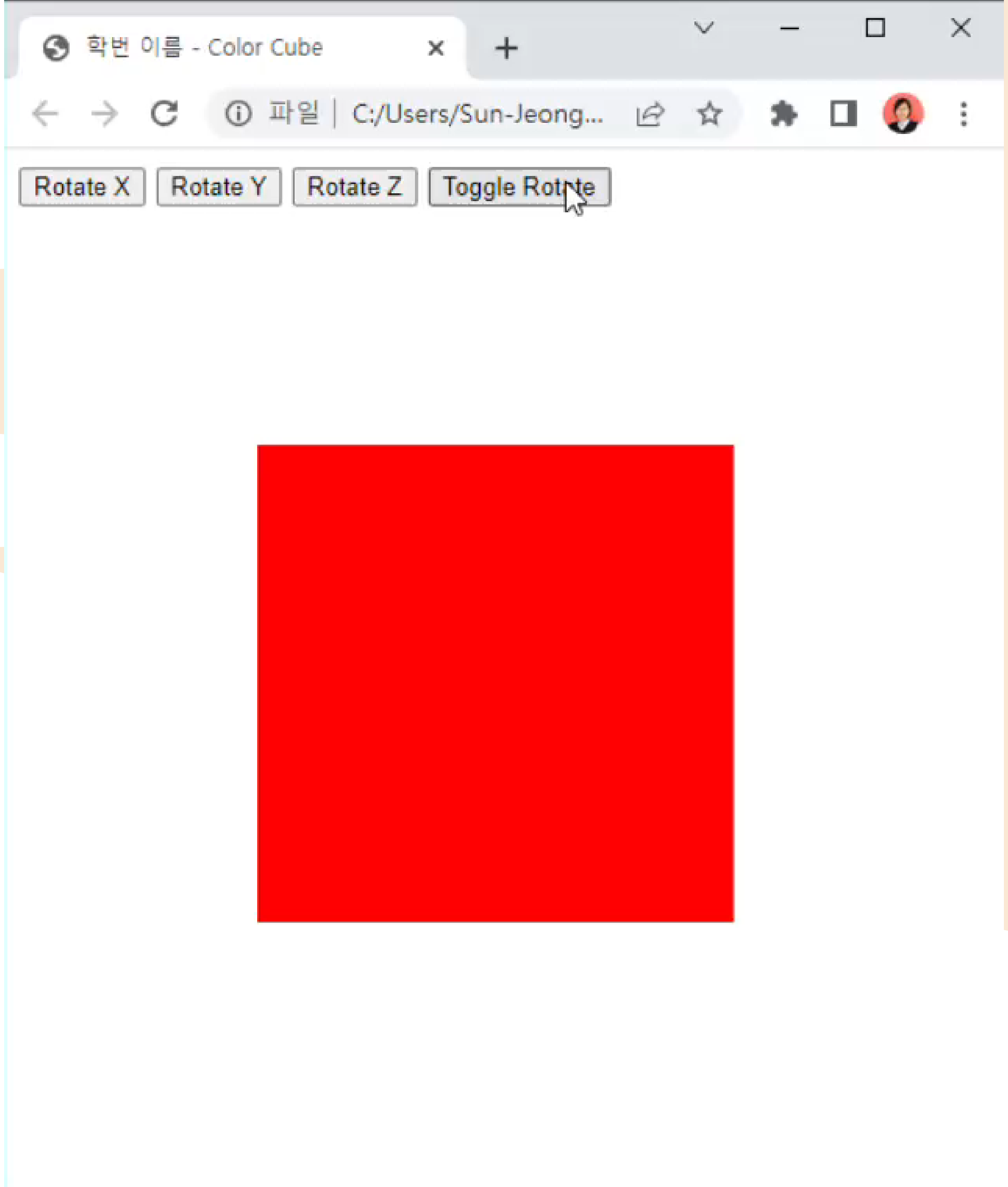

학번 이름 - Color Cube
파일 | C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate



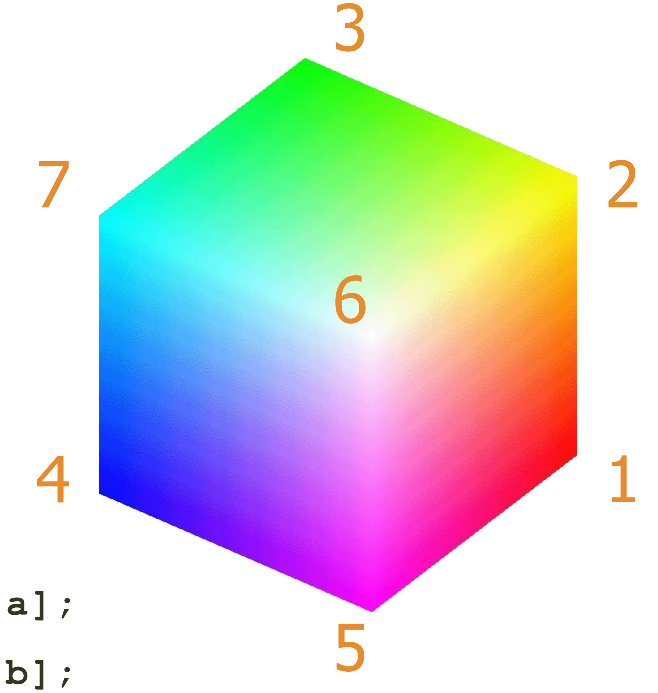
```
var gl;
var points = [];
var colors = [];

var axis = 0;
var theta = [0, 0, 0];
var thetaLoc;

var rotation = false;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }
/*
    // axes
    points.push(vec4(0.0, 0.0, 0.0, 1.0));
    points.push(vec4(1.0, 0.0, 0.0, 1.0));  // x-axis
    colors.push(vec4(1.0, 0.0, 0.0, 1.0));
    colors.push(vec4(1.0, 0.0, 0.0, 1.0));

    points.push(vec4(0.0, 0.0, 0.0, 1.0));
    points.push(vec4(0.0, 1.0, 0.0, 1.0));  // y-axis
    colors.push(vec4(0.0, 1.0, 0.0, 1.0));
    colors.push(vec4(0.0, 1.0, 0.0, 1.0));

    points.push(vec4(0.0, 0.0, 0.0, 1.0));
    points.push(vec4(0.0, 0.0, 1.0, 1.0));  // z-axis
    colors.push(vec4(0.0, 0.0, 1.0, 1.0));
    colors.push(vec4(0.0, 0.0, 1.0, 1.0));
*/
```



```javascript
    document.getElementById("yButton").onclick = function () {
        axis = 1;
    };
    document.getElementById("zButton").onclick = function () {
        axis = 2;
    };
    document.getElementById("toggleButton").onclick = function () {
        rotation = !rotation;
    };

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    if( rotation ) {
        theta[axis] += 2.0;
    }
    gl.uniform3fv(thetaLoc, theta)

    //gl.drawArrays(gl.LINES, 0, 6);
    gl.drawArrays(gl.TRIANGLES, 0, points.length);

    window.requestAnimationFrame(render);
}

function generateColorCube() {
    quad(1, 0, 3, 2);
    quad(2, 3, 7, 6);
    quad(3, 0, 4, 7);
    quad(4, 5, 6, 7);
    quad(5, 4, 0, 1);
    quad(6, 5, 1, 2);
}
```


학번 이름 - Color Cube
파일 | C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

Color Cube (1)



```
float vertex_pos[8][3]= { {-1.0, -1.0, -1.0}, {1.0, -1.0, -1.0},
        {1.0, 1.0, -1.0}, {-1.0, 1.0, -1.0}, {-1.0, -1.0, 1.0},
        {1.0, -1.0, 1.0}, {1.0, 1.0, 1.0}, {-1.0, 1.0, 1.0}};

float vertex_color[8][3] = { {0.0, 0.0, 0.0}, {1.0, 0.0, 0.0},
        {1.0, 1.0, 0.0}, {0.0, 1.0, 0.0}, {0.0, 0.0, 1.0},
        {1.0, 0.0, 1.0}, {1.0, 1.0, 1.0}, {0.0, 1.0, 1.0}};

int Index = 0;
void quad( int a, int b, int c, int d )
{
    colors[Index] = vertex_color[a];    points[Index] = vertex_pos[a];
    Index++;
    colors[Index] = vertex_color[b];    points[Index] = vertex_pos[b];
    Index++;
    colors[Index] = vertex_color[c];    points[Index] = vertex_pos[c];
    Index++;

    colors[Index] = vertex_color[a];    points[Index] = vertex_pos[a];
    Index++;
    colors[Index] = vertex_color[c];    points[Index] = vertex_pos[c];
    Index++;
    colors[Index] = vertex_color[b];    points[Index] = vertex_pos[b];
    Index++;
}
```

Color Cube (2)

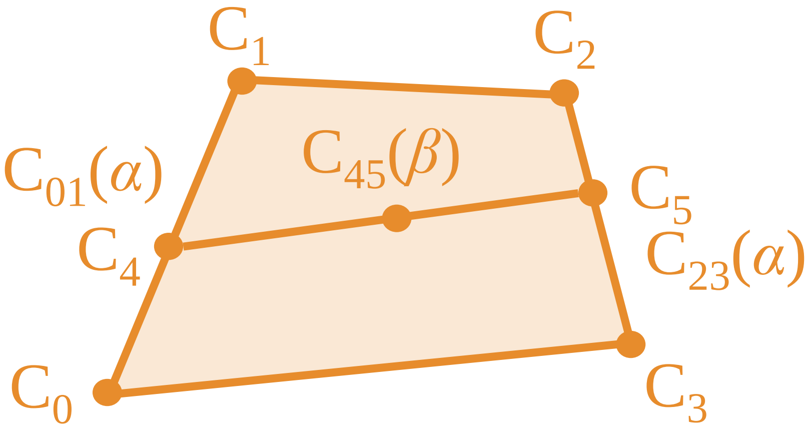
```
void generateColorCube()
{
    quad( 1, 0, 3, 2 );
    quad( 2, 3, 7, 6 );
    quad( 3, 0, 4, 7 );
    quad( 4, 5, 6, 7 );
    quad( 5, 4, 0, 1 );
    quad( 6, 5, 1, 2 );
}
```

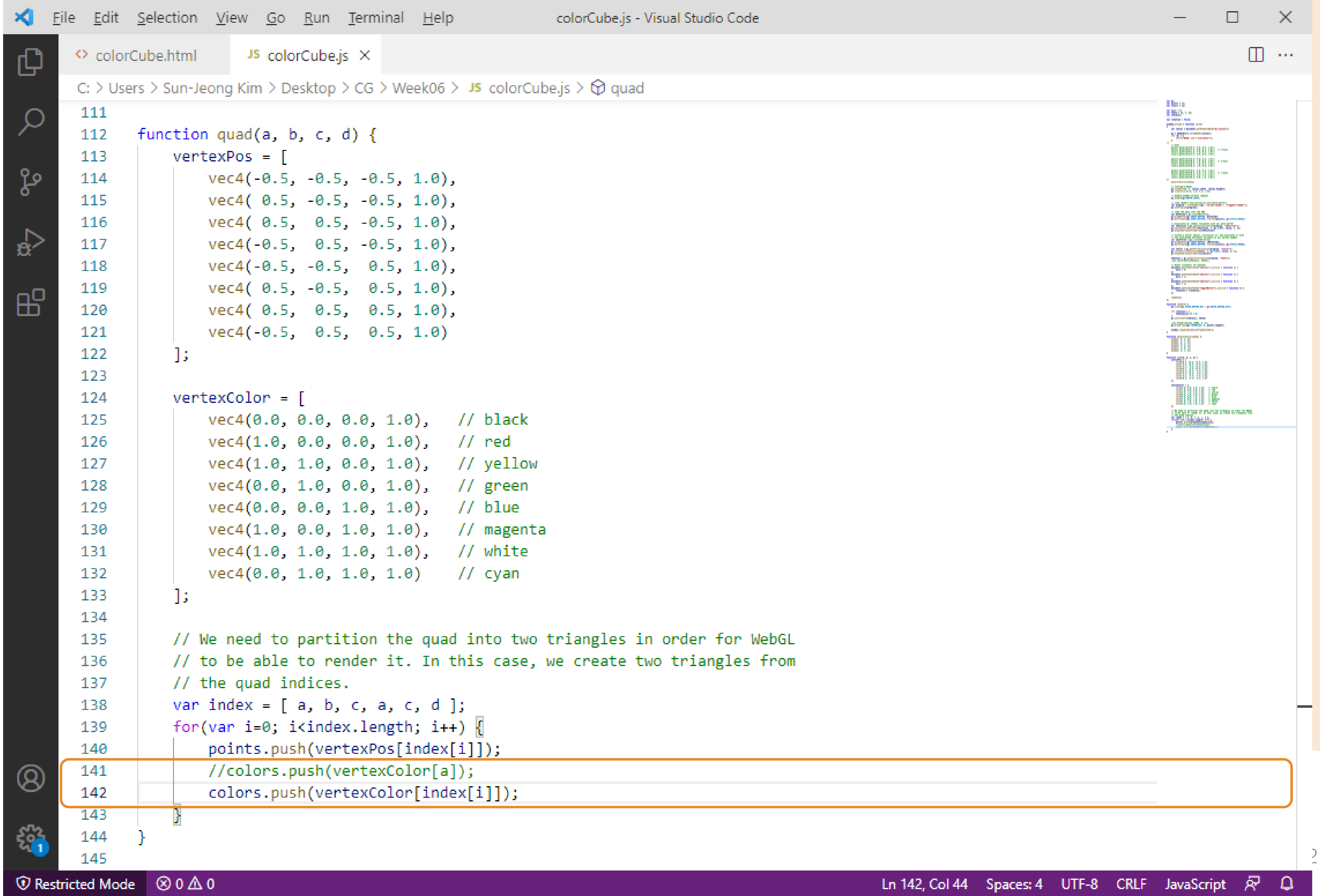
- Bilinear interpolation

$$C_{01}(\alpha) = (1-\alpha)C_0 + \alpha C_1$$

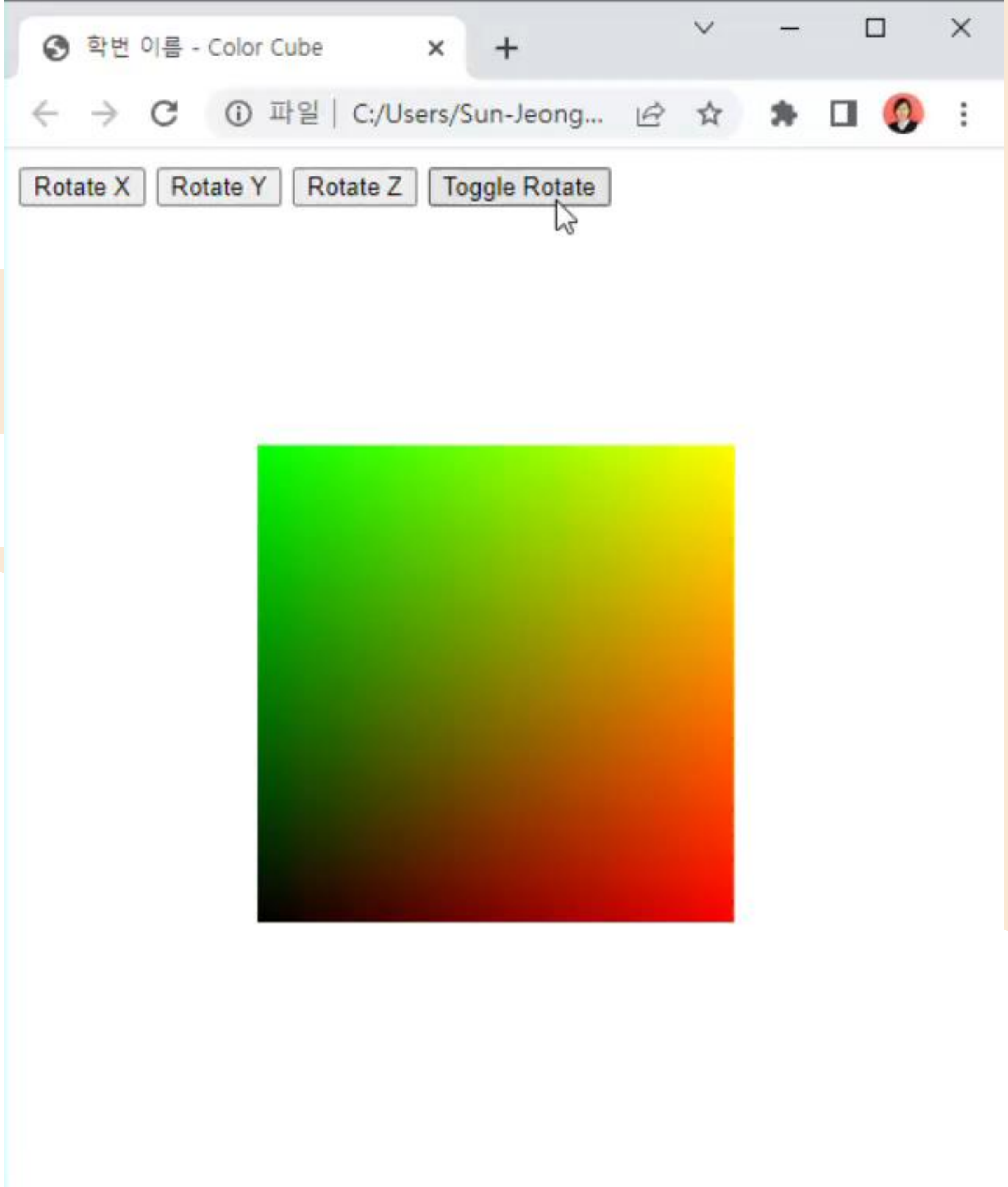
$$C_{23}(\alpha) = (1-\alpha)C_2 + \alpha C_3$$

$$C_{45}(\beta) = (1-\beta)C_4 + \beta C_5$$



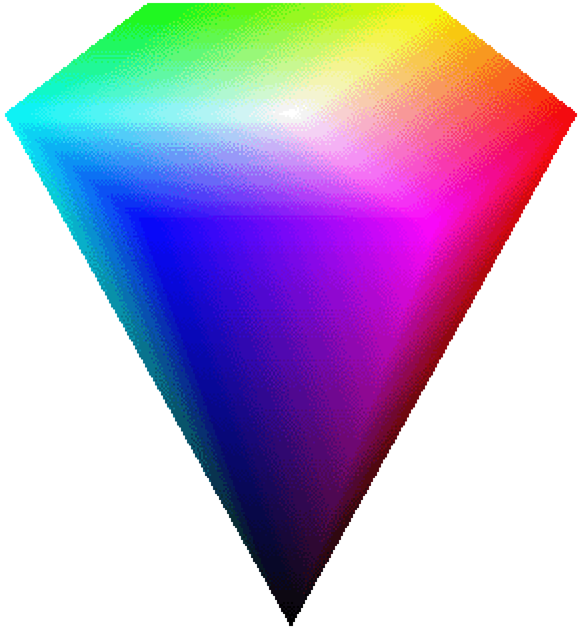


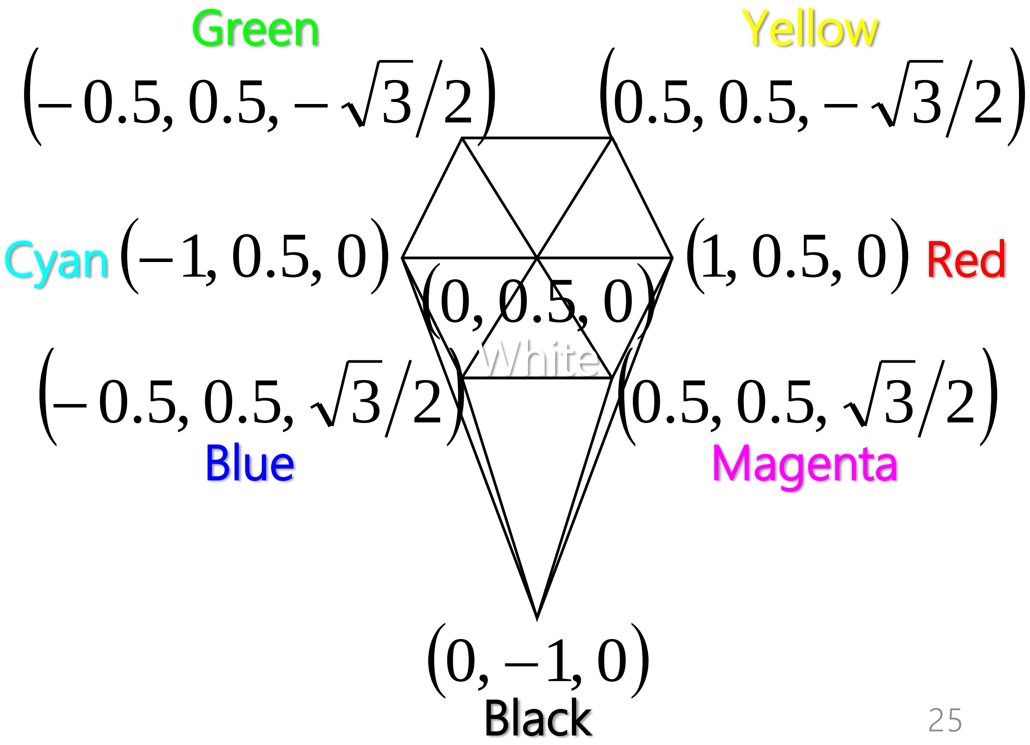
```
function quad(a, b, c, d) {
    vertexPos = [
        vec4(-0.5, -0.5, -0.5, 1.0),
        vec4( 0.5, -0.5, -0.5, 1.0),
        vec4( 0.5,  0.5, -0.5, 1.0),
        vec4(-0.5,  0.5, -0.5, 1.0),
        vec4(-0.5, -0.5,  0.5, 1.0),
        vec4( 0.5, -0.5,  0.5, 1.0),
        vec4( 0.5,  0.5,  0.5, 1.0),
        vec4(-0.5,  0.5,  0.5, 1.0)
    ];

    vertexColor = [
        vec4(0.0, 0.0, 0.0, 1.0),   // black
        vec4(1.0, 0.0, 0.0, 1.0),   // red
        vec4(1.0, 1.0, 0.0, 1.0),   // yellow
        vec4(0.0, 1.0, 0.0, 1.0),   // green
        vec4(0.0, 0.0, 1.0, 1.0),   // blue
        vec4(1.0, 0.0, 1.0, 1.0),   // magenta
        vec4(1.0, 1.0, 1.0, 1.0),   // white
        vec4(0.0, 1.0, 1.0, 1.0)    // cyan
    ];

    // We need to partition the quad into two triangles in order for WebGL
    // to be able to render it. In this case, we create two triangles from
    // the quad indices.
    var index = [ a, b, c, a, c, d ];
    for(var i=0; i<index.length; i++) {
        points.push(vertexPos[index[i]]);
        //colors.push(vertexColor[a]);
        colors.push(vertexColor[index[i]]);
    }
}
```

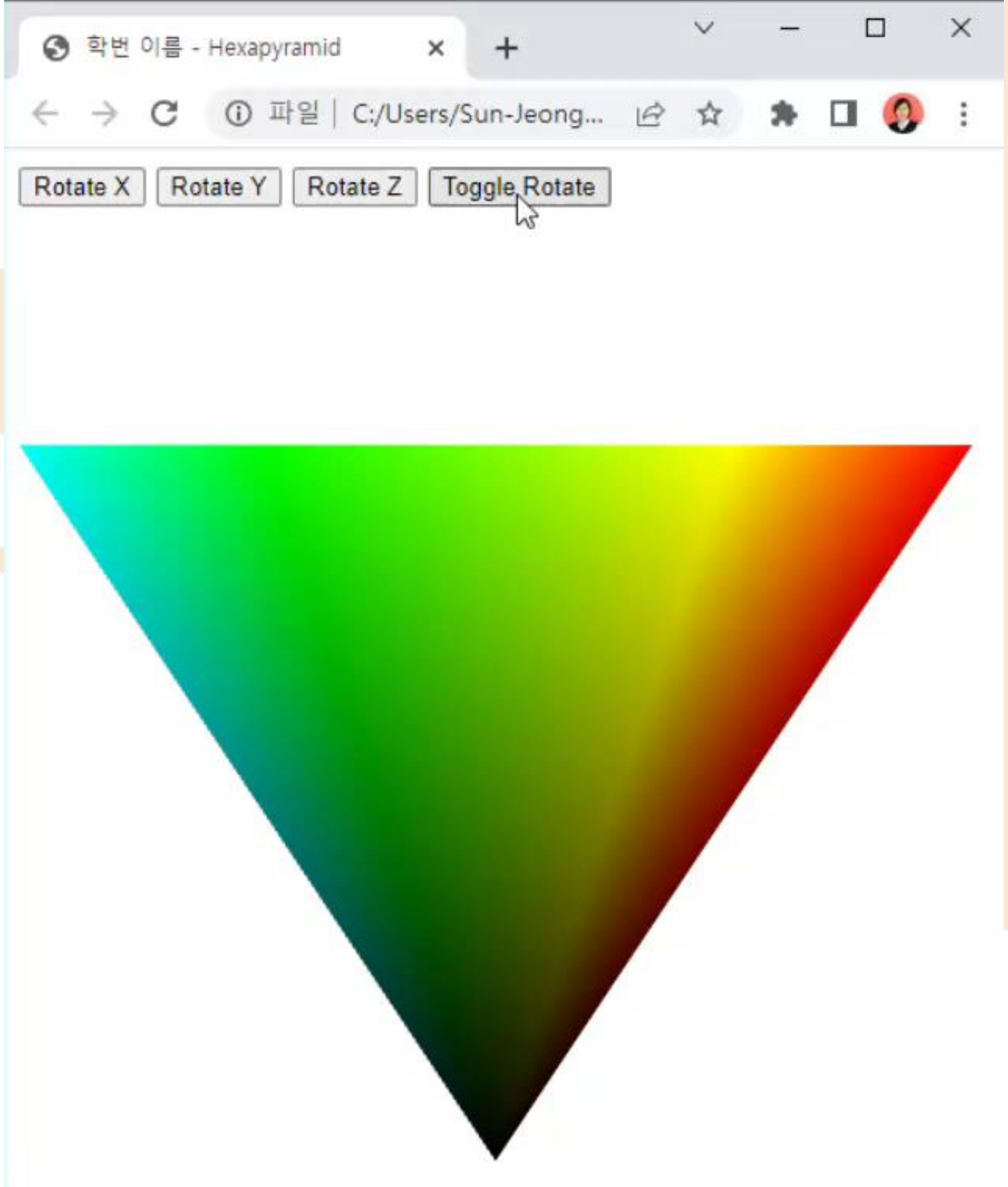

학번 이름 - Color Cube
파일 | C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

연습 문제 (1)

- Hexagonal Pyramid를 그리시오.
  - colorCube.html, js ➔ hexaPyramid.html, js로 복사
  - Vertex 8개
  - Triangle 12개

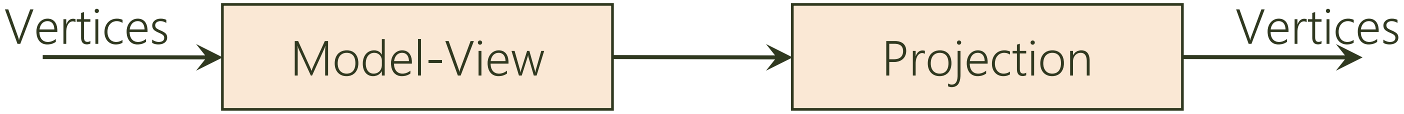



Green
$\left(-0.5, 0.5, -\sqrt{3}/2\right)$
Yellow
$\left(0.5, 0.5, -\sqrt{3}/2\right)$
Cyan $\left(-1, 0.5, 0\right)$
$\left(1, 0.5, 0\right)$ Red
$\left(0, 0.5, 0\right)$
White
$\left(-0.5, 0.5, \sqrt{3}/2\right)$
Blue
$\left(0.5, 0.5, \sqrt{3}/2\right)$
Magenta
$\left(0, -1, 0\right)$
Black

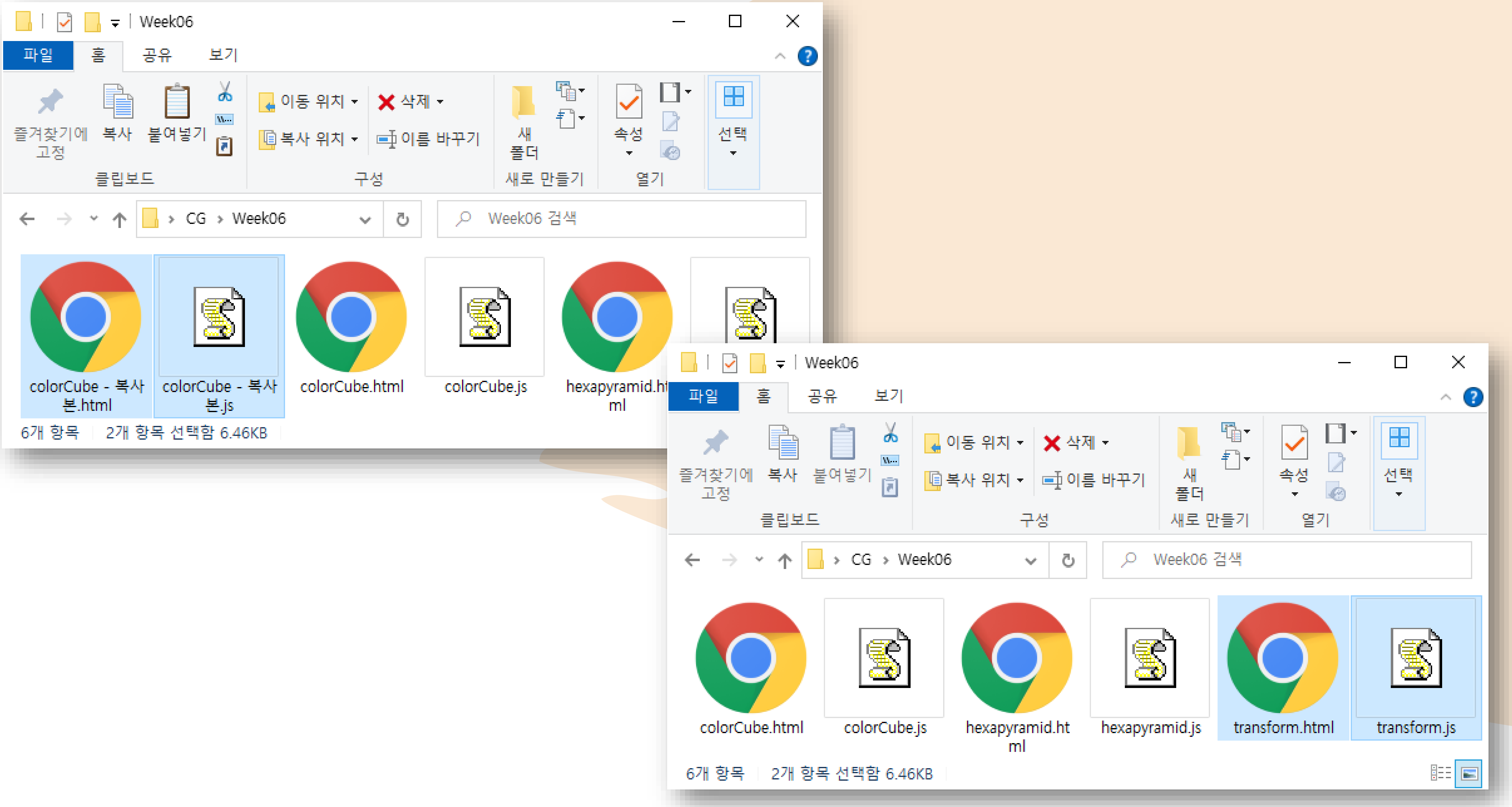

학번 이름 - Hexapyramid
파일 | C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

Line-Preserving Transformations

- Two classes of transformations of importance to CG that preserves lines
  - Affine and projective transformations
- Homogeneous coordinates
  - Vectors: $(x, y, z, \underline{0})$, points: $(x, y, z, w) = (x/w, y/w, z/w, \underline{1})$
  - 4x4 matrices: modeling, viewing, and projection
- OpenGL pipeline



  - Model-view: to position objects relative to camera
  - Projection: to specify clipping volume and map vertices to a normalized coordinate system

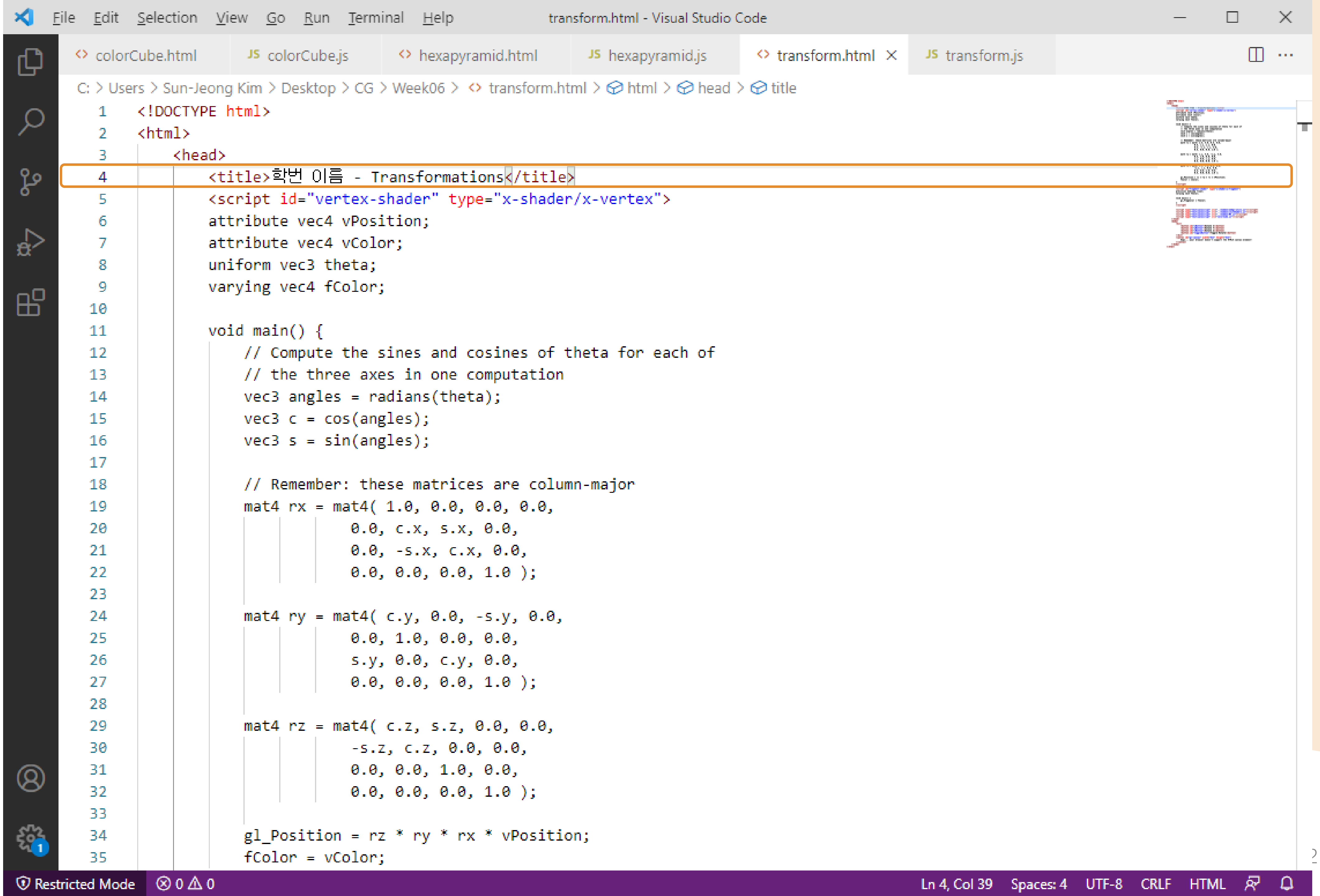


colorCube - 복사본.html
colorCube - 복사본.js
colorCube.html
colorCube.js
hexapyramid.html

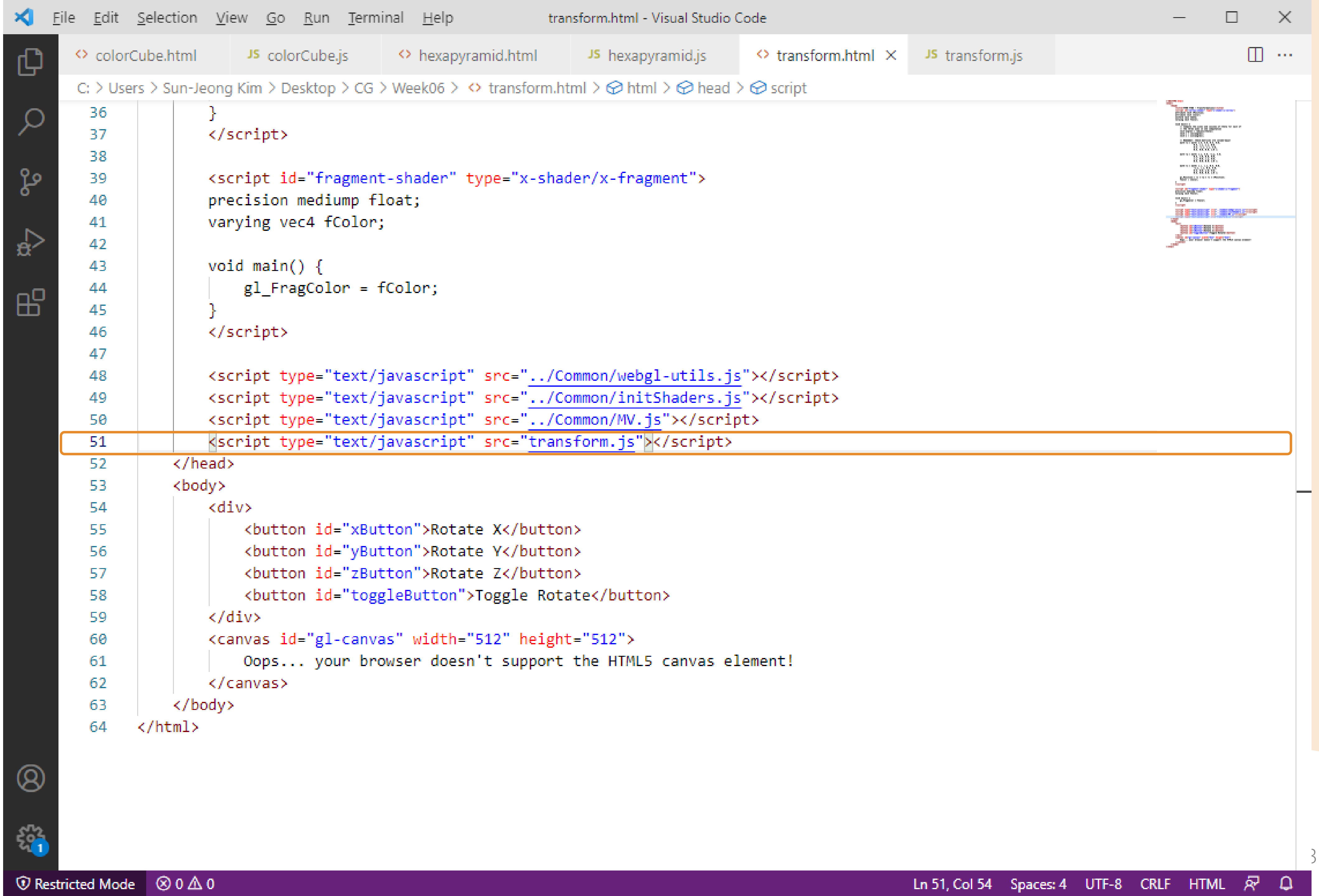
6개 항목 2개 항목 선택함 6.46KB

colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js

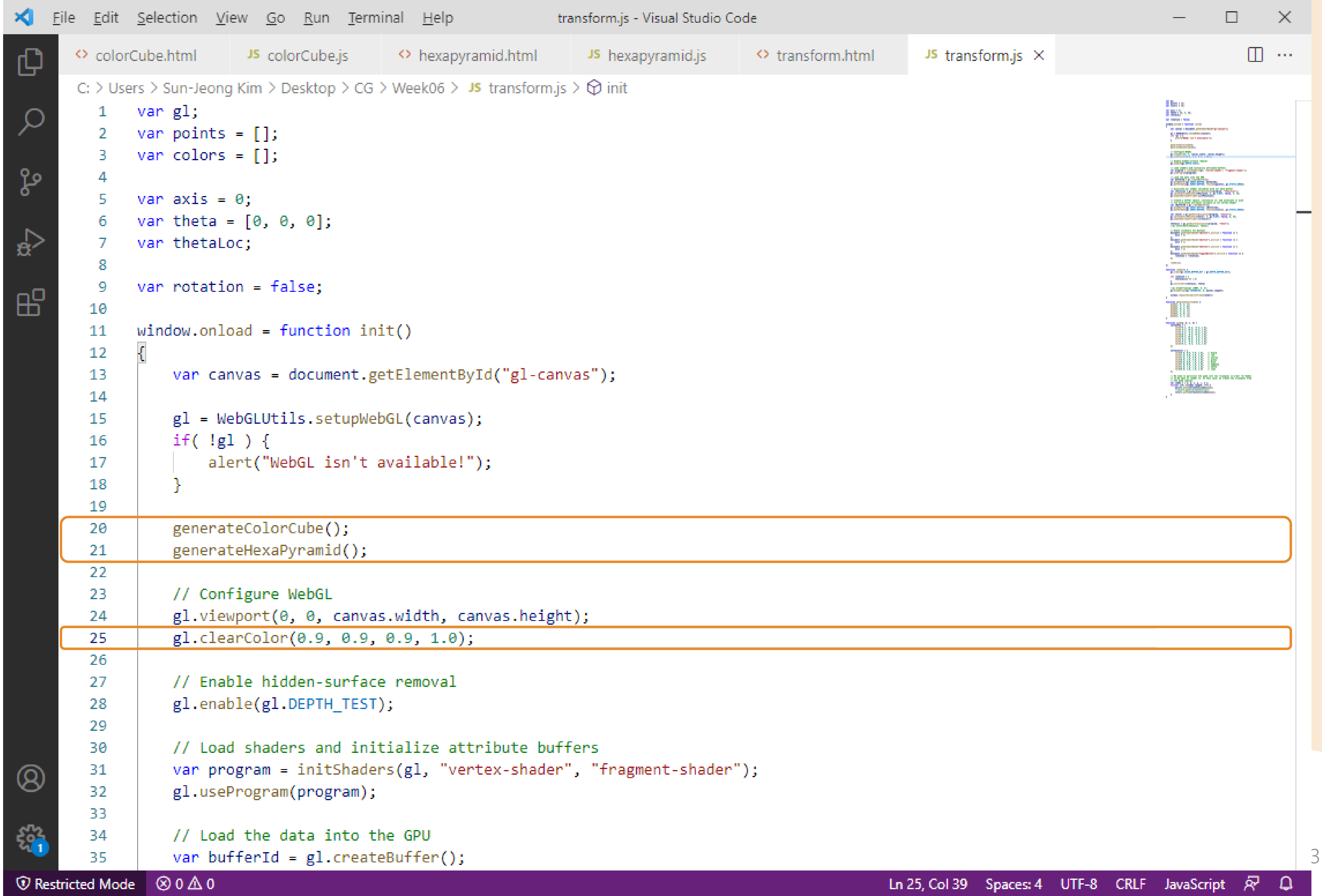
6개 항목 2개 항목 선택함 6.46KB


File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > title
Restricted Mode
0 0
Ln 4, Col 39
Spaces: 4
UTF-8
CRLF
HTML

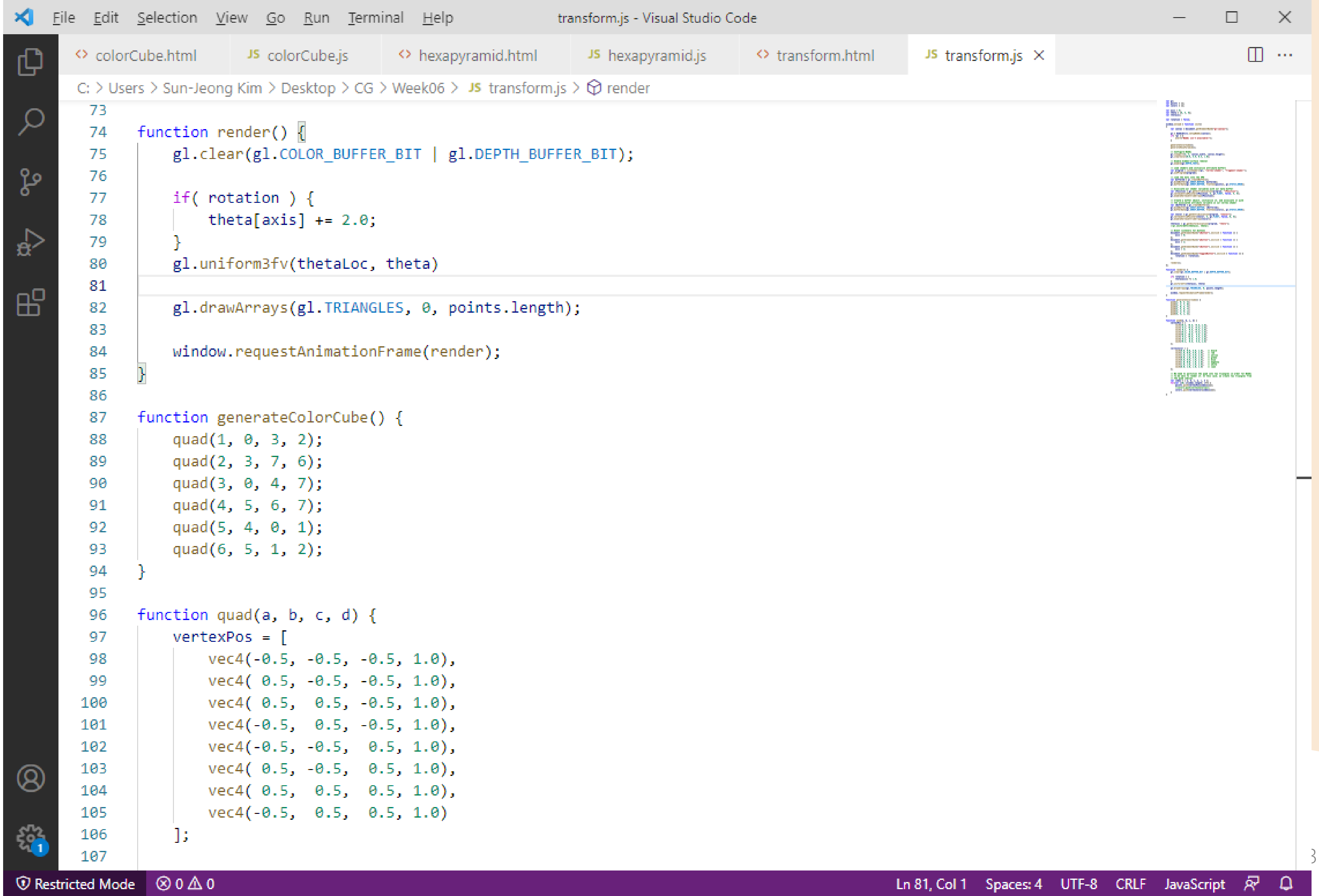
```
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Transformations</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform vec3 theta;
        varying vec4 fColor;

        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                            0.0, c.x, s.x, 0.0,
                            0.0, -s.x, c.x, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                            0.0, 1.0, 0.0, 0.0,
                            s.y, 0.0, c.y, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                            -s.z, c.z, 0.0, 0.0,
                            0.0, 0.0, 1.0, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            gl_Position = rz * ry * rx * vPosition;
            fColor = vColor;
```


File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script
Restricted Mode
0 0
Ln 51, Col 54 Spaces: 4 UTF-8 CRLF HTML

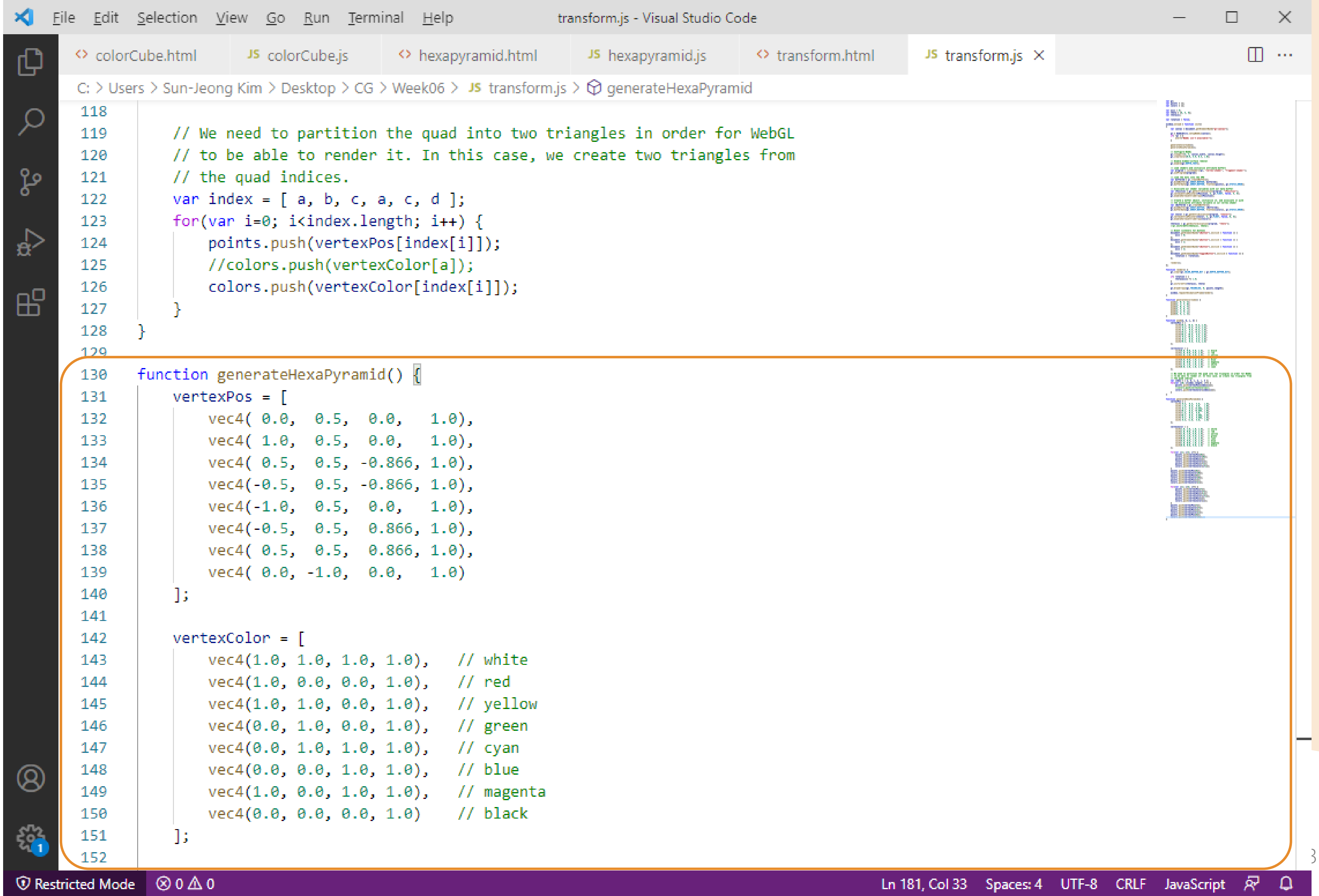
```
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="../Common/webgl-utils.js"></script>
        <script type="text/javascript" src="../Common/initShaders.js"></script>
        <script type="text/javascript" src="../Common/MV.js"></script>
        <script type="text/javascript" src="transform.js"></script>
    </head>
    <body>
        <div>
            <button id="xButton">Rotate X</button>
            <button id="yButton">Rotate Y</button>
            <button id="zButton">Rotate Z</button>
            <button id="toggleButton">Toggle Rotate</button>
        </div>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
    </body>
</html>
```



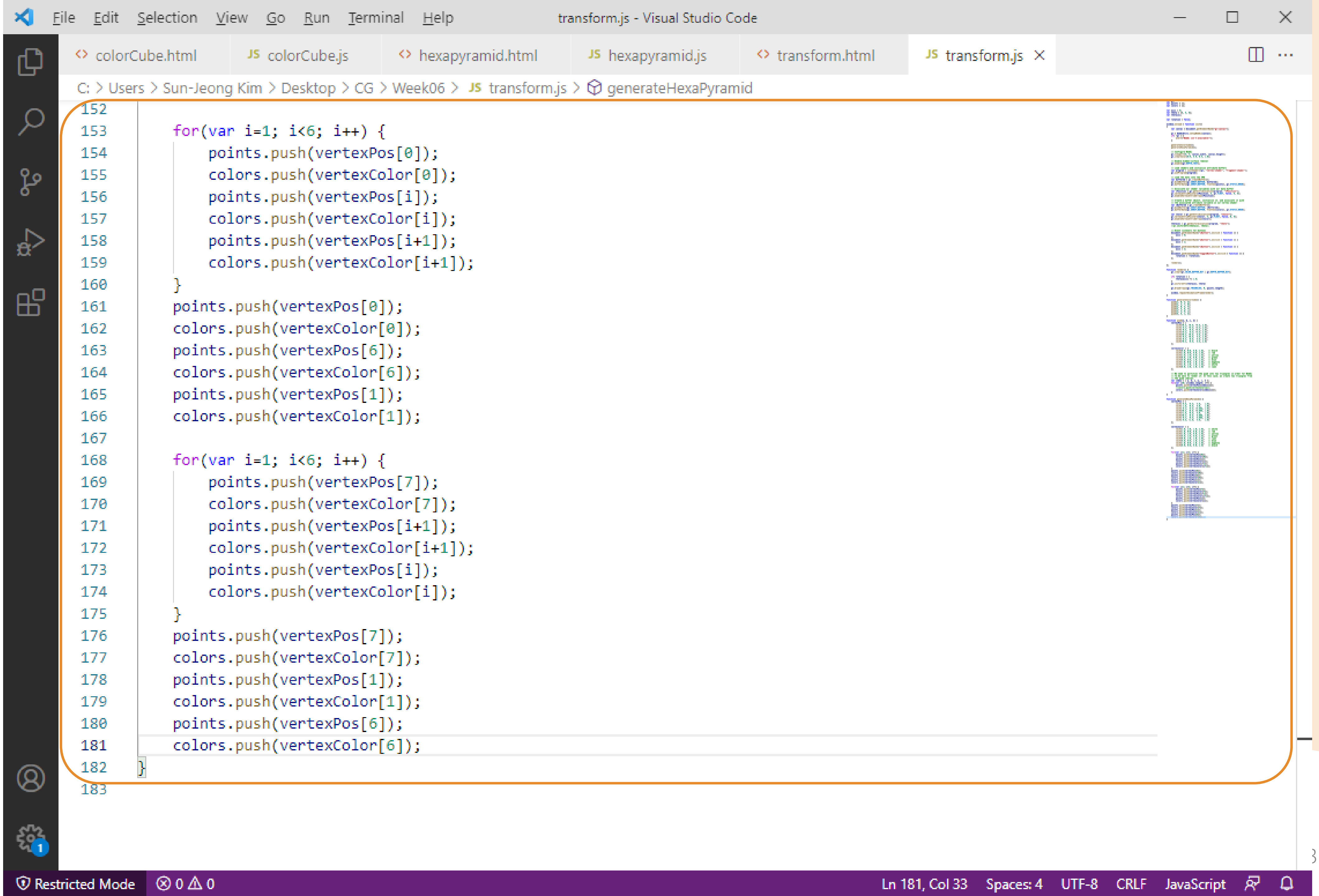
```
var gl;
var points = [];
var colors = [];

var axis = 0;
var theta = [0, 0, 0];
var thetaLoc;

var rotation = false;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    generateColorCube();
    generateHexaPyramid();

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(0.9, 0.9, 0.9, 1.0);

    // Enable hidden-surface removal
    gl.enable(gl.DEPTH_TEST);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
```


File Edit Selection View Go Run Terminal Help
transform.js - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.js > render
Restricted Mode
0 0
Ln 81, Col 1
Spaces: 4
UTF-8
CRLF
JavaScript

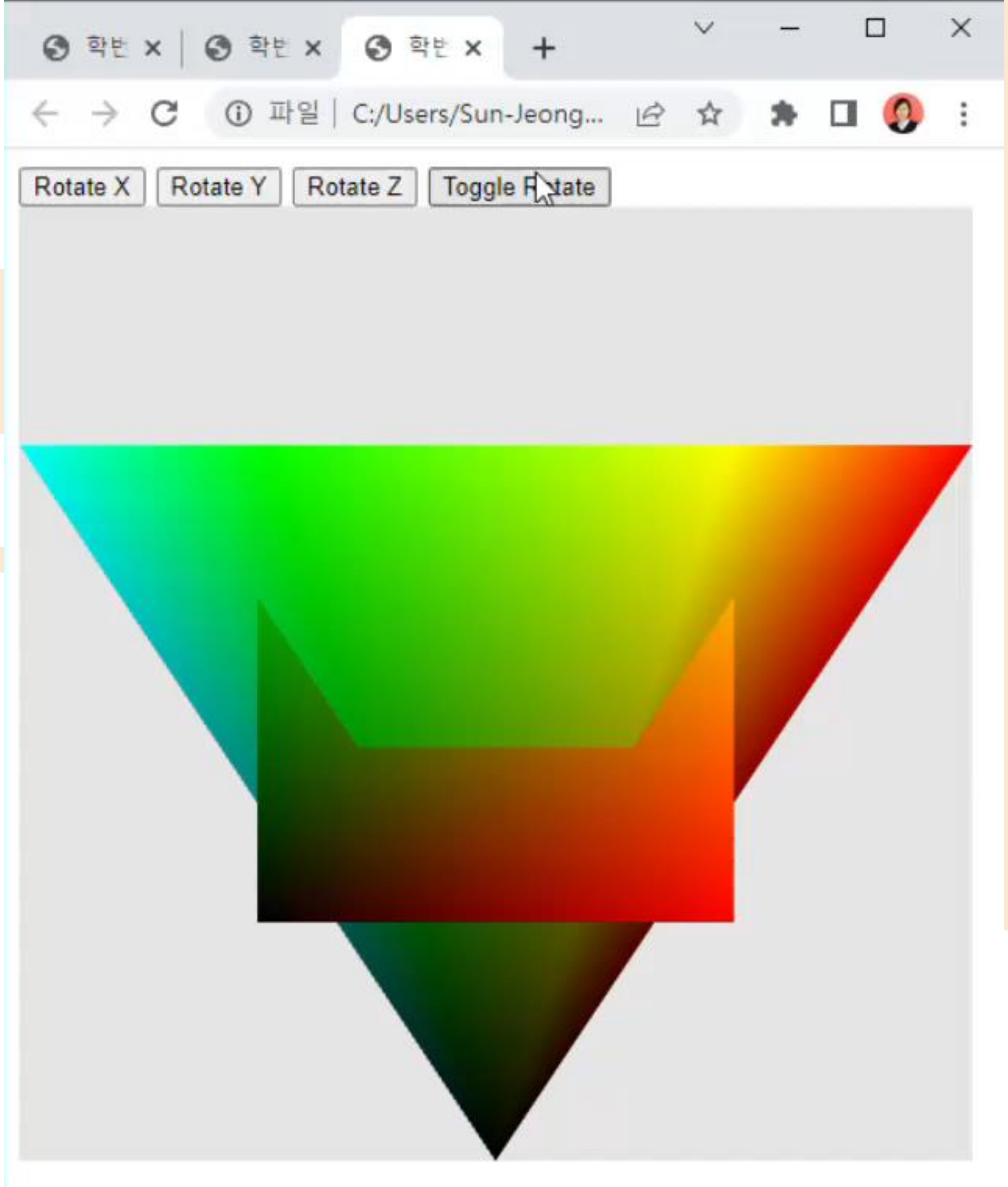
```

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    if( rotation ) {
        theta[axis] += 2.0;
    }
    gl.uniform3fv(thetaLoc, theta)

    gl.drawArrays(gl.TRIANGLES, 0, points.length);

    window.requestAnimationFrame(render);
}

function generateColorCube() {
    quad(1, 0, 3, 2);
    quad(2, 3, 7, 6);
    quad(3, 0, 4, 7);
    quad(4, 5, 6, 7);
    quad(5, 4, 0, 1);
    quad(6, 5, 1, 2);
}

function quad(a, b, c, d) {
    vertexPos = [
        vec4(-0.5, -0.5, -0.5, 1.0),
        vec4( 0.5, -0.5, -0.5, 1.0),
        vec4( 0.5,  0.5, -0.5, 1.0),
        vec4(-0.5,  0.5, -0.5, 1.0),
        vec4(-0.5, -0.5,  0.5, 1.0),
        vec4( 0.5, -0.5,  0.5, 1.0),
        vec4( 0.5,  0.5,  0.5, 1.0),
        vec4(-0.5,  0.5,  0.5, 1.0)
    ];

```


File Edit Selection View Go Run Terminal Help
transform.js - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.js > generateHexaPyramid
Restricted Mode
0 0
Ln 181, Col 33
Spaces: 4
UTF-8
CRLF
JavaScript

```

    // We need to partition the quad into two triangles in order for WebGL
    // to be able to render it. In this case, we create two triangles from
    // the quad indices.
    var index = [ a, b, c, a, c, d ];
    for(var i=0; i<index.length; i++) {
        points.push(vertexPos[index[i]]);
        //colors.push(vertexColor[a]);
        colors.push(vertexColor[index[i]]);
    }
}

function generateHexaPyramid() {
    vertexPos = [
        vec4( 0.0,  0.5,  0.0,   1.0),
        vec4( 1.0,  0.5,  0.0,   1.0),
        vec4( 0.5,  0.5, -0.866, 1.0),
        vec4(-0.5,  0.5, -0.866, 1.0),
        vec4(-1.0,  0.5,  0.0,   1.0),
        vec4(-0.5,  0.5,  0.866, 1.0),
        vec4( 0.5,  0.5,  0.866, 1.0),
        vec4( 0.0, -1.0,  0.0,   1.0)
    ];

    vertexColor = [
        vec4(1.0, 1.0, 1.0, 1.0),    // white
        vec4(1.0, 0.0, 0.0, 1.0),    // red
        vec4(1.0, 1.0, 0.0, 1.0),    // yellow
        vec4(0.0, 1.0, 0.0, 1.0),    // green
        vec4(0.0, 1.0, 1.0, 1.0),    // cyan
        vec4(0.0, 0.0, 1.0, 1.0),    // blue
        vec4(1.0, 0.0, 1.0, 1.0),    // magenta
        vec4(0.0, 0.0, 0.0, 1.0)     // black
    ];

```


File Edit Selection View Go Run Terminal Help
transform.js - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.js > generateHexaPyramid
Restricted Mode
0 0
Ln 181, Col 33
Spaces: 4
UTF-8
CRLF
JavaScript

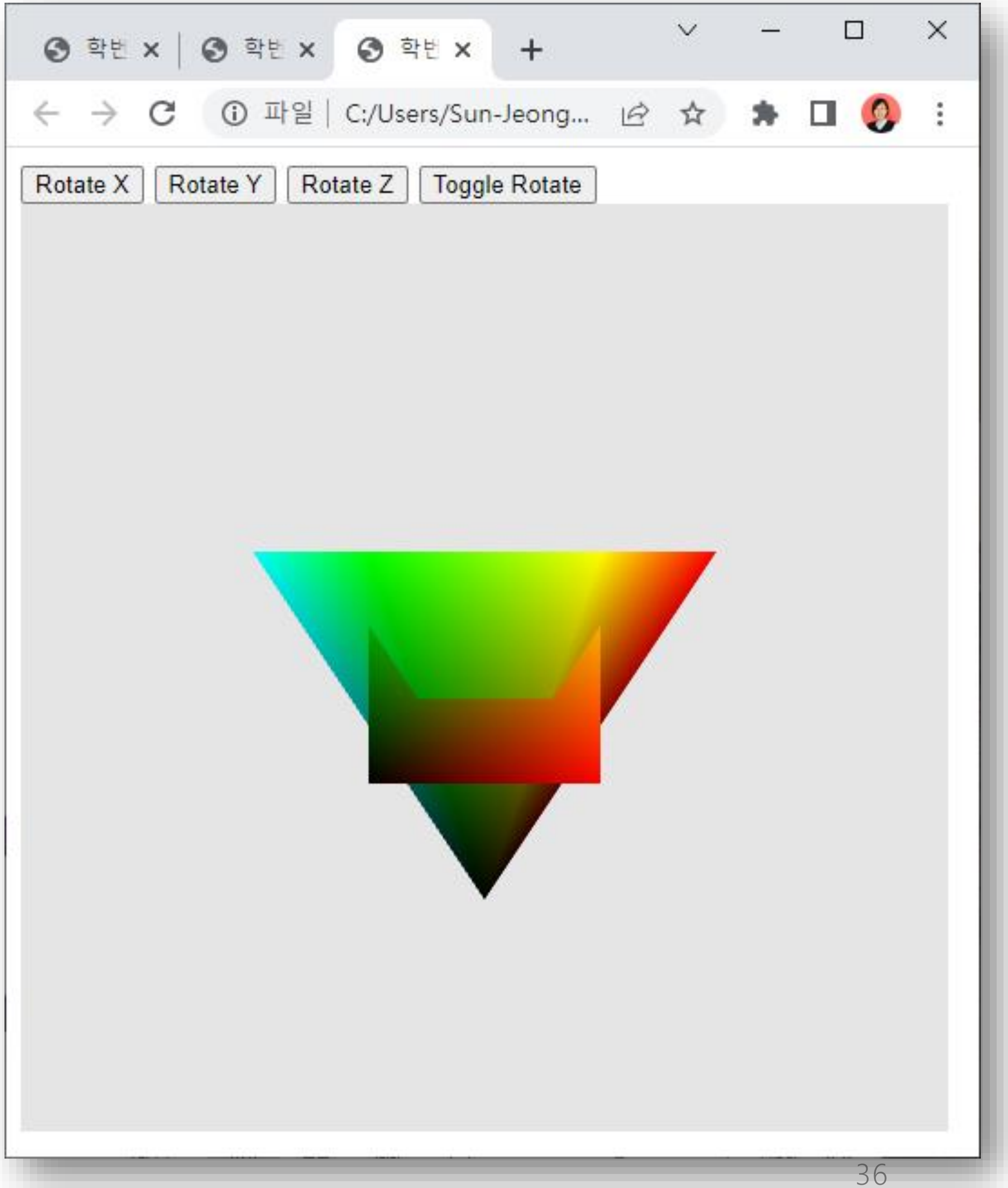
```javascript

    for(var i=1; i<6; i++) {
        points.push(vertexPos[0]);
        colors.push(vertexColor[0]);
        points.push(vertexPos[i]);
        colors.push(vertexColor[i]);
        points.push(vertexPos[i+1]);
        colors.push(vertexColor[i+1]);
    }
    points.push(vertexPos[0]);
    colors.push(vertexColor[0]);
    points.push(vertexPos[6]);
    colors.push(vertexColor[6]);
    points.push(vertexPos[1]);
    colors.push(vertexColor[1]);

    for(var i=1; i<6; i++) {
        points.push(vertexPos[7]);
        colors.push(vertexColor[7]);
        points.push(vertexPos[i+1]);
        colors.push(vertexColor[i+1]);
        points.push(vertexPos[i]);
        colors.push(vertexColor[i]);
    }
    points.push(vertexPos[7]);
    colors.push(vertexColor[7]);
    points.push(vertexPos[1]);
    colors.push(vertexColor[1]);
    points.push(vertexPos[6]);
    colors.push(vertexColor[6]);
}

```

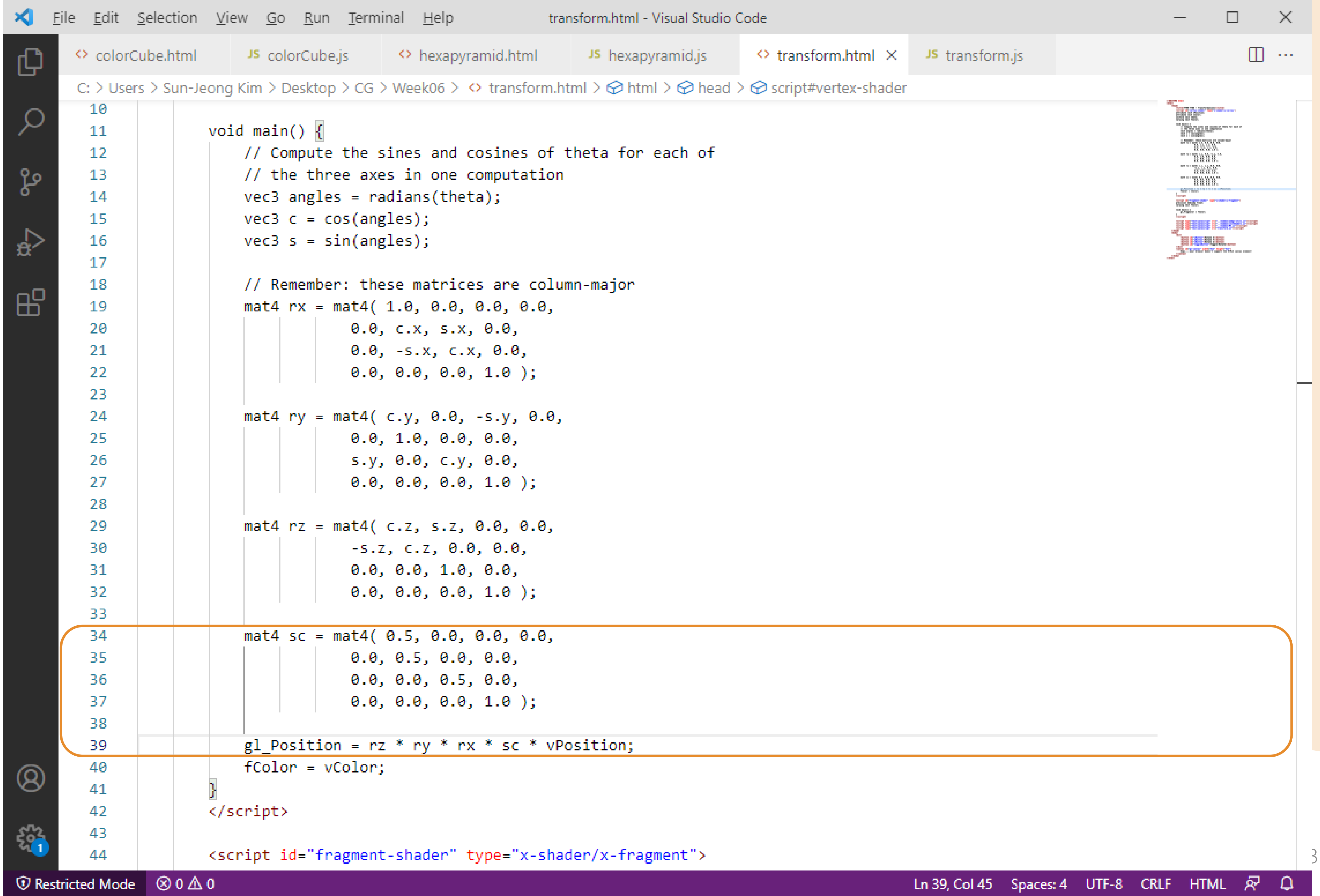


학번
학번
학번
파일
C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

연습 문제 (2)

- 오브젝트들의 크기를 반으로 줄이시오.

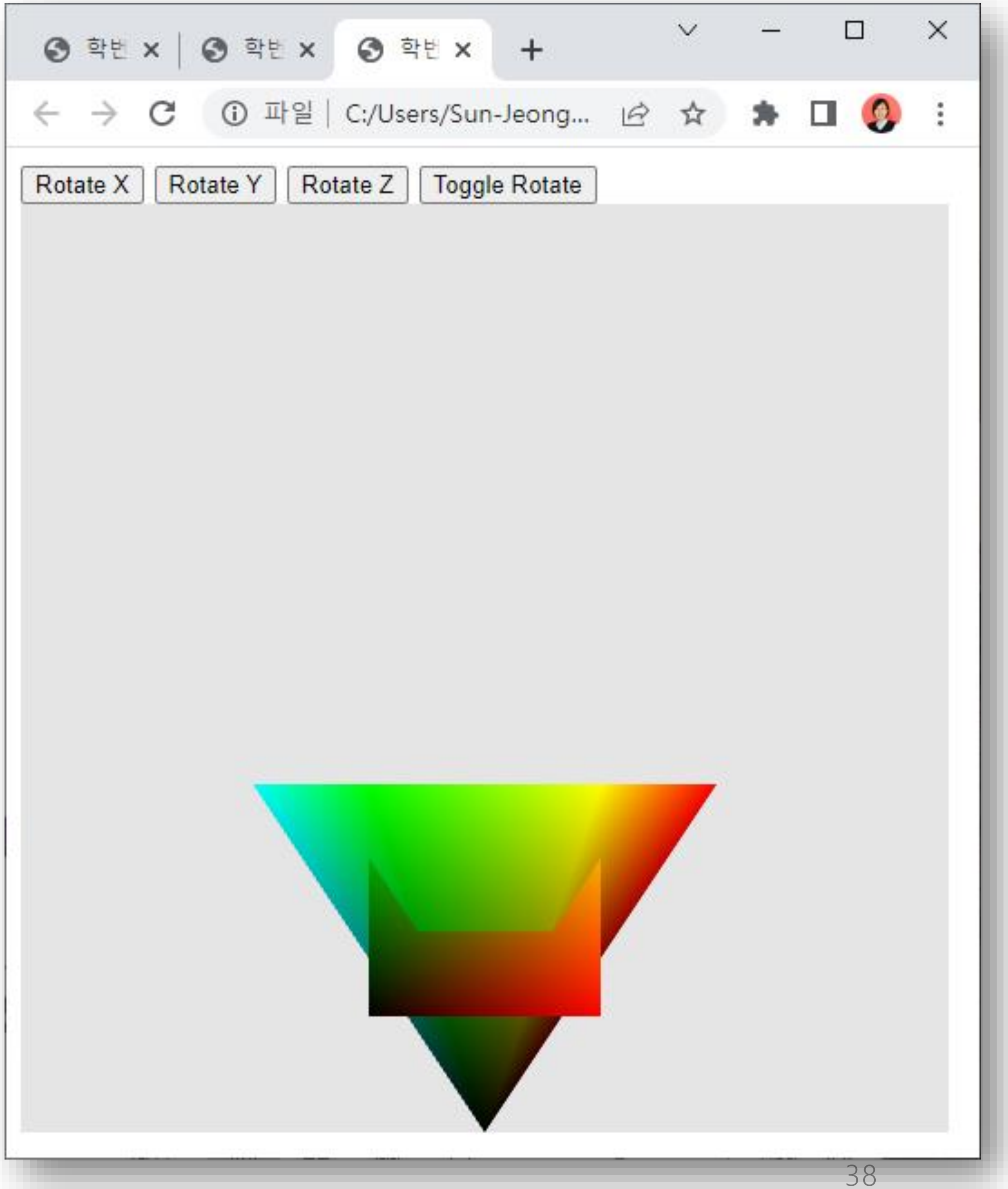


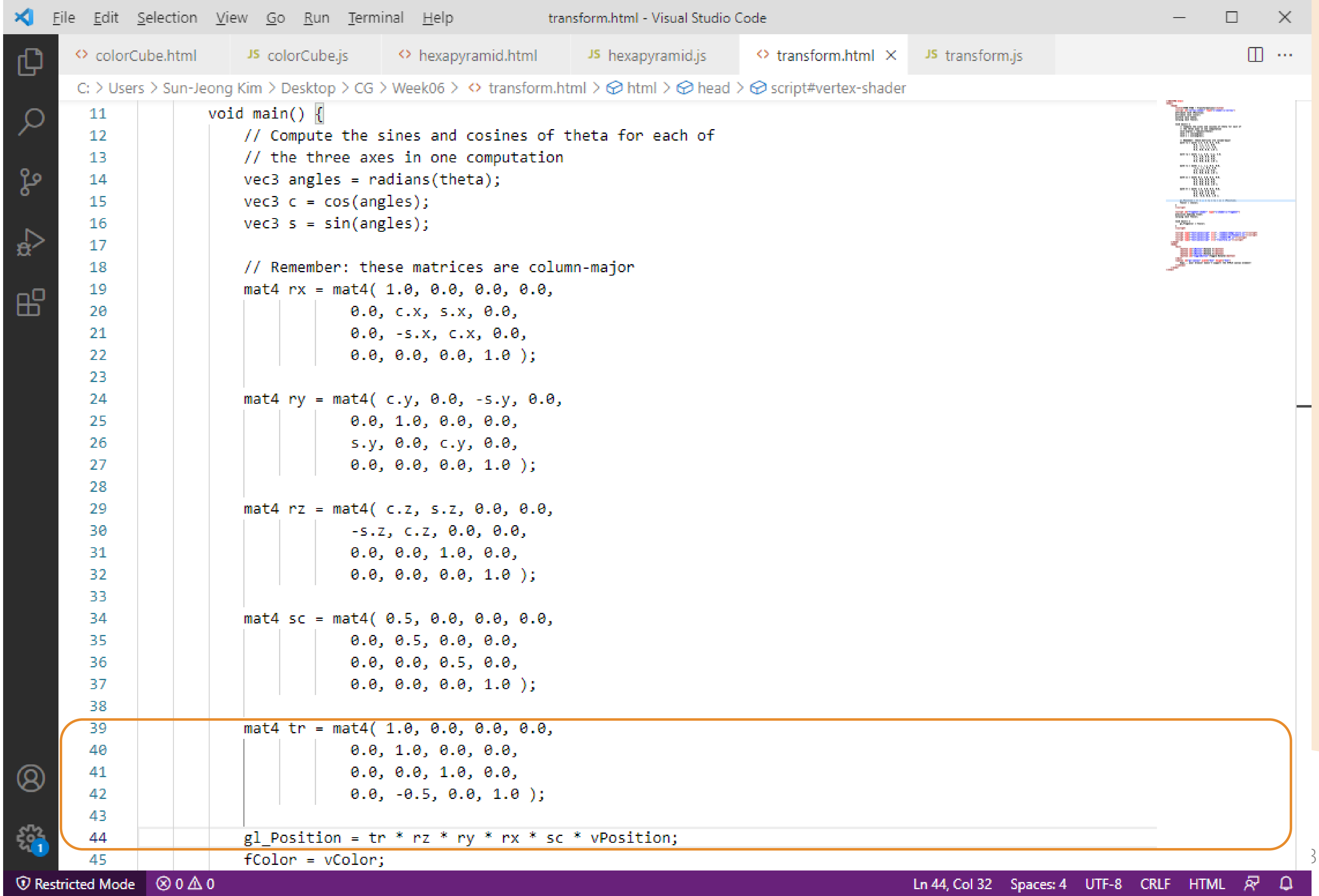

File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script#vertex-shader
Restricted Mode
0 0
Ln 39, Col 45
Spaces: 4
UTF-8
CRLF
HTML

```

        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, c.x, s.x, 0.0,
                        0.0, -s.x, c.x, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        s.y, 0.0, c.y, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                        -s.z, c.z, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
                        0.0, 0.5, 0.0, 0.0,
                        0.0, 0.0, 0.5, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            gl_Position = rz * ry * rx * sc * vPosition;
            fColor = vColor;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
```

연습 문제 (3)

- 오브젝트들을 y축 방향으로 -0.5 만큼 이동 시키시오.

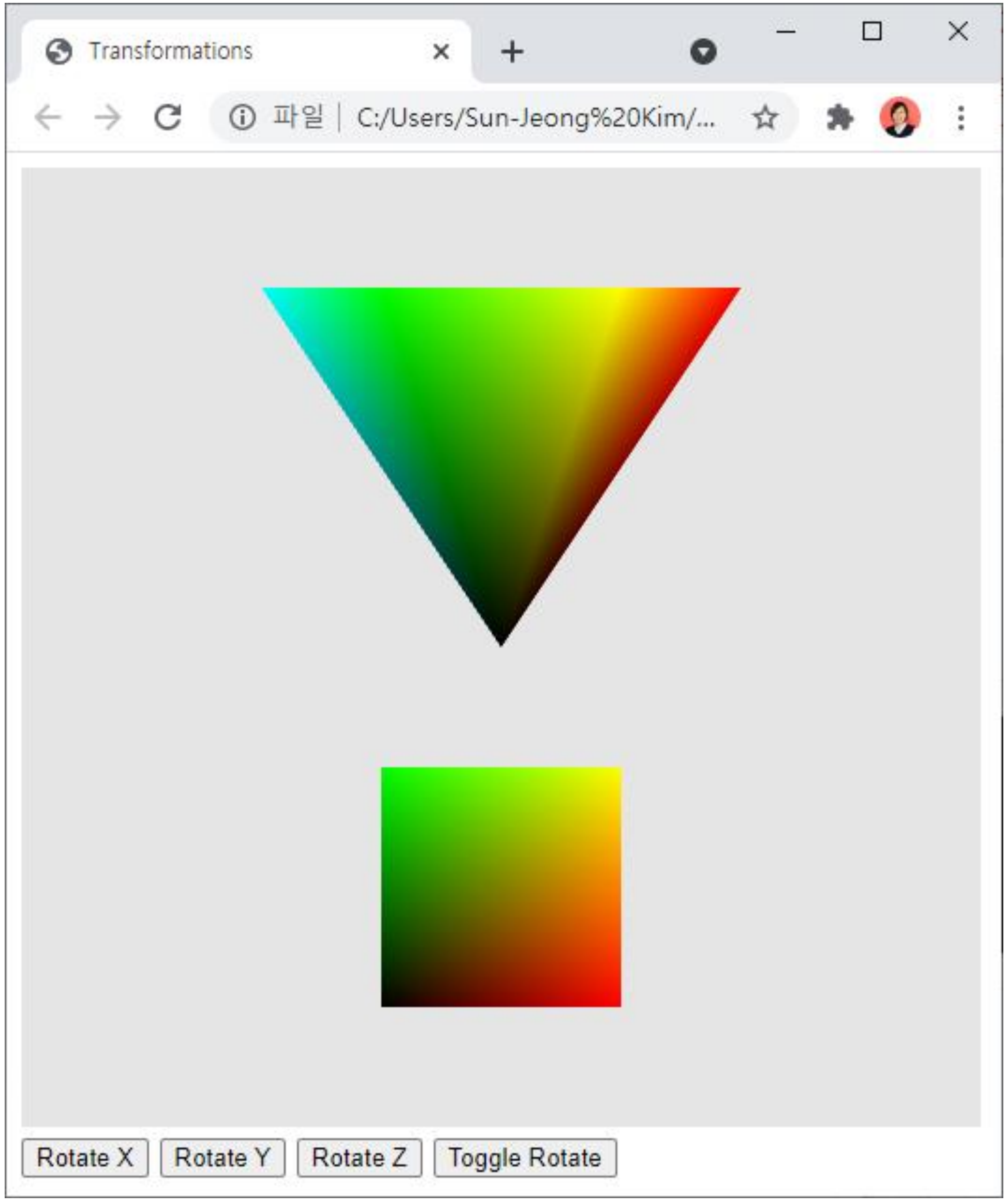


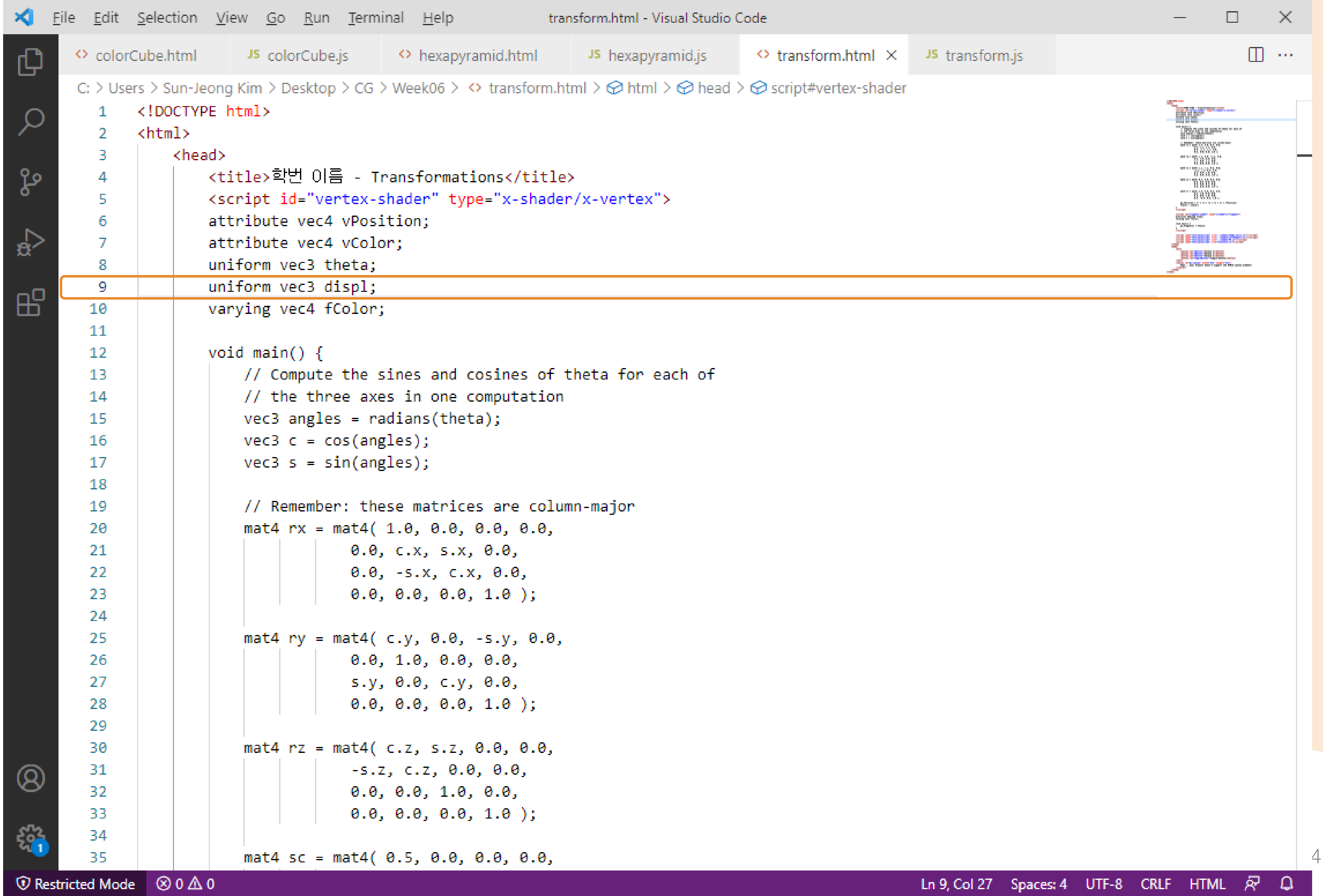


```
        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, c.x, s.x, 0.0,
                        0.0, -s.x, c.x, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        s.y, 0.0, c.y, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                        -s.z, c.z, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
                        0.0, 0.5, 0.0, 0.0,
                        0.0, 0.0, 0.5, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 tr = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        0.0, -0.5, 0.0, 1.0 );

            gl_Position = tr * rz * ry * rx * sc * vPosition;
            fColor = vColor;
```

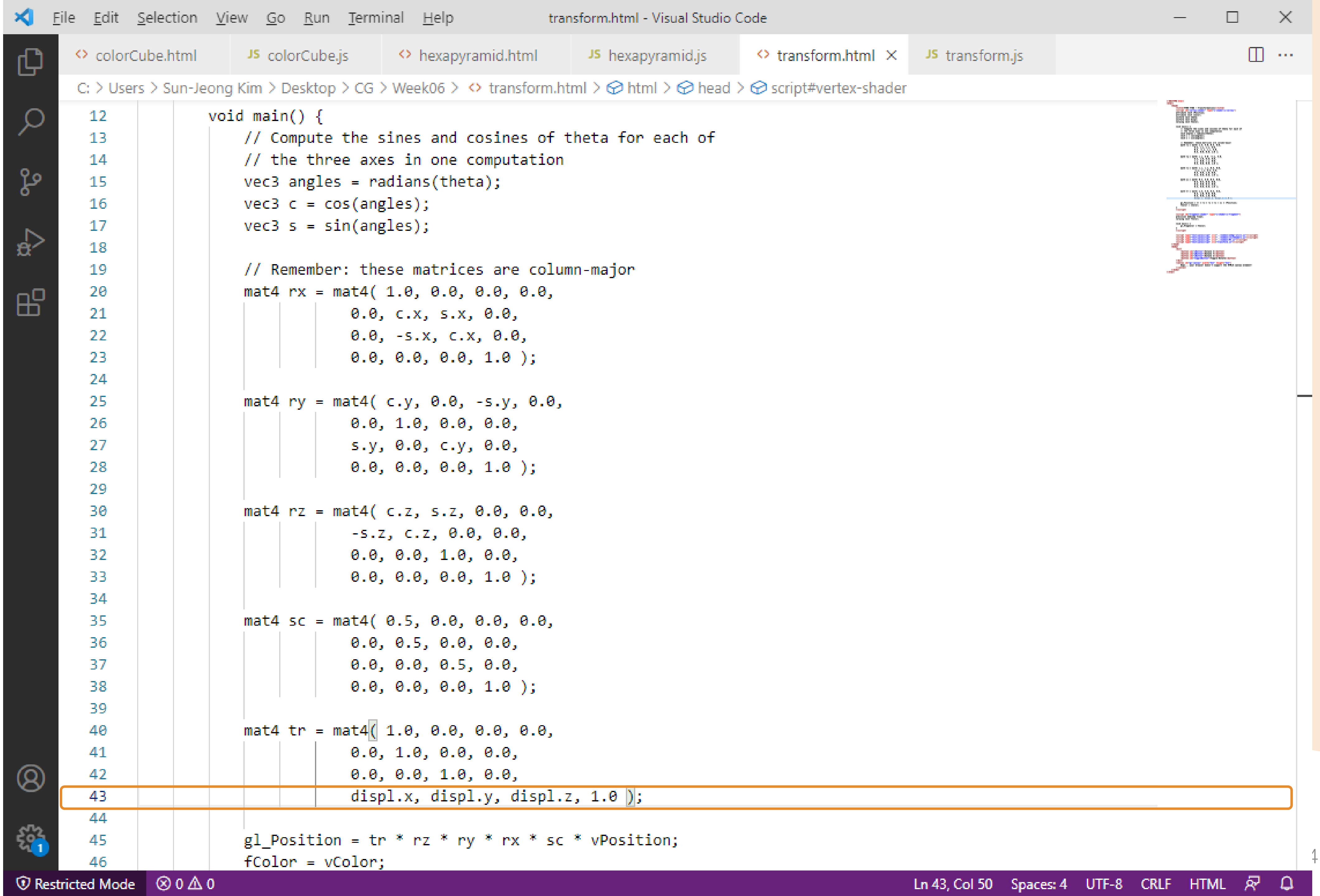
연습 문제 (4)

- Hexagonal Pyramid를 y축 방향으로 0.5만큼 이동 시키시오.

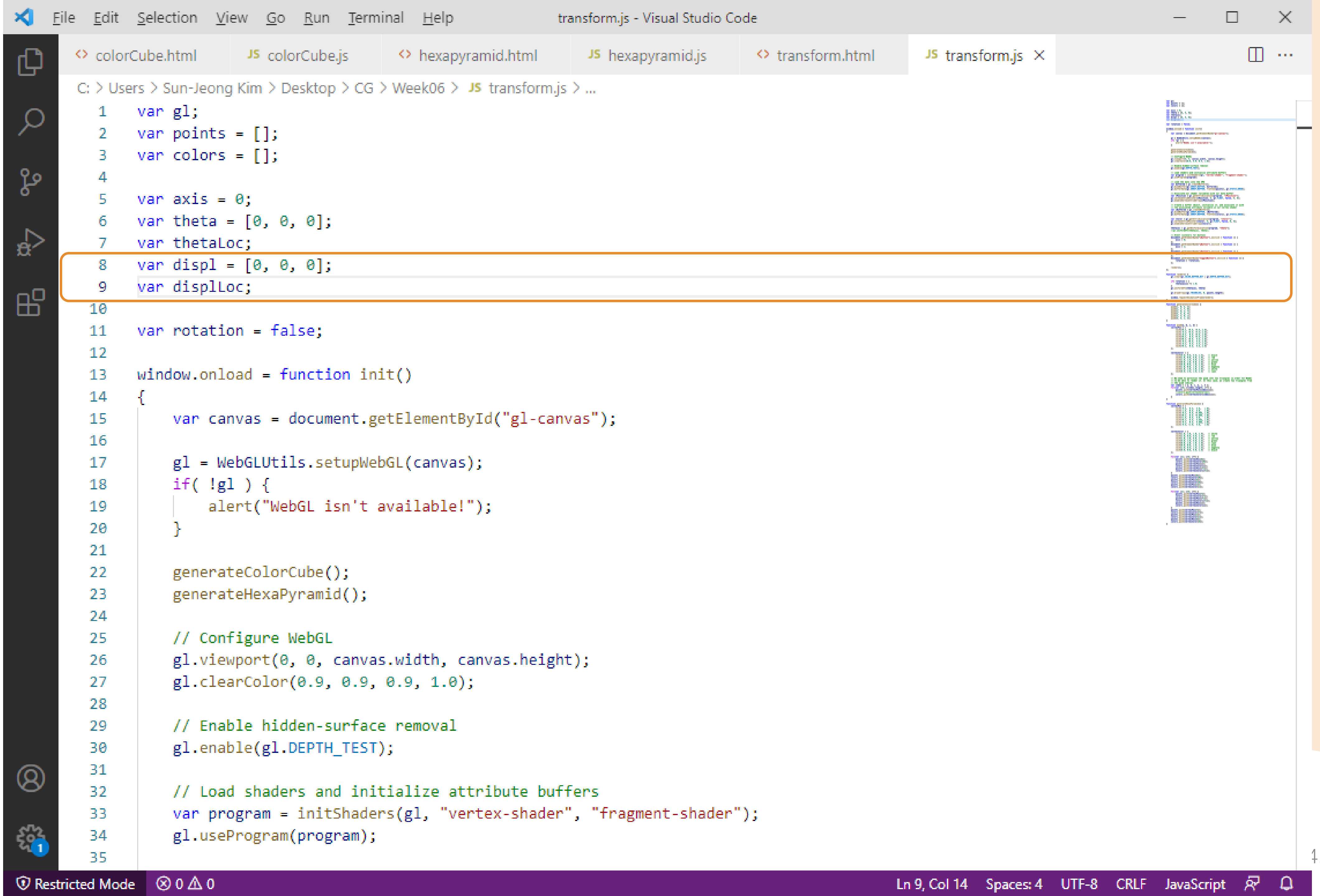



File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script#vertex-shader
Restricted Mode
0 0
Ln 9, Col 27 Spaces: 4 UTF-8 CRLF HTML

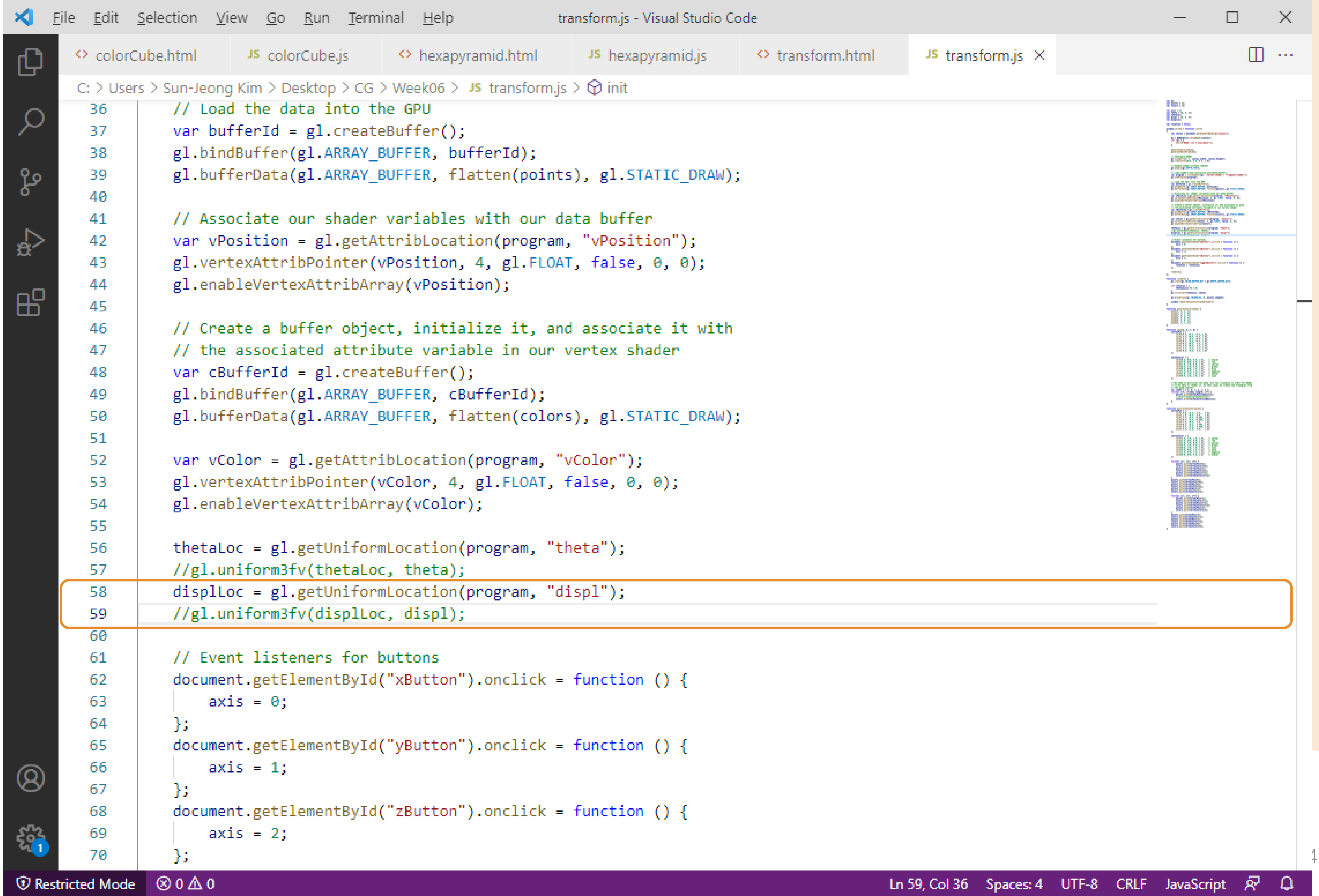
```
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Transformations</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform vec3 theta;
        uniform vec3 displ;
        varying vec4 fColor;

        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                            0.0, c.x, s.x, 0.0,
                            0.0, -s.x, c.x, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                            0.0, 1.0, 0.0, 0.0,
                            s.y, 0.0, c.y, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                            -s.z, c.z, 0.0, 0.0,
                            0.0, 0.0, 1.0, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
```


File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script#vertex-shader
Restricted Mode
0 0
Ln 43, Col 50 Spaces: 4 UTF-8 CRLF HTML

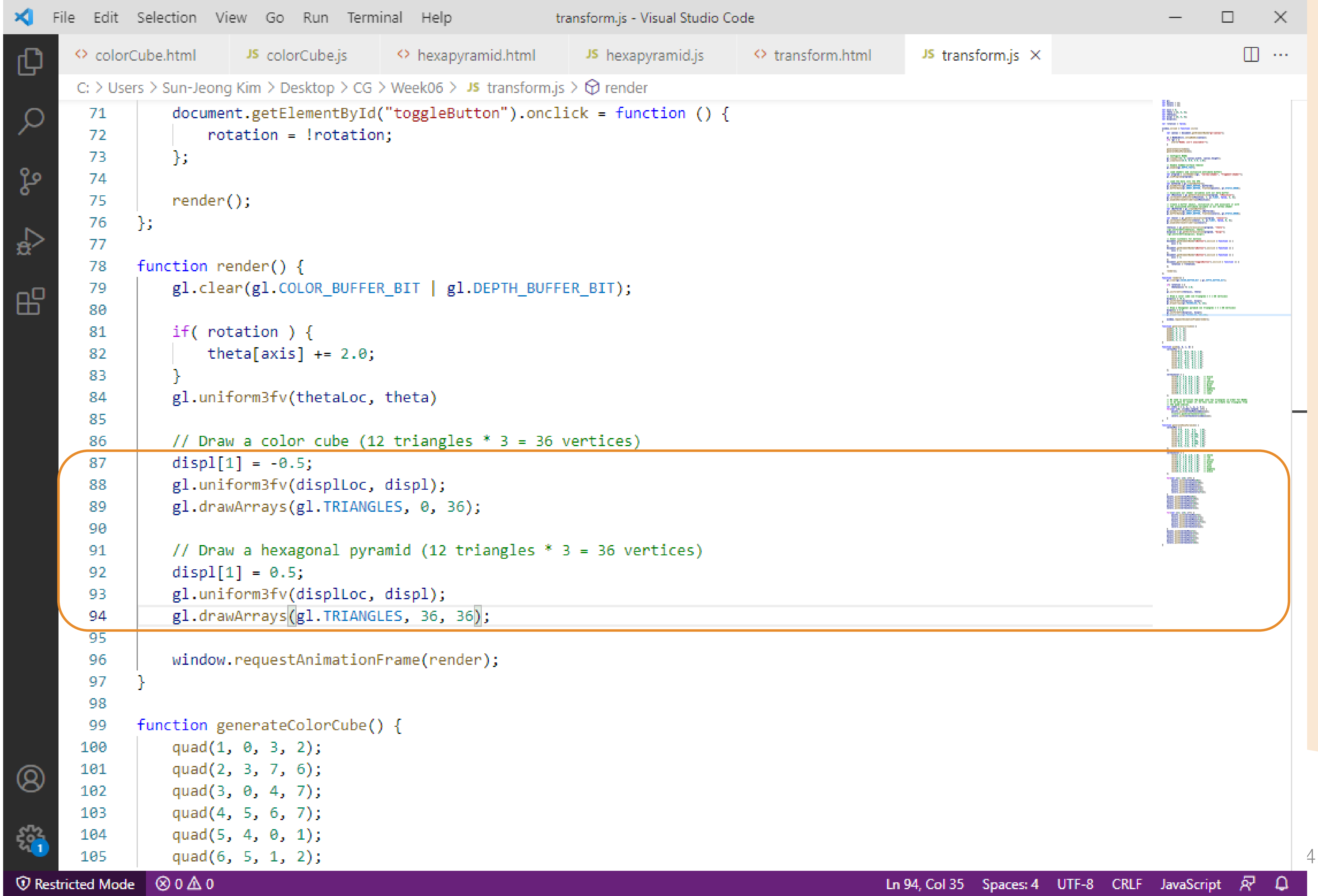
```
        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                            0.0, c.x, s.x, 0.0,
                            0.0, -s.x, c.x, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                            0.0, 1.0, 0.0, 0.0,
                            s.y, 0.0, c.y, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                            -s.z, c.z, 0.0, 0.0,
                            0.0, 0.0, 1.0, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
                            0.0, 0.5, 0.0, 0.0,
                            0.0, 0.0, 0.5, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 tr = mat4( 1.0, 0.0, 0.0, 0.0,
                            0.0, 1.0, 0.0, 0.0,
                            0.0, 0.0, 1.0, 0.0,
                            displ.x, displ.y, displ.z, 1.0 );

            gl_Position = tr * rz * ry * rx * sc * vPosition;
            fColor = vColor;
```



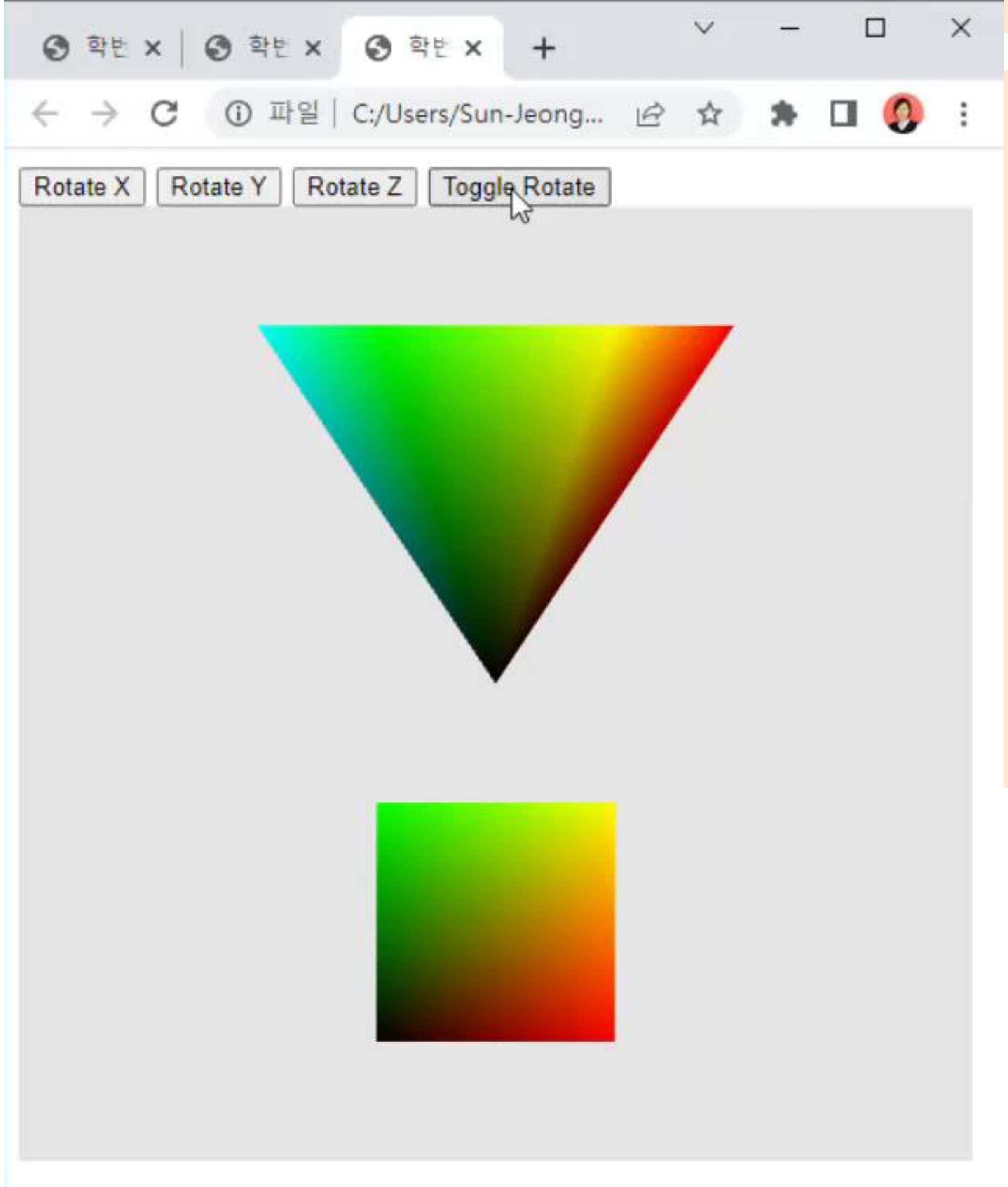
```
var gl;
var points = [];
var colors = [];

var axis = 0;
var theta = [0, 0, 0];
var thetaLoc;
var displ = [0, 0, 0];
var displLoc;

var rotation = false;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    generateColorCube();
    generateHexaPyramid();

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(0.9, 0.9, 0.9, 1.0);

    // Enable hidden-surface removal
    gl.enable(gl.DEPTH_TEST);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

```

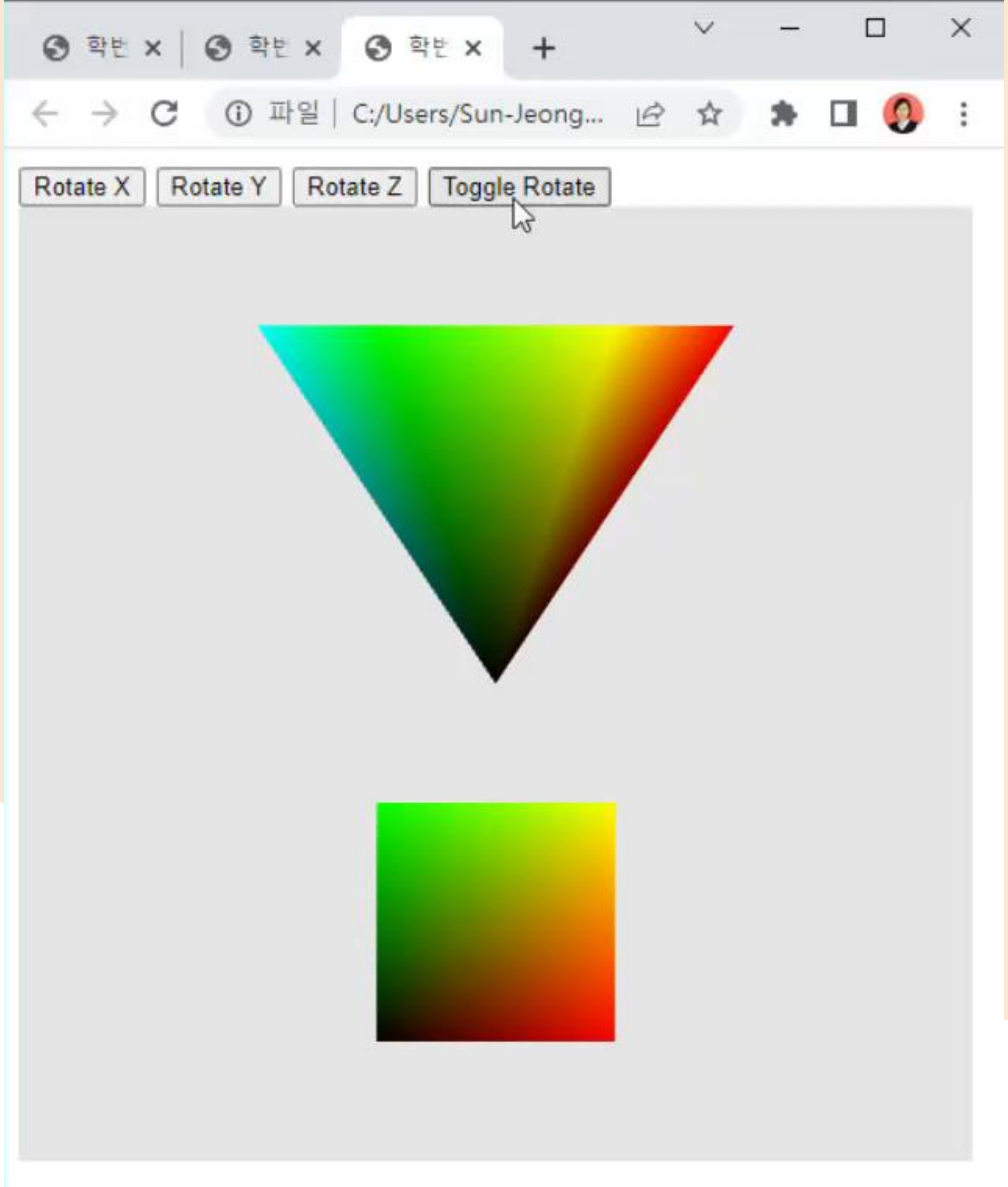

File Edit Selection View Go Run Terminal Help
transform.js - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.js > init
Restricted Mode
0 0
Ln 59, Col 36
Spaces: 4
UTF-8
CRLF
JavaScript

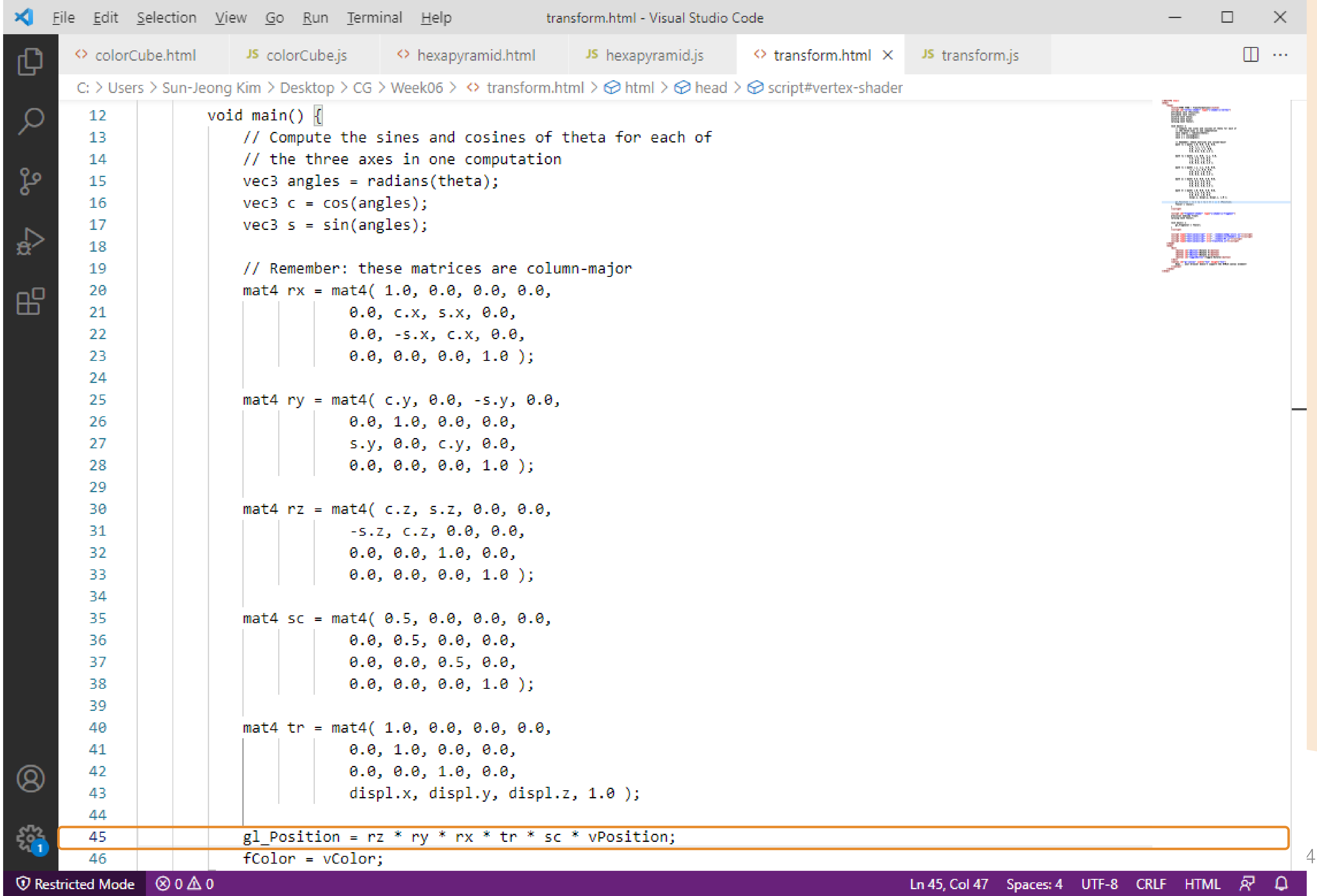
```
    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    // Create a buffer object, initialize it, and associate it with
    // the associated attribute variable in our vertex shader
    var cBufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, cBufferId);
    gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);

    var vColor = gl.getAttribLocation(program, "vColor");
    gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vColor);

    thetaLoc = gl.getUniformLocation(program, "theta");
    //gl.uniform3fv(thetaLoc, theta);
    displLoc = gl.getUniformLocation(program, "displ");
    //gl.uniform3fv(displLoc, displ);

    // Event listeners for buttons
    document.getElementById("xButton").onclick = function () {
        axis = 0;
    };
    document.getElementById("yButton").onclick = function () {
        axis = 1;
    };
    document.getElementById("zButton").onclick = function () {
        axis = 2;
    };
```


File Edit Selection View Go Run Terminal Help
transform.js - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.js > render

```
    document.getElementById("toggleButton").onclick = function () {
        rotation = !rotation;
    };

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    if( rotation ) {
        theta[axis] += 2.0;
    }
    gl.uniform3fv(thetaLoc, theta)

    // Draw a color cube (12 triangles * 3 = 36 vertices)
    displ[1] = -0.5;
    gl.uniform3fv(displLoc, displ);
    gl.drawArrays(gl.TRIANGLES, 0, 36);

    // Draw a hexagonal pyramid (12 triangles * 3 = 36 vertices)
    displ[1] = 0.5;
    gl.uniform3fv(displLoc, displ);
    gl.drawArrays(gl.TRIANGLES, 36, 36);

    window.requestAnimationFrame(render);
}

function generateColorCube() {
    quad(1, 0, 3, 2);
    quad(2, 3, 7, 6);
    quad(3, 0, 4, 7);
    quad(4, 5, 6, 7);
    quad(5, 4, 0, 1);
    quad(6, 5, 1, 2);
```


학번
학번
학번
파일
C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

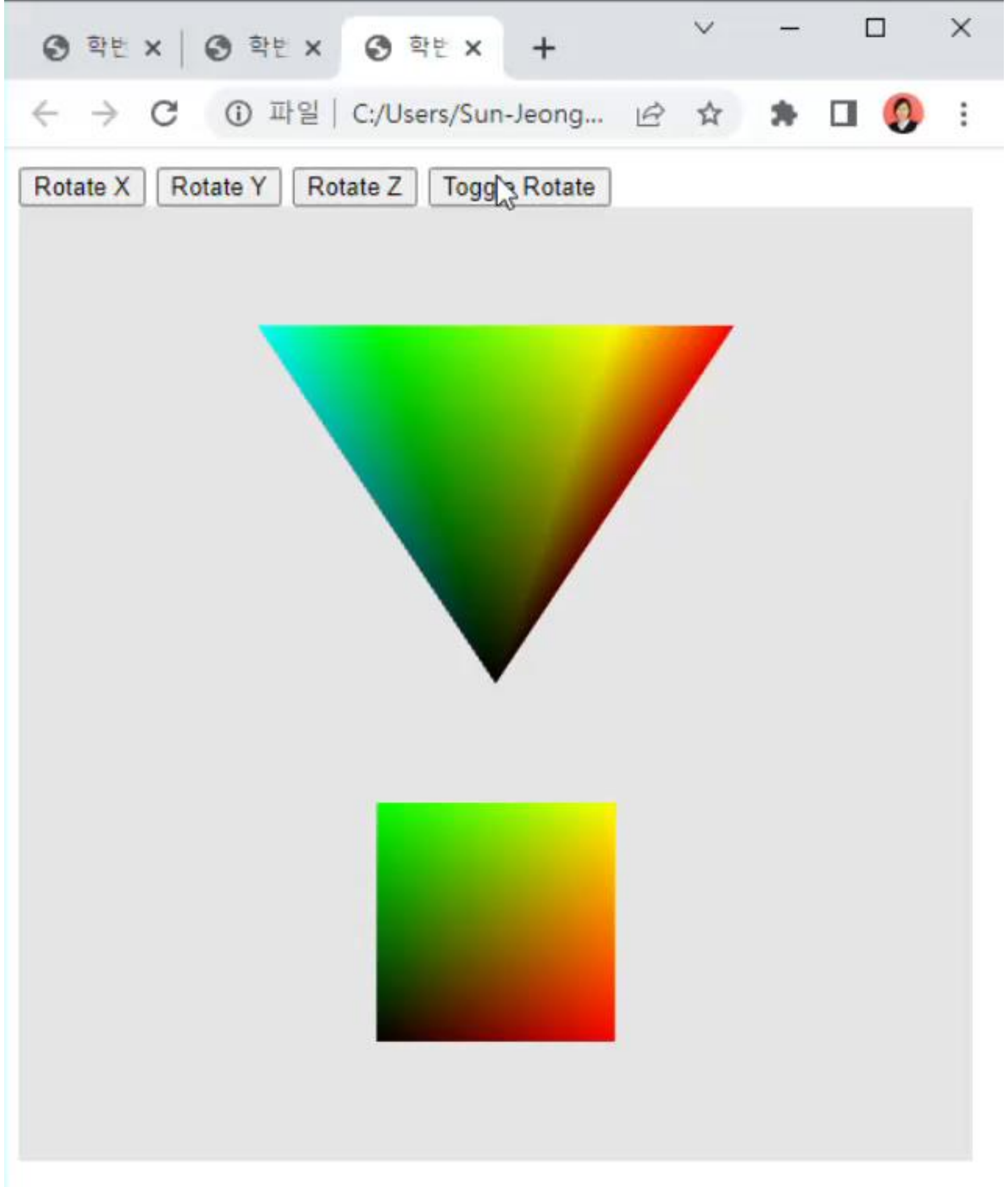

학번
학번
학번
파일
C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate


File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script#vertex-shader
Restricted Mode
0 0
Ln 45, Col 47 Spaces: 4 UTF-8 CRLF HTML

```
    void main() {
        // Compute the sines and cosines of theta for each of
        // the three axes in one computation
        vec3 angles = radians(theta);
        vec3 c = cos(angles);
        vec3 s = sin(angles);

        // Remember: these matrices are column-major
        mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, c.x, s.x, 0.0,
                        0.0, -s.x, c.x, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

        mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        s.y, 0.0, c.y, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

        mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                        -s.z, c.z, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

        mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
                        0.0, 0.5, 0.0, 0.0,
                        0.0, 0.0, 0.5, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

        mat4 tr = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        displ.x, displ.y, displ.z, 1.0 );

        gl_Position = rz * ry * rx * tr * sc * vPosition;
        fColor = vColor;
```

연습 문제 (5)

- 두 오브젝트가 서로 반대 방향으로 회전하도록 구현하시오.

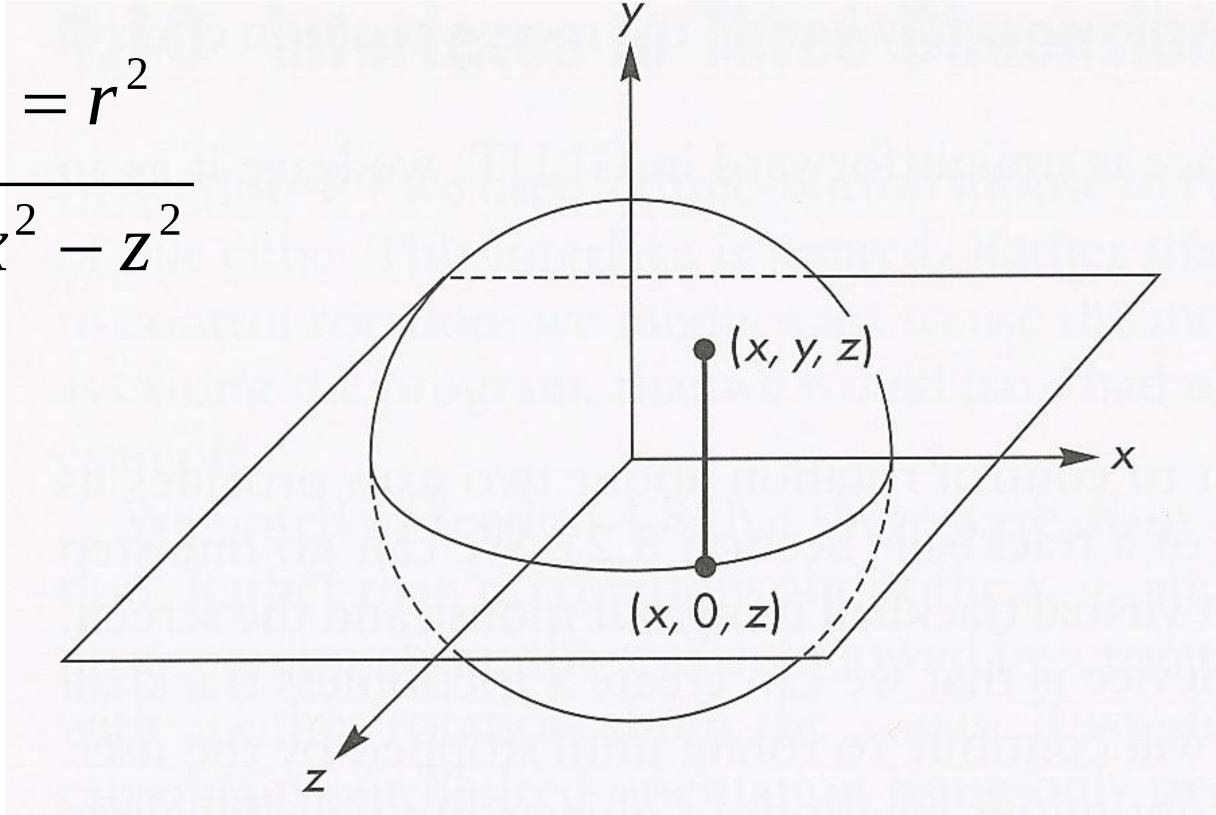


Rotation with a Virtual Trackball (1)

- Projection of the trackball position to the plane

$$x^2 + y^2 + z^2 = r^2$$

$$\therefore y = \sqrt{r^2 - x^2 - z^2}$$



Rotation with a Virtual Trackball (2)

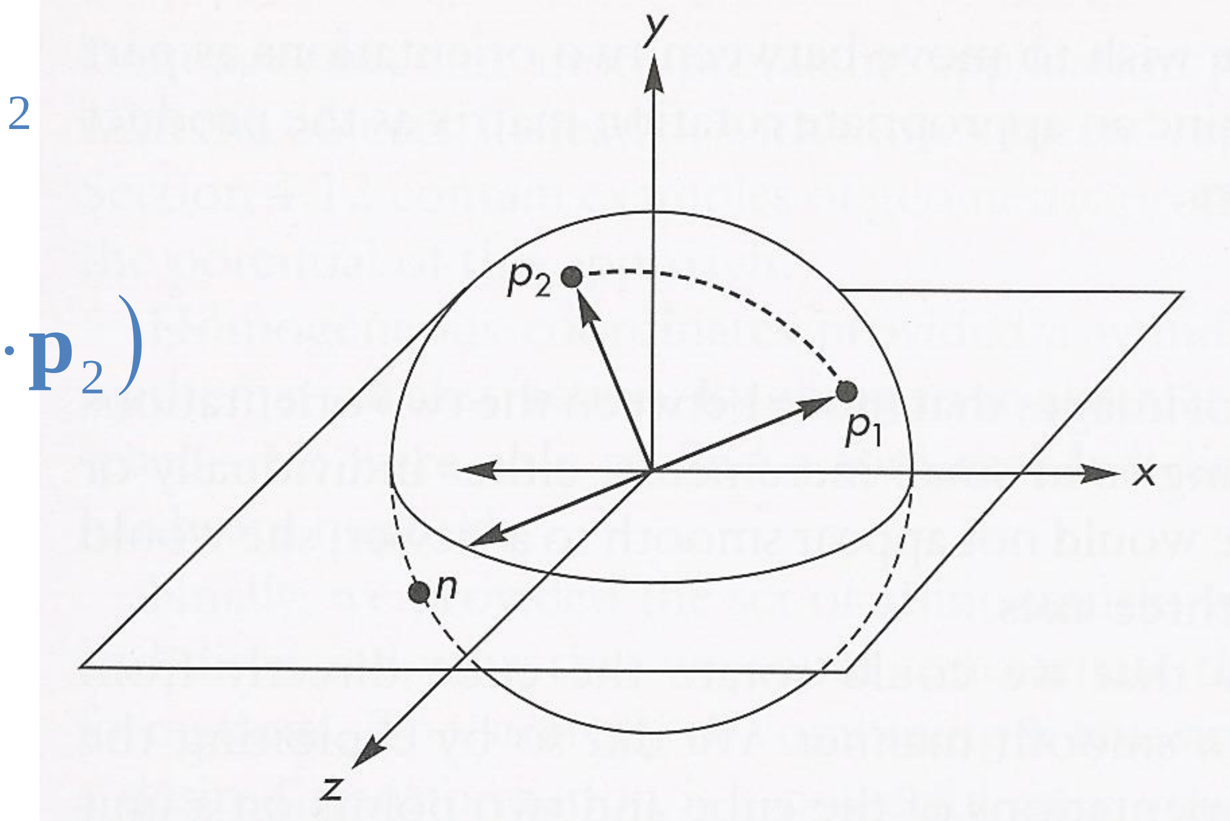
- Determination of the orientation of a plane

$$\mathbf{n} = \mathbf{p}_1 \times \mathbf{p}_2$$

- Rotation angle

$$\theta = \cos^{-1}(\mathbf{p}_1 \cdot \mathbf{p}_2)$$

➔ Quaternions



Rotations with Quaternions (1)

- Rotation about an arbitrary axis
  - Setting up a unit quaternion (**u**: unit vector)

$$s = \cos\frac{\theta}{2}, \quad \mathbf{v} = \mathbf{u}\sin\frac{\theta}{2} = (a, b, c)$$

  - Representing any point position **P** in quaternion notation (**p** = (*x*, *y*, *z*))

$$\mathbf{P} = (0, \ \mathbf{p})$$

  - Carrying out with the quaternion operation (q^{-1}=(s, –**v**))

$$\mathbf{P}' = q\mathbf{P}q^{-1}$$

  - Producing the new quaternion

$$\mathbf{P}' = (0, \ \mathbf{p}')$$

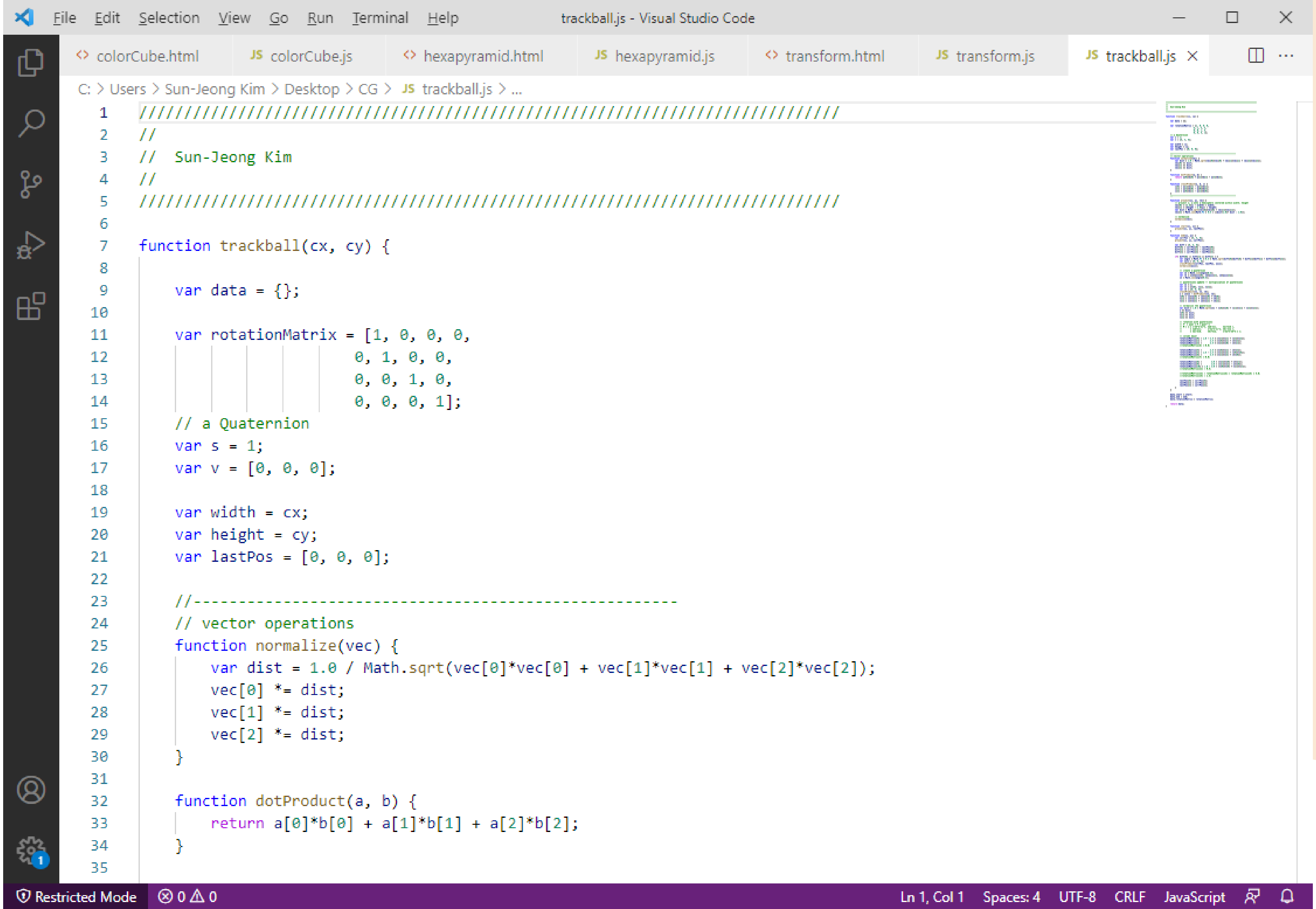
$$\mathbf{p}' = s^2\mathbf{p} + \mathbf{v}(\mathbf{p}\cdot\mathbf{v}) + 2s(\mathbf{v}\times\mathbf{p}) + \mathbf{v}\times(\mathbf{v}\times\mathbf{p})$$

Rotations with Quaternions (2)

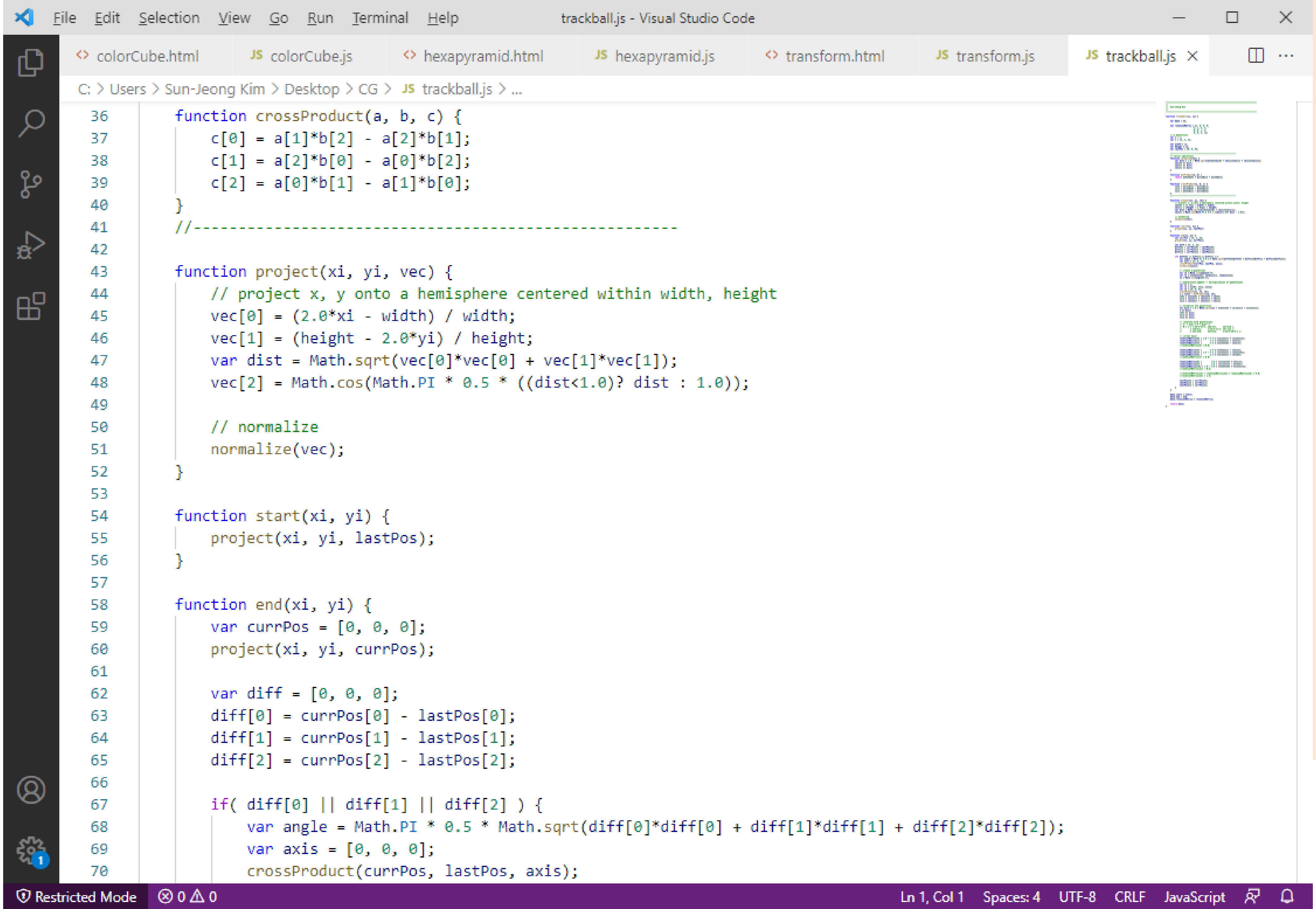
- Obtaining the rotation matrix by quaternion multiplication

$$\mathbf{M}_R(\theta) = \begin{bmatrix} 1-2b^2-2c^2 & 2ab-2sc & 2ac+2sb \\ 2ab+2sc & 1-2a^2-2c^2 & 2bc-2sa \\ 2ac-2sb & 2bc+2sa & 1-2a^2-2b^2 \end{bmatrix}$$

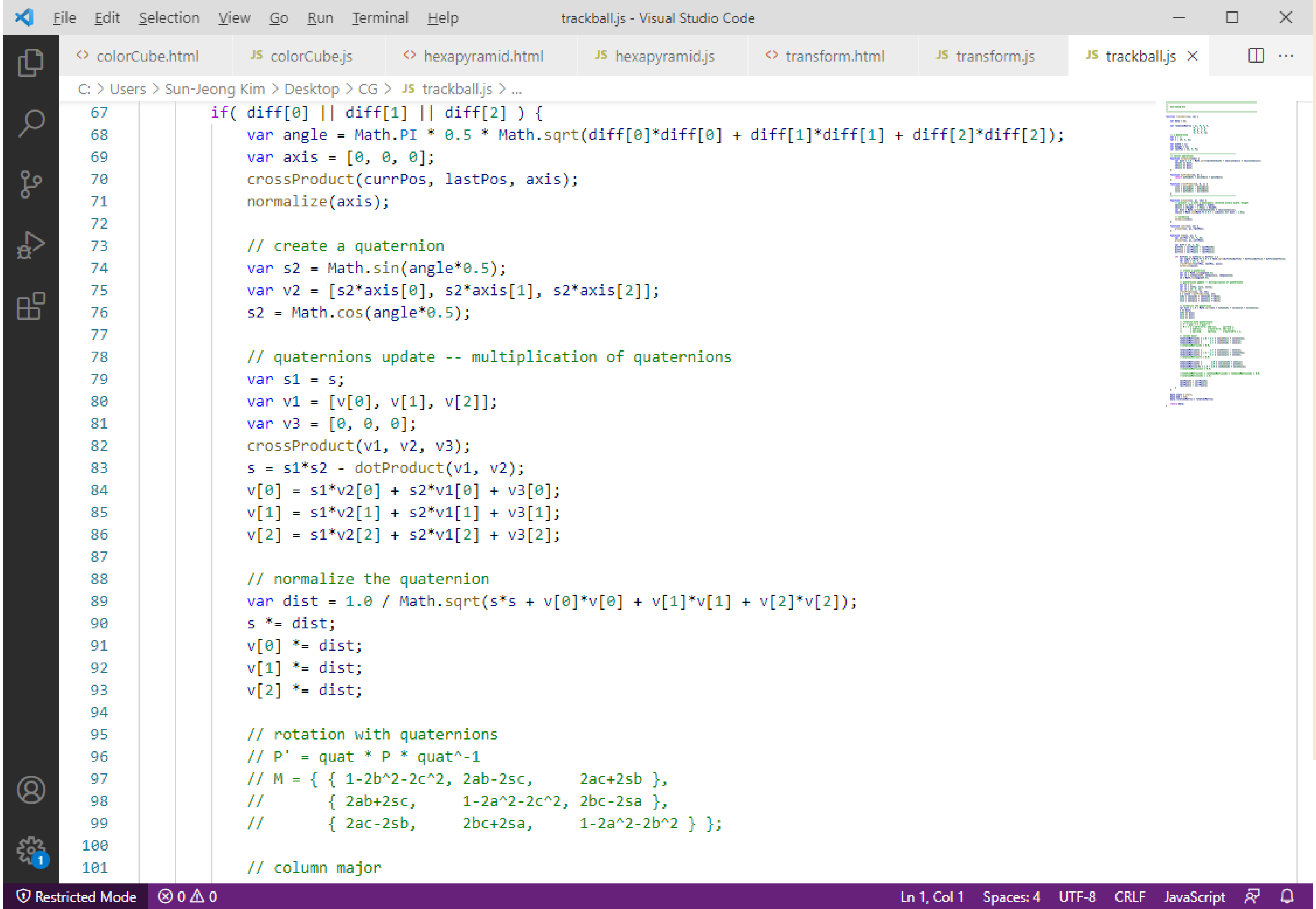
$$= \mathbf{R}_x(-\theta_x)\mathbf{R}_y(-\theta_y)\mathbf{R}_z(\theta)\mathbf{R}_y(\theta_y)\mathbf{R}_x(\theta_x)$$



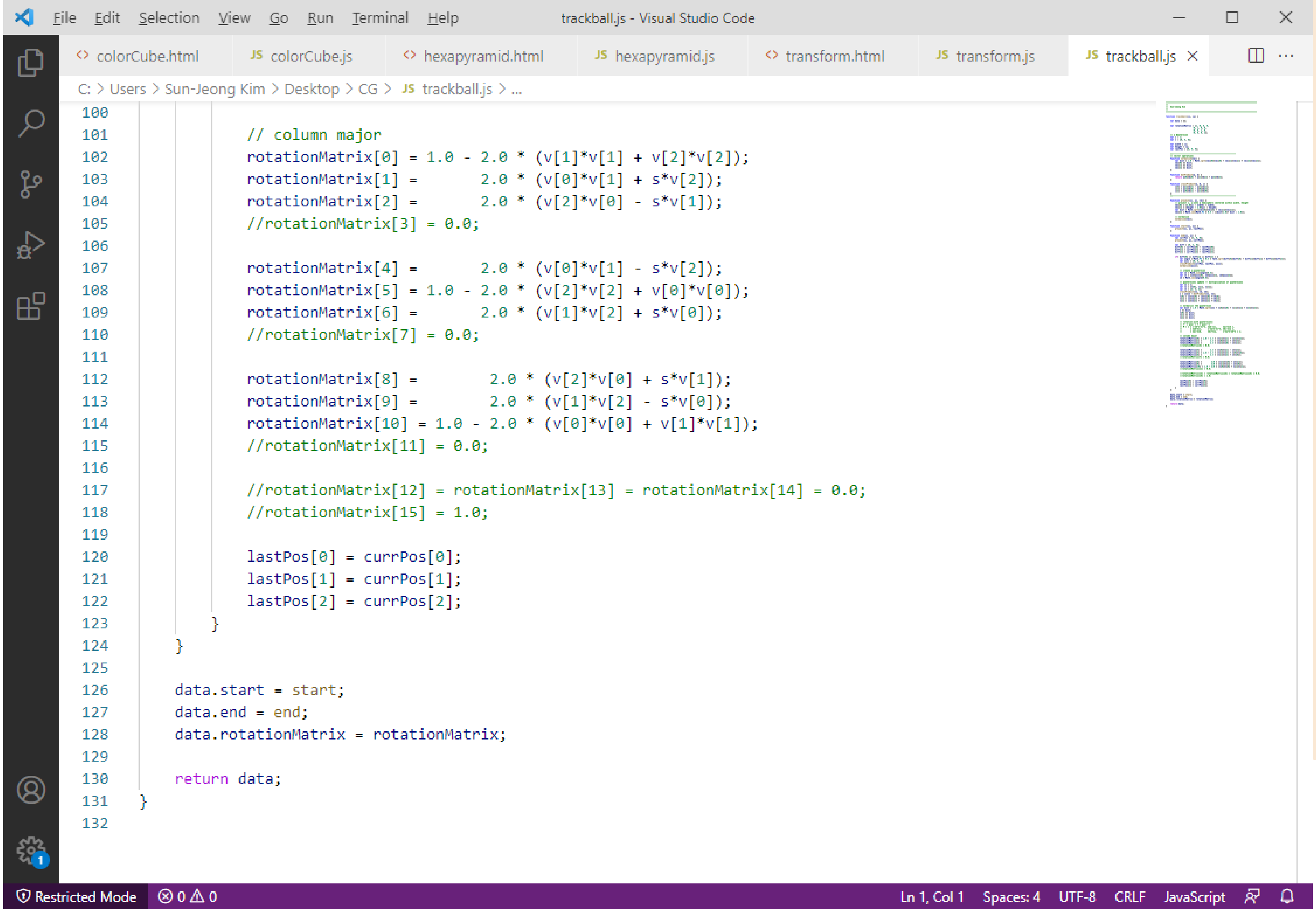
```javascript
//////////////////////////////////////////////////////////////////////////////
//
//  Sun-Jeong Kim
//
//////////////////////////////////////////////////////////////////////////////

function trackball(cx, cy) {

    var data = {};

    var rotationMatrix = [1, 0, 0, 0,
                          0, 1, 0, 0,
                          0, 0, 1, 0,
                          0, 0, 0, 1];
    // a Quaternion
    var s = 1;
    var v = [0, 0, 0];

    var width = cx;
    var height = cy;
    var lastPos = [0, 0, 0];

    //---------------------------------------------------------
    // vector operations
    function normalize(vec) {
        var dist = 1.0 / Math.sqrt(vec[0]*vec[0] + vec[1]*vec[1] + vec[2]*vec[2]);
        vec[0] *= dist;
        vec[1] *= dist;
        vec[2] *= dist;
    }

    function dotProduct(a, b) {
        return a[0]*b[0] + a[1]*b[1] + a[2]*b[2];
    }

```



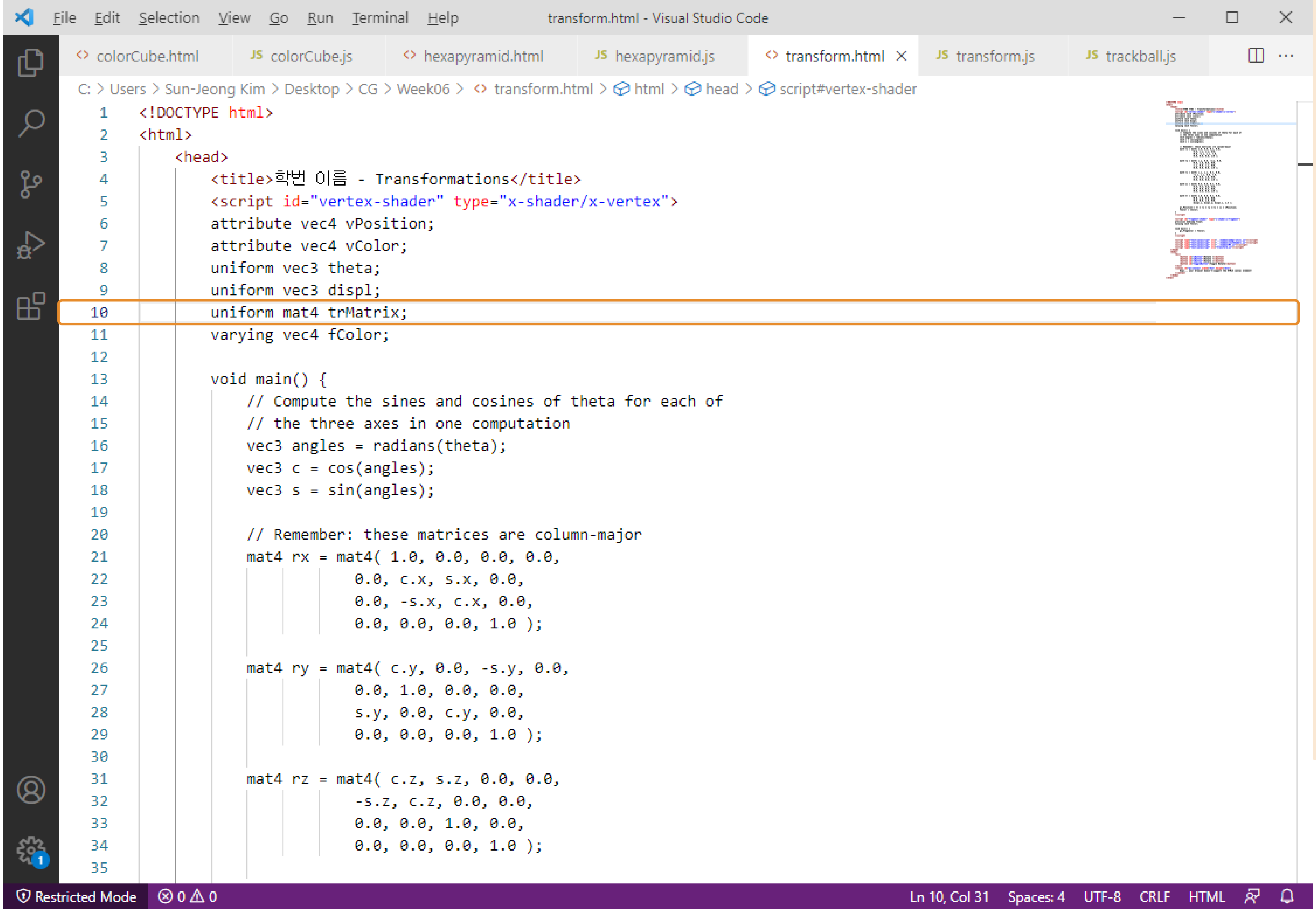
```
    function crossProduct(a, b, c) {
        c[0] = a[1]*b[2] - a[2]*b[1];
        c[1] = a[2]*b[0] - a[0]*b[2];
        c[2] = a[0]*b[1] - a[1]*b[0];
    }
    //------------------------------------------------------

    function project(xi, yi, vec) {
        // project x, y onto a hemisphere centered within width, height
        vec[0] = (2.0*xi - width) / width;
        vec[1] = (height - 2.0*yi) / height;
        var dist = Math.sqrt(vec[0]*vec[0] + vec[1]*vec[1]);
        vec[2] = Math.cos(Math.PI * 0.5 * ((dist<1.0)? dist : 1.0));

        // normalize
        normalize(vec);
    }

    function start(xi, yi) {
        project(xi, yi, lastPos);
    }

    function end(xi, yi) {
        var currPos = [0, 0, 0];
        project(xi, yi, currPos);

        var diff = [0, 0, 0];
        diff[0] = currPos[0] - lastPos[0];
        diff[1] = currPos[1] - lastPos[1];
        diff[2] = currPos[2] - lastPos[2];

        if( diff[0] || diff[1] || diff[2] ) {
            var angle = Math.PI * 0.5 * Math.sqrt(diff[0]*diff[0] + diff[1]*diff[1] + diff[2]*diff[2]);
            var axis = [0, 0, 0];
            crossProduct(currPos, lastPos, axis);
```



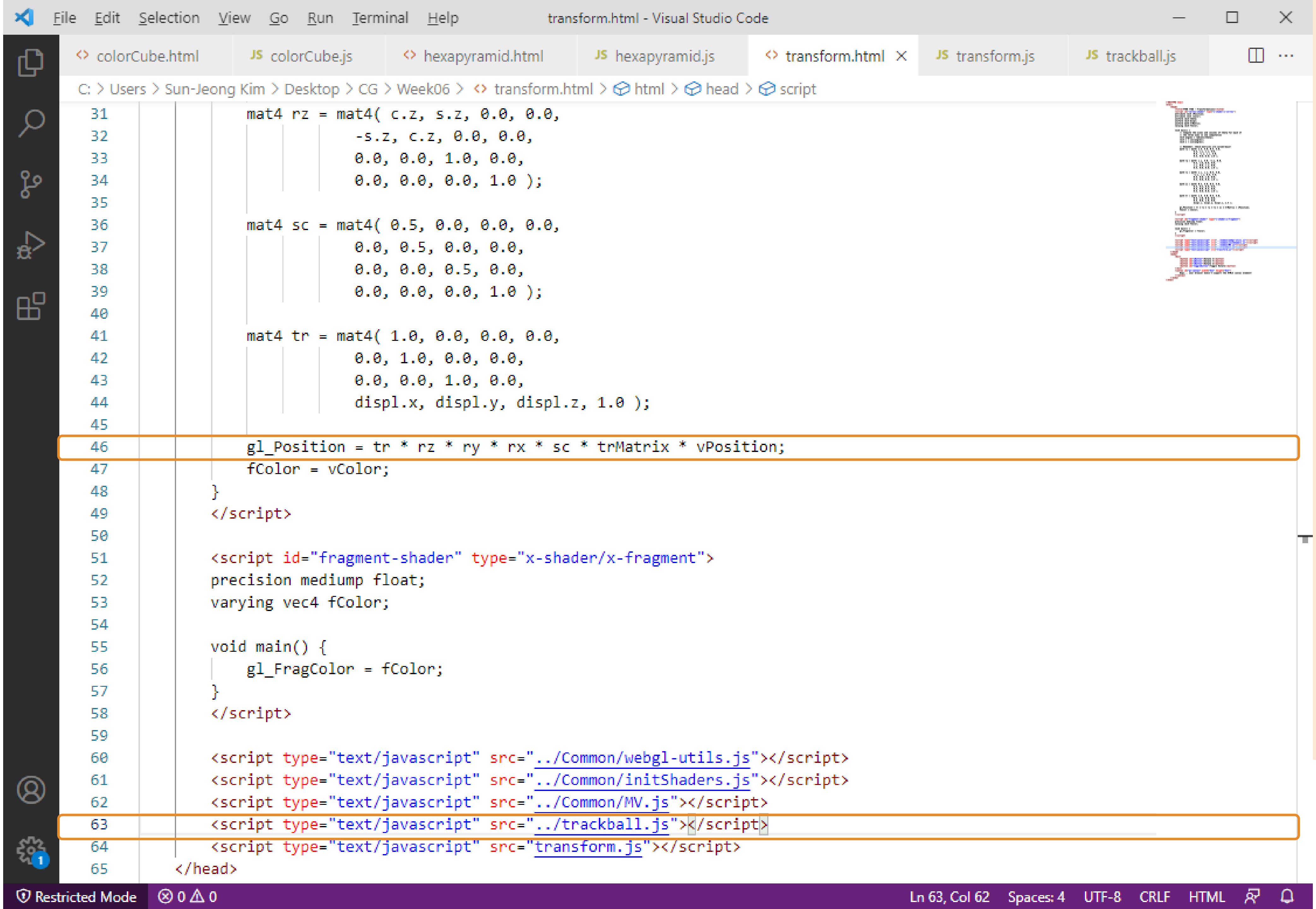
```javascript
        if( diff[0] || diff[1] || diff[2] ) {
            var angle = Math.PI * 0.5 * Math.sqrt(diff[0]*diff[0] + diff[1]*diff[1] + diff[2]*diff[2]);
            var axis = [0, 0, 0];
            crossProduct(currPos, lastPos, axis);
            normalize(axis);

            // create a quaternion
            var s2 = Math.sin(angle*0.5);
            var v2 = [s2*axis[0], s2*axis[1], s2*axis[2]];
            s2 = Math.cos(angle*0.5);

            // quaternions update -- multiplication of quaternions
            var s1 = s;
            var v1 = [v[0], v[1], v[2]];
            var v3 = [0, 0, 0];
            crossProduct(v1, v2, v3);
            s = s1*s2 - dotProduct(v1, v2);
            v[0] = s1*v2[0] + s2*v1[0] + v3[0];
            v[1] = s1*v2[1] + s2*v1[1] + v3[1];
            v[2] = s1*v2[2] + s2*v1[2] + v3[2];

            // normalize the quaternion
            var dist = 1.0 / Math.sqrt(s*s + v[0]*v[0] + v[1]*v[1] + v[2]*v[2]);
            s *= dist;
            v[0] *= dist;
            v[1] *= dist;
            v[2] *= dist;

            // rotation with quaternions
            // P' = quat * P * quat^-1
            // M = { { 1-2b^2-2c^2, 2ab-2sc,     2ac+2sb },
            //       { 2ab+2sc,     1-2a^2-2c^2, 2bc-2sa },
            //       { 2ac-2sb,     2bc+2sa,     1-2a^2-2b^2 } };

            // column major
```



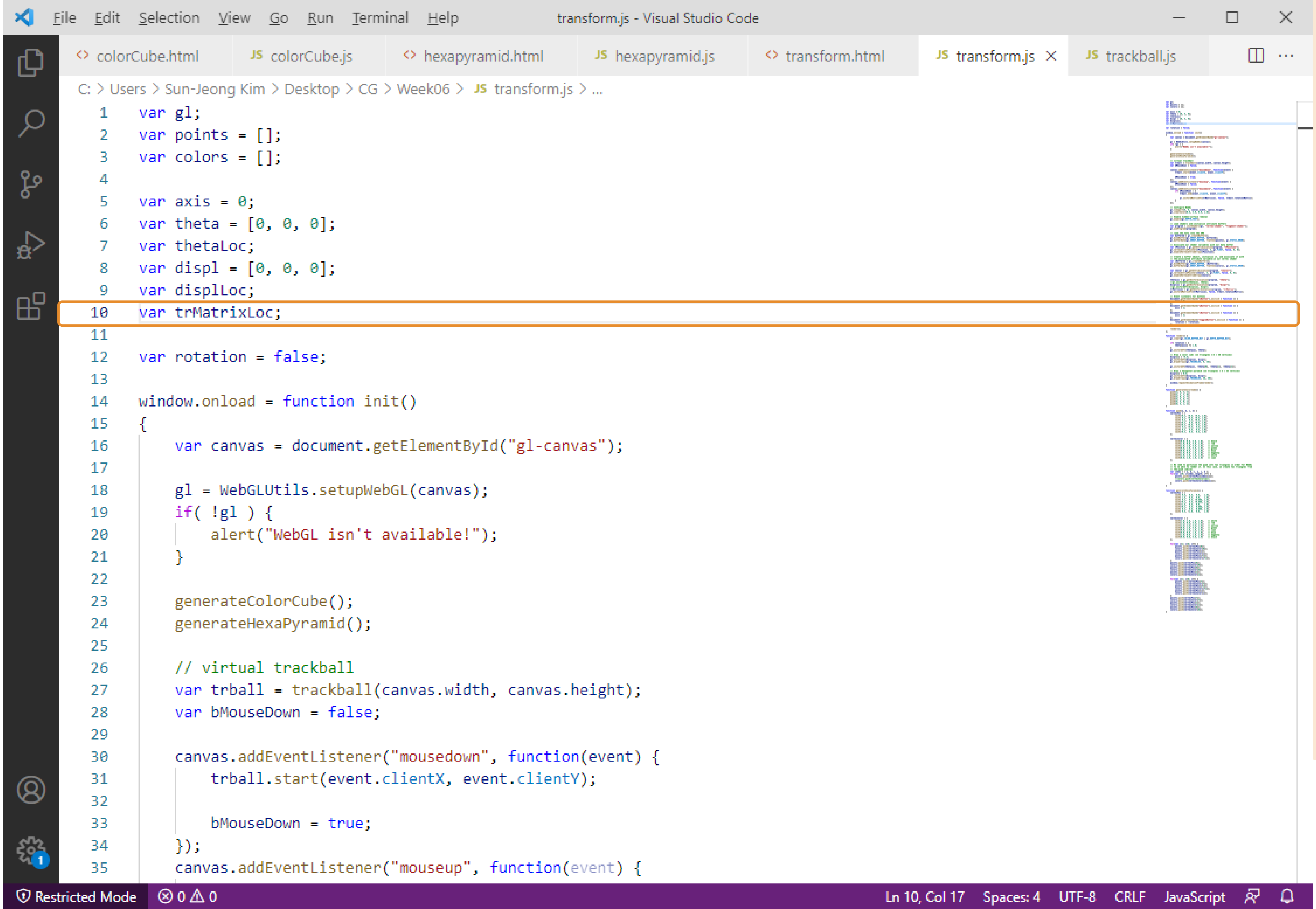
```

            // column major
            rotationMatrix[0] = 1.0 - 2.0 * (v[1]*v[1] + v[2]*v[2]);
            rotationMatrix[1] =       2.0 * (v[0]*v[1] + s*v[2]);
            rotationMatrix[2] =       2.0 * (v[2]*v[0] - s*v[1]);
            //rotationMatrix[3] = 0.0;

            rotationMatrix[4] =       2.0 * (v[0]*v[1] - s*v[2]);
            rotationMatrix[5] = 1.0 - 2.0 * (v[2]*v[2] + v[0]*v[0]);
            rotationMatrix[6] =       2.0 * (v[1]*v[2] + s*v[0]);
            //rotationMatrix[7] = 0.0;

            rotationMatrix[8] =        2.0 * (v[2]*v[0] + s*v[1]);
            rotationMatrix[9] =        2.0 * (v[1]*v[2] - s*v[0]);
            rotationMatrix[10] = 1.0 - 2.0 * (v[0]*v[0] + v[1]*v[1]);
            //rotationMatrix[11] = 0.0;

            //rotationMatrix[12] = rotationMatrix[13] = rotationMatrix[14] = 0.0;
            //rotationMatrix[15] = 1.0;

            lastPos[0] = currPos[0];
            lastPos[1] = currPos[1];
            lastPos[2] = currPos[2];
        }
    }

    data.start = start;
    data.end = end;
    data.rotationMatrix = rotationMatrix;

    return data;
}

```



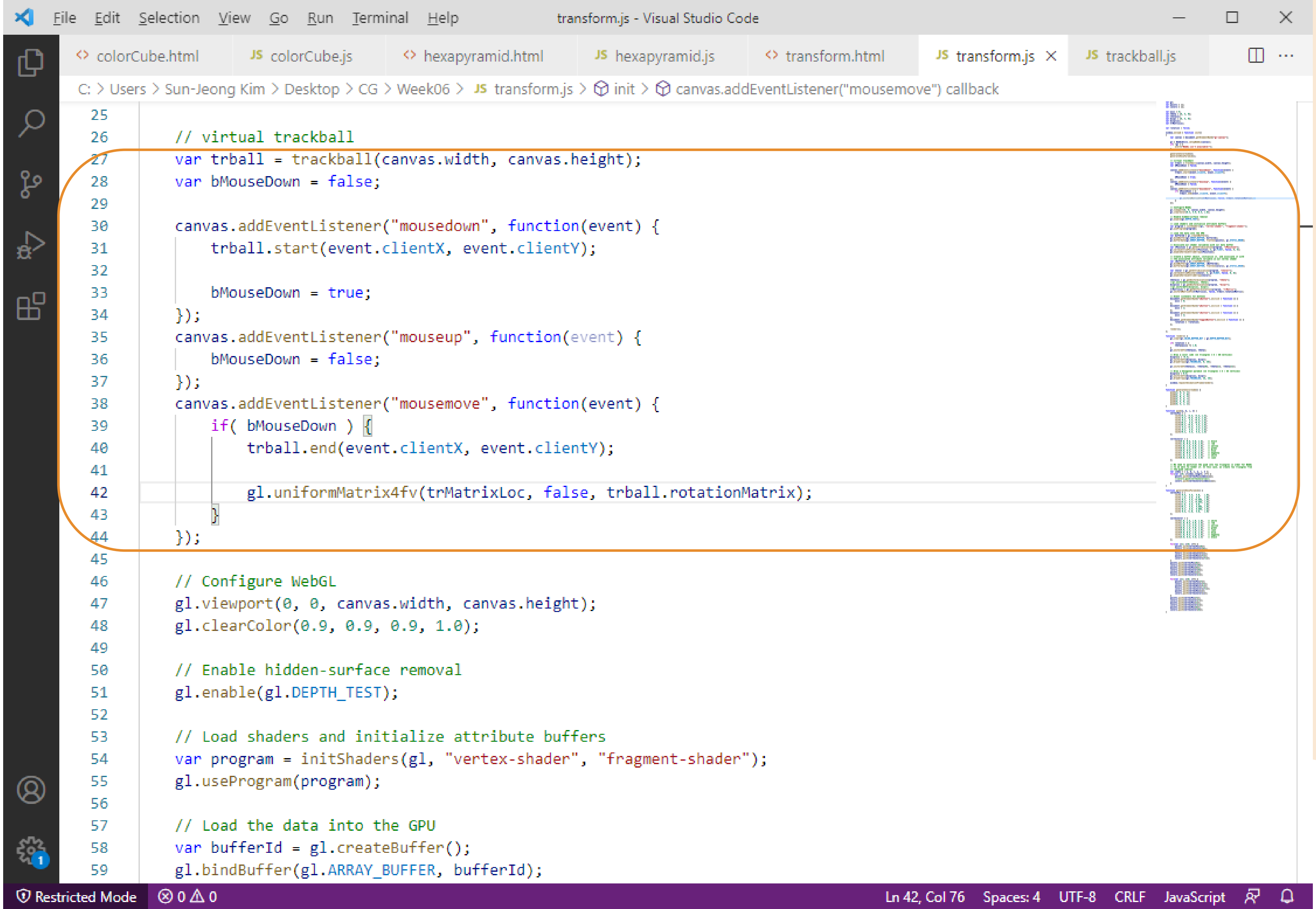
```
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Transformations</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;
        attribute vec4 vColor;
        uniform vec3 theta;
        uniform vec3 displ;
        uniform mat4 trMatrix;
        varying vec4 fColor;

        void main() {
            // Compute the sines and cosines of theta for each of
            // the three axes in one computation
            vec3 angles = radians(theta);
            vec3 c = cos(angles);
            vec3 s = sin(angles);

            // Remember: these matrices are column-major
            mat4 rx = mat4( 1.0, 0.0, 0.0, 0.0,
                            0.0, c.x, s.x, 0.0,
                            0.0, -s.x, c.x, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 ry = mat4( c.y, 0.0, -s.y, 0.0,
                            0.0, 1.0, 0.0, 0.0,
                            s.y, 0.0, c.y, 0.0,
                            0.0, 0.0, 0.0, 1.0 );

            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                            -s.z, c.z, 0.0, 0.0,
                            0.0, 0.0, 1.0, 0.0,
                            0.0, 0.0, 0.0, 1.0 );
```


File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
trackball.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script
Restricted Mode
0 0
Ln 63, Col 62
Spaces: 4
UTF-8
CRLF
HTML

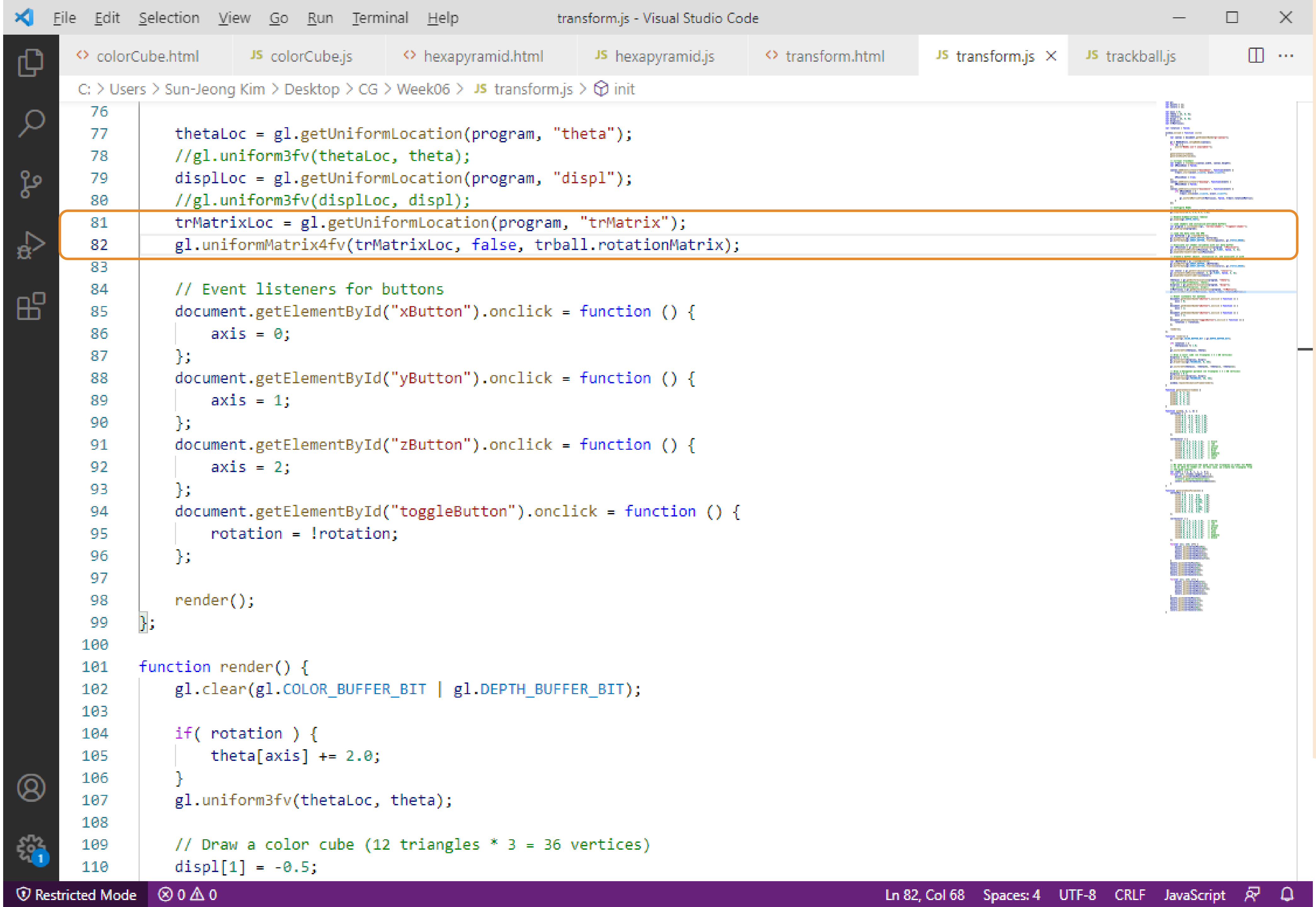
```
        mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                    -s.z, c.z, 0.0, 0.0,
                    0.0, 0.0, 1.0, 0.0,
                    0.0, 0.0, 0.0, 1.0 );

        mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
                    0.0, 0.5, 0.0, 0.0,
                    0.0, 0.0, 0.5, 0.0,
                    0.0, 0.0, 0.0, 1.0 );

        mat4 tr = mat4( 1.0, 0.0, 0.0, 0.0,
                    0.0, 1.0, 0.0, 0.0,
                    0.0, 0.0, 1.0, 0.0,
                    displ.x, displ.y, displ.z, 1.0 );

        gl_Position = tr * rz * ry * rx * sc * trMatrix * vPosition;
        fColor = vColor;
    }
    </script>

    <script id="fragment-shader" type="x-shader/x-fragment">
    precision mediump float;
    varying vec4 fColor;

    void main() {
        gl_FragColor = fColor;
    }
    </script>

    <script type="text/javascript" src="../Common/webgl-utils.js"></script>
    <script type="text/javascript" src="../Common/initShaders.js"></script>
    <script type="text/javascript" src="../Common/MV.js"></script>
    <script type="text/javascript" src="../trackball.js"></script>
    <script type="text/javascript" src="transform.js"></script>
</head>
```



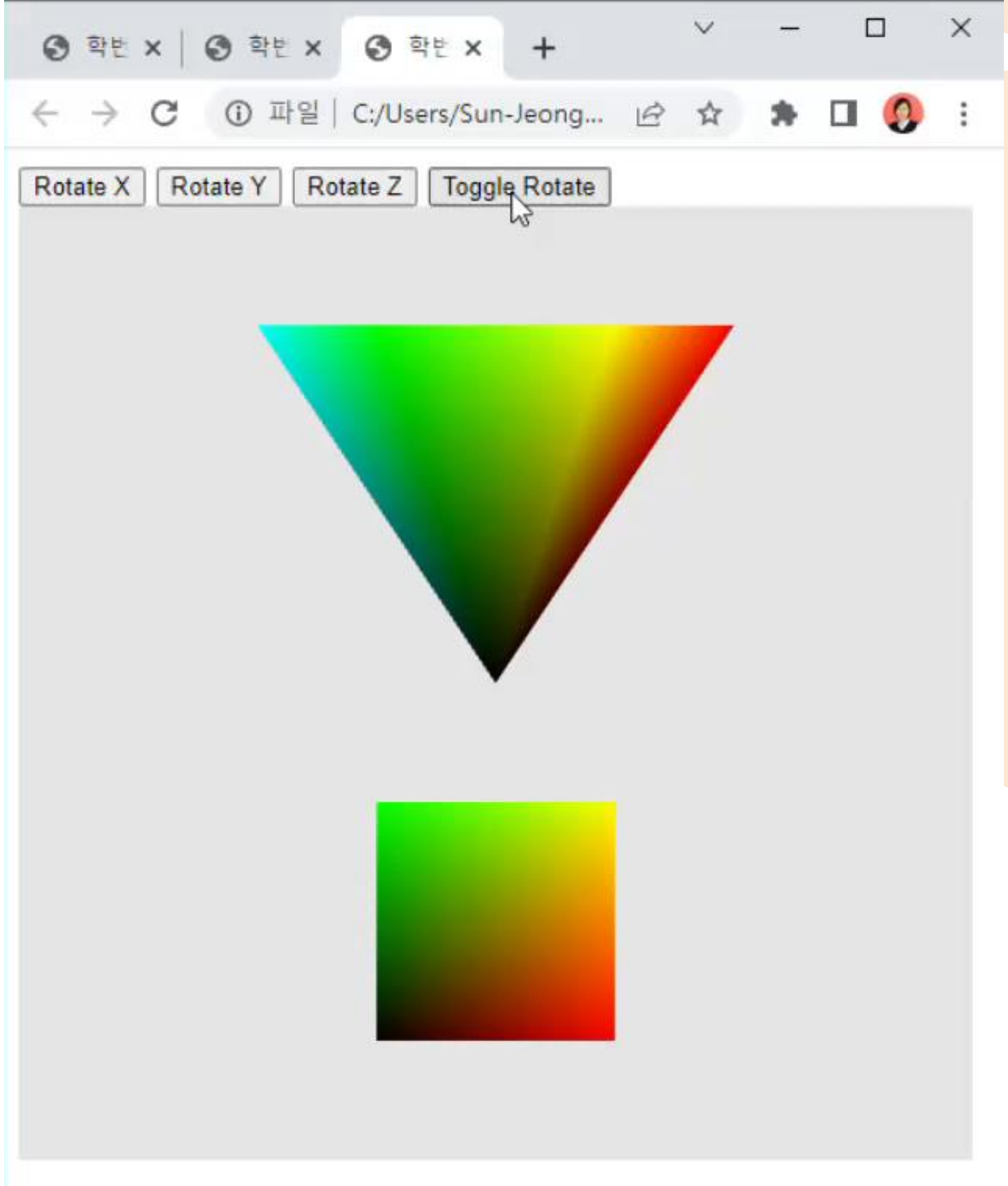
```javascript
var gl;
var points = [];
var colors = [];

var axis = 0;
var theta = [0, 0, 0];
var thetaLoc;
var displ = [0, 0, 0];
var displLoc;
var trMatrixLoc;

var rotation = false;

window.onload = function init()
{
    var canvas = document.getElementById("gl-canvas");

    gl = WebGLUtils.setupWebGL(canvas);
    if( !gl ) {
        alert("WebGL isn't available!");
    }

    generateColorCube();
    generateHexaPyramid();

    // virtual trackball
    var trball = trackball(canvas.width, canvas.height);
    var bMouseDown = false;

    canvas.addEventListener("mousedown", function(event) {
        trball.start(event.clientX, event.clientY);

        bMouseDown = true;
    });
    canvas.addEventListener("mouseup", function(event) {
```

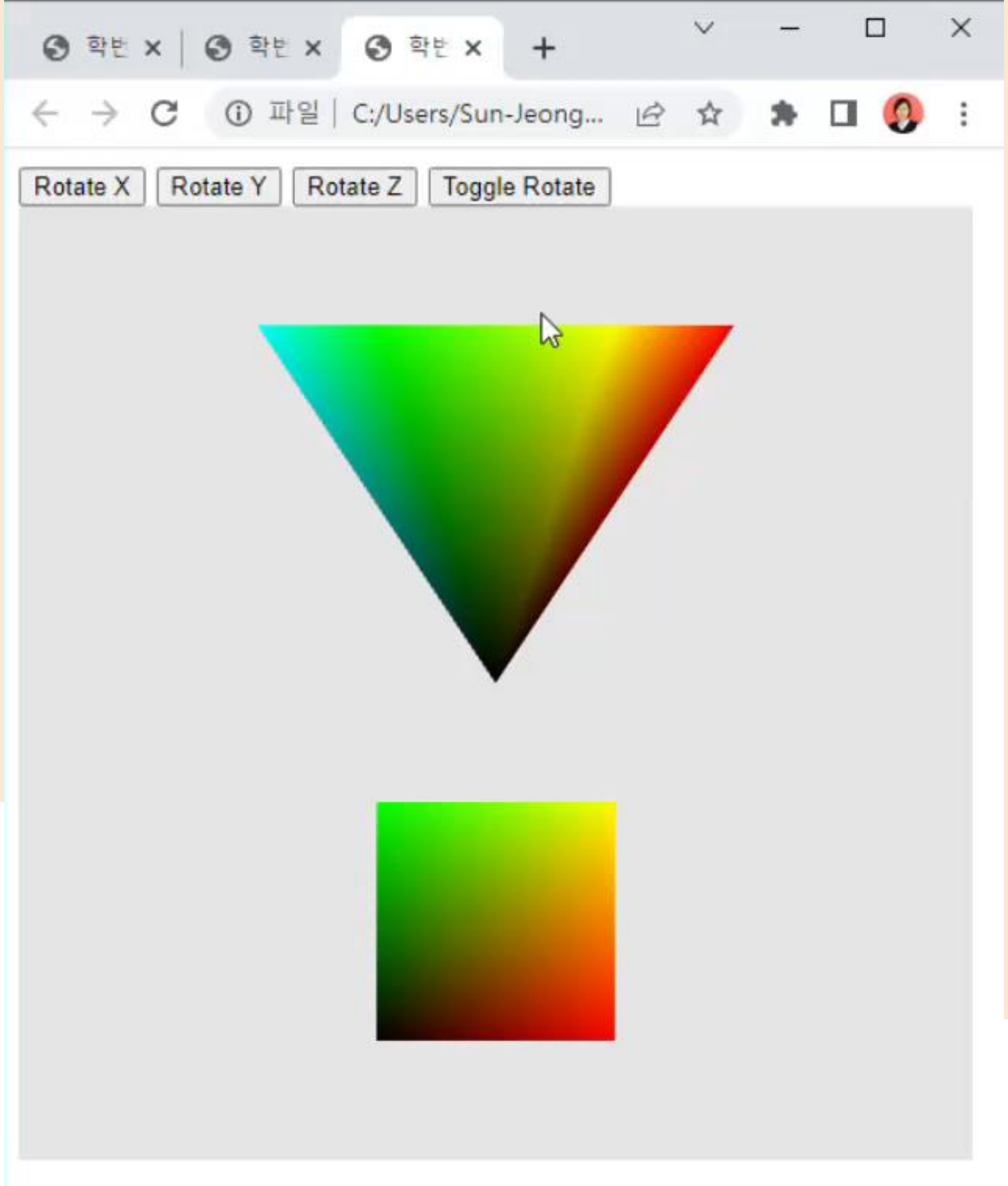

File Edit Selection View Go Run Terminal Help
transform.js - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
trackball.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.js > init > canvas.addEventListener("mousemove") callback
Restricted Mode
0 0
Ln 42, Col 76
Spaces: 4
UTF-8
CRLF
JavaScript
1

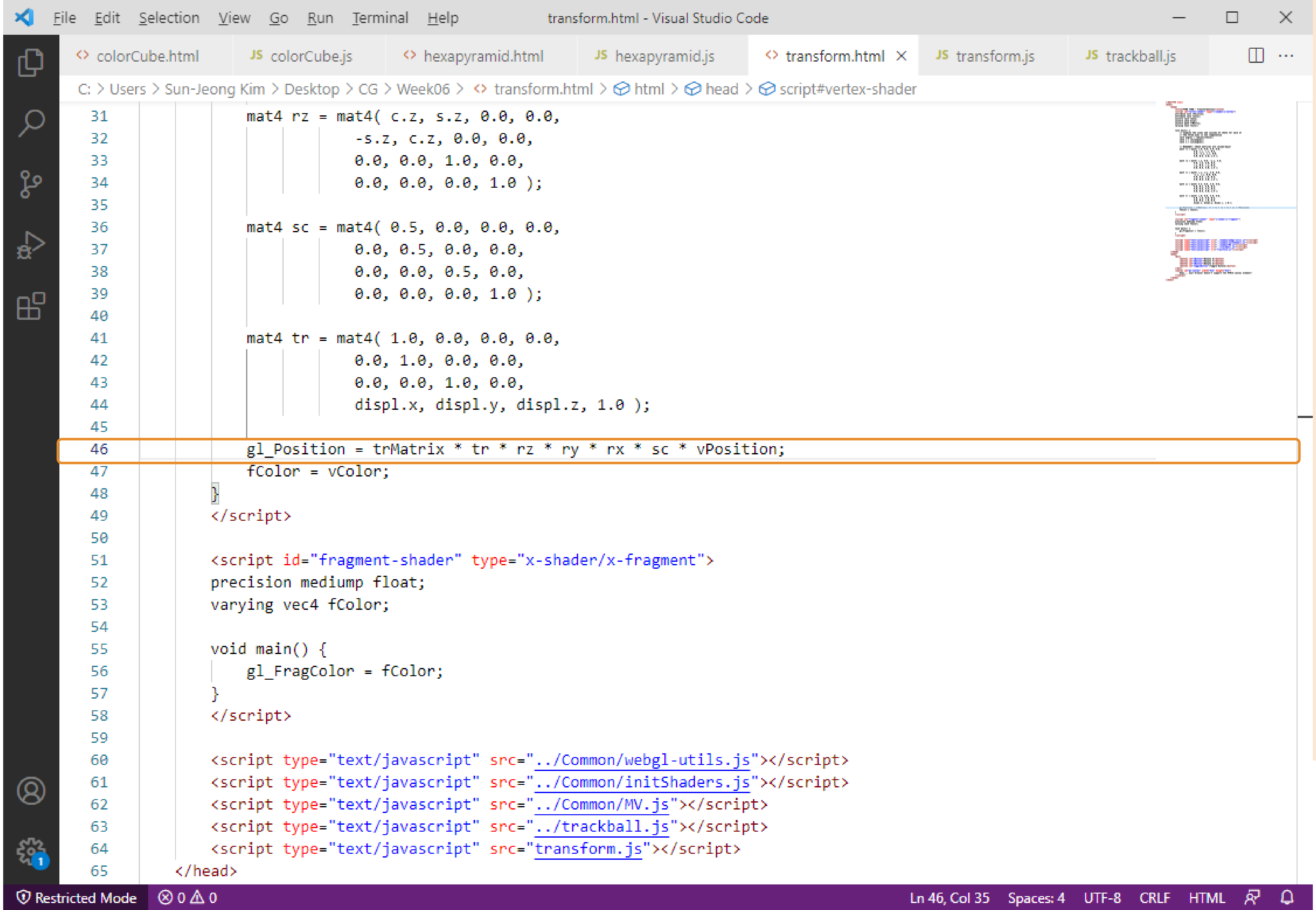
```javascript

    // virtual trackball
    var trball = trackball(canvas.width, canvas.height);
    var bMouseDown = false;

    canvas.addEventListener("mousedown", function(event) {
        trball.start(event.clientX, event.clientY);

        bMouseDown = true;
    });
    canvas.addEventListener("mouseup", function(event) {
        bMouseDown = false;
    });
    canvas.addEventListener("mousemove", function(event) {
        if( bMouseDown ) {
            trball.end(event.clientX, event.clientY);

            gl.uniformMatrix4fv(trMatrixLoc, false, trball.rotationMatrix);
        }
    });

    // Configure WebGL
    gl.viewport(0, 0, canvas.width, canvas.height);
    gl.clearColor(0.9, 0.9, 0.9, 1.0);

    // Enable hidden-surface removal
    gl.enable(gl.DEPTH_TEST);

    // Load shaders and initialize attribute buffers
    var program = initShaders(gl, "vertex-shader", "fragment-shader");
    gl.useProgram(program);

    // Load the data into the GPU
    var bufferId = gl.createBuffer();
    gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
```



```

    thetaLoc = gl.getUniformLocation(program, "theta");
    //gl.uniform3fv(thetaLoc, theta);
    displLoc = gl.getUniformLocation(program, "displ");
    //gl.uniform3fv(displLoc, displ);
    trMatrixLoc = gl.getUniformLocation(program, "trMatrix");
    gl.uniformMatrix4fv(trMatrixLoc, false, trball.rotationMatrix);

    // Event listeners for buttons
    document.getElementById("xButton").onclick = function () {
        axis = 0;
    };
    document.getElementById("yButton").onclick = function () {
        axis = 1;
    };
    document.getElementById("zButton").onclick = function () {
        axis = 2;
    };
    document.getElementById("toggleButton").onclick = function () {
        rotation = !rotation;
    };

    render();
};

function render() {
    gl.clear(gl.COLOR_BUFFER_BIT | gl.DEPTH_BUFFER_BIT);

    if( rotation ) {
        theta[axis] += 2.0;
    }
    gl.uniform3fv(thetaLoc, theta);

    // Draw a color cube (12 triangles * 3 = 36 vertices)
    displ[1] = -0.5;
```


학번
학번
학번
파일
C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate

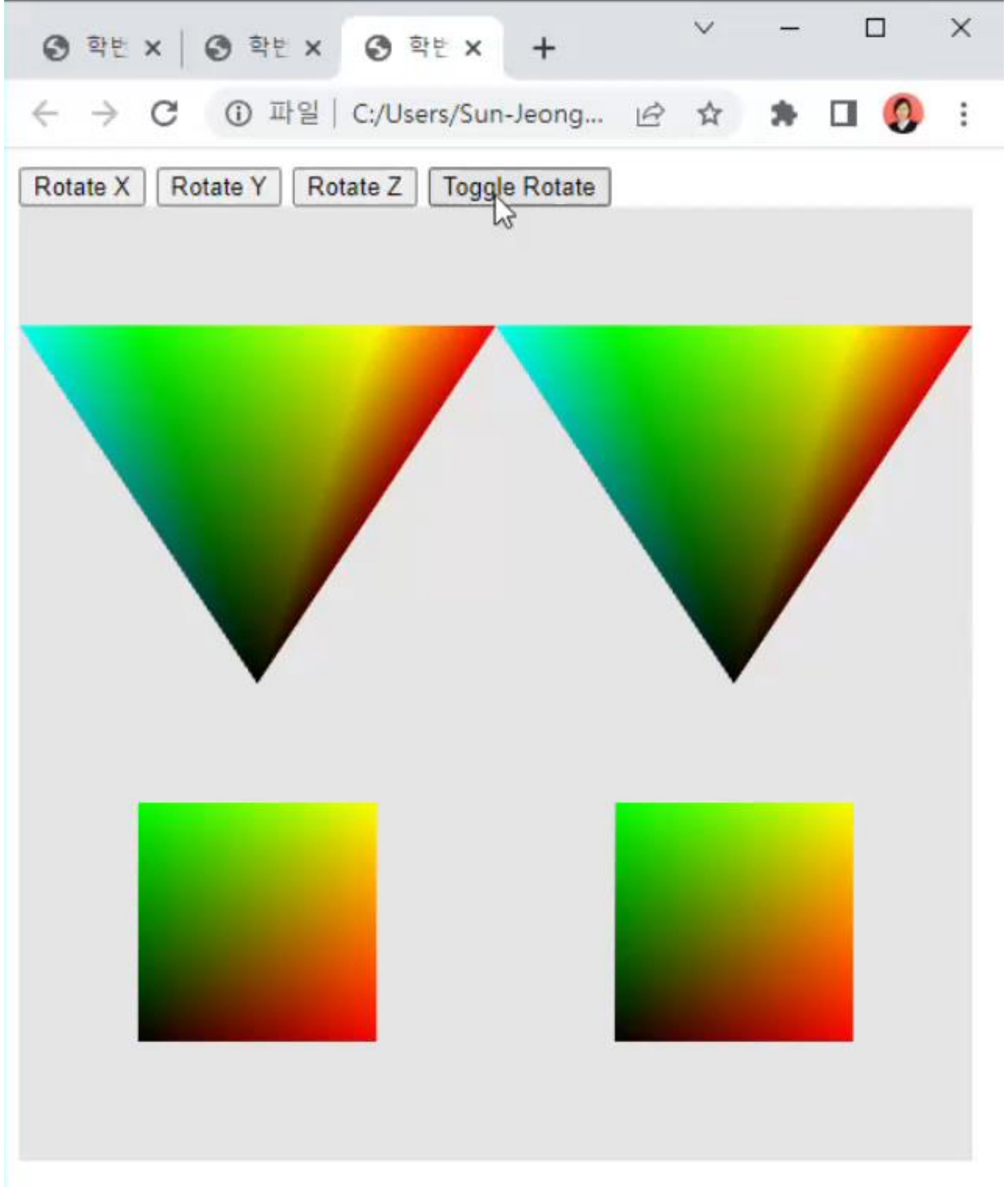

학번
학번
학번
파일
C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate


File Edit Selection View Go Run Terminal Help
transform.html - Visual Studio Code
colorCube.html
colorCube.js
hexapyramid.html
hexapyramid.js
transform.html
transform.js
trackball.js
C: > Users > Sun-Jeong Kim > Desktop > CG > Week06 > transform.html > html > head > script#vertex-shader
Restricted Mode
0 0
Ln 46, Col 35
Spaces: 4
UTF-8
CRLF
HTML

```
            mat4 rz = mat4( c.z, s.z, 0.0, 0.0,
                        -s.z, c.z, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 sc = mat4( 0.5, 0.0, 0.0, 0.0,
                        0.0, 0.5, 0.0, 0.0,
                        0.0, 0.0, 0.5, 0.0,
                        0.0, 0.0, 0.0, 1.0 );

            mat4 tr = mat4( 1.0, 0.0, 0.0, 0.0,
                        0.0, 1.0, 0.0, 0.0,
                        0.0, 0.0, 1.0, 0.0,
                        displ.x, displ.y, displ.z, 1.0 );

            gl_Position = trMatrix * tr * rz * ry * rx * sc * vPosition;
            fColor = vColor;
        }
        </script>

        <script id="fragment-shader" type="x-shader/x-fragment">
        precision mediump float;
        varying vec4 fColor;

        void main() {
            gl_FragColor = fColor;
        }
        </script>

        <script type="text/javascript" src="../Common/webgl-utils.js"></script>
        <script type="text/javascript" src="../Common/initShaders.js"></script>
        <script type="text/javascript" src="../Common/MV.js"></script>
        <script type="text/javascript" src="../trackball.js"></script>
        <script type="text/javascript" src="transform.js"></script>
    </head>
```

연습 문제 (6)

- 4개 오브젝트를 그리시오.
  - 상하 오브젝트들은 서로 반대 방향으로 회전
  - 좌우 오브젝트들도 서로 반대 방향으로 회전


학번
학번
학번
파일
C:/Users/Sun-Jeong...
Rotate X
Rotate Y
Rotate Z
Toggle Rotate