Graphics Programming

3RD WEEK, 2022
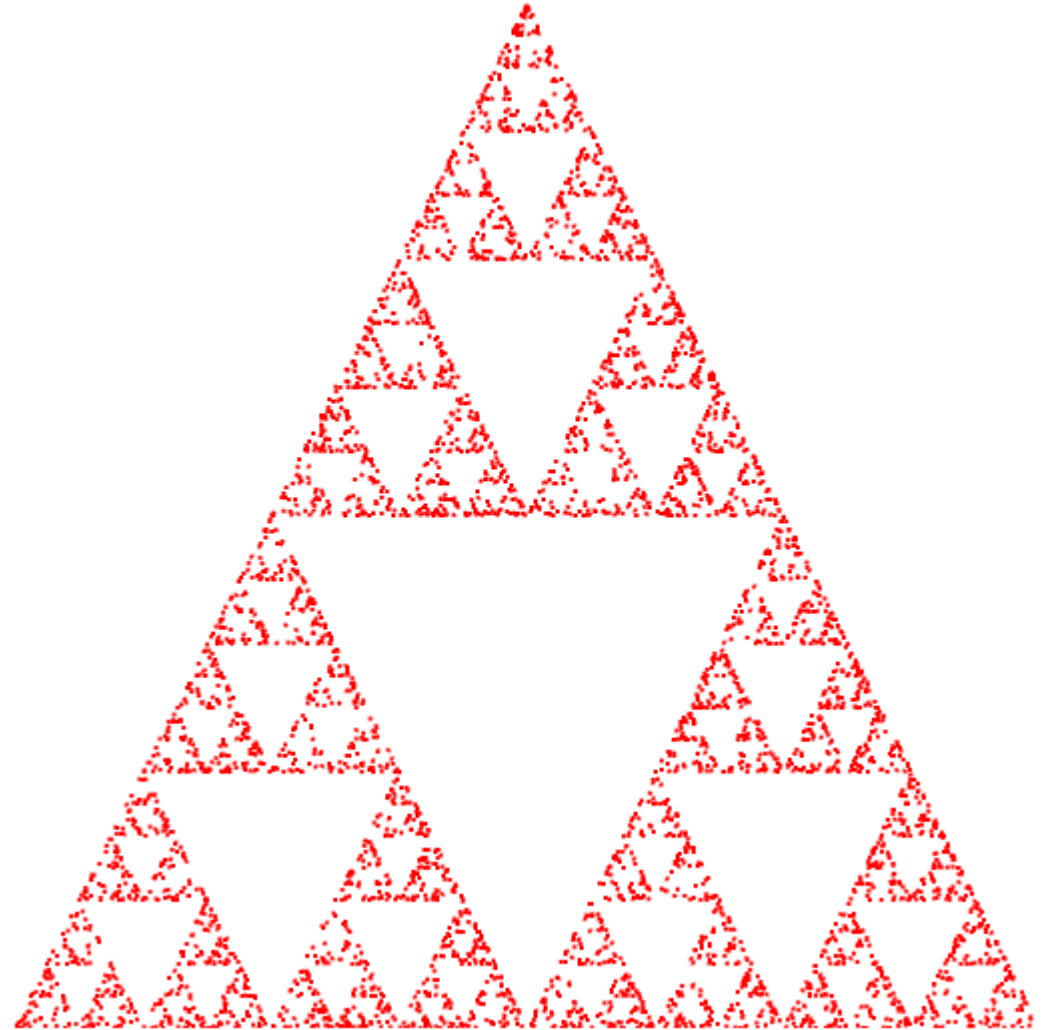
The Sierpinski Gasket (1)

- What is?
  - Interesting shape in area such as fractal geometry
  - Object that can be defined recursively and randomly
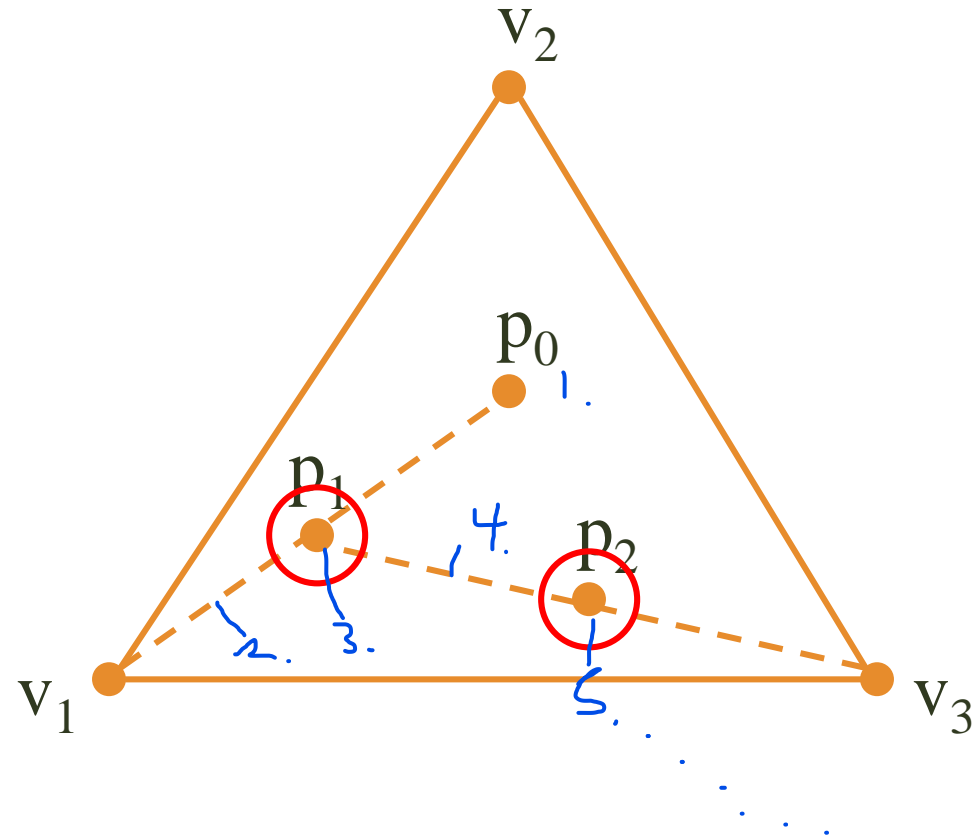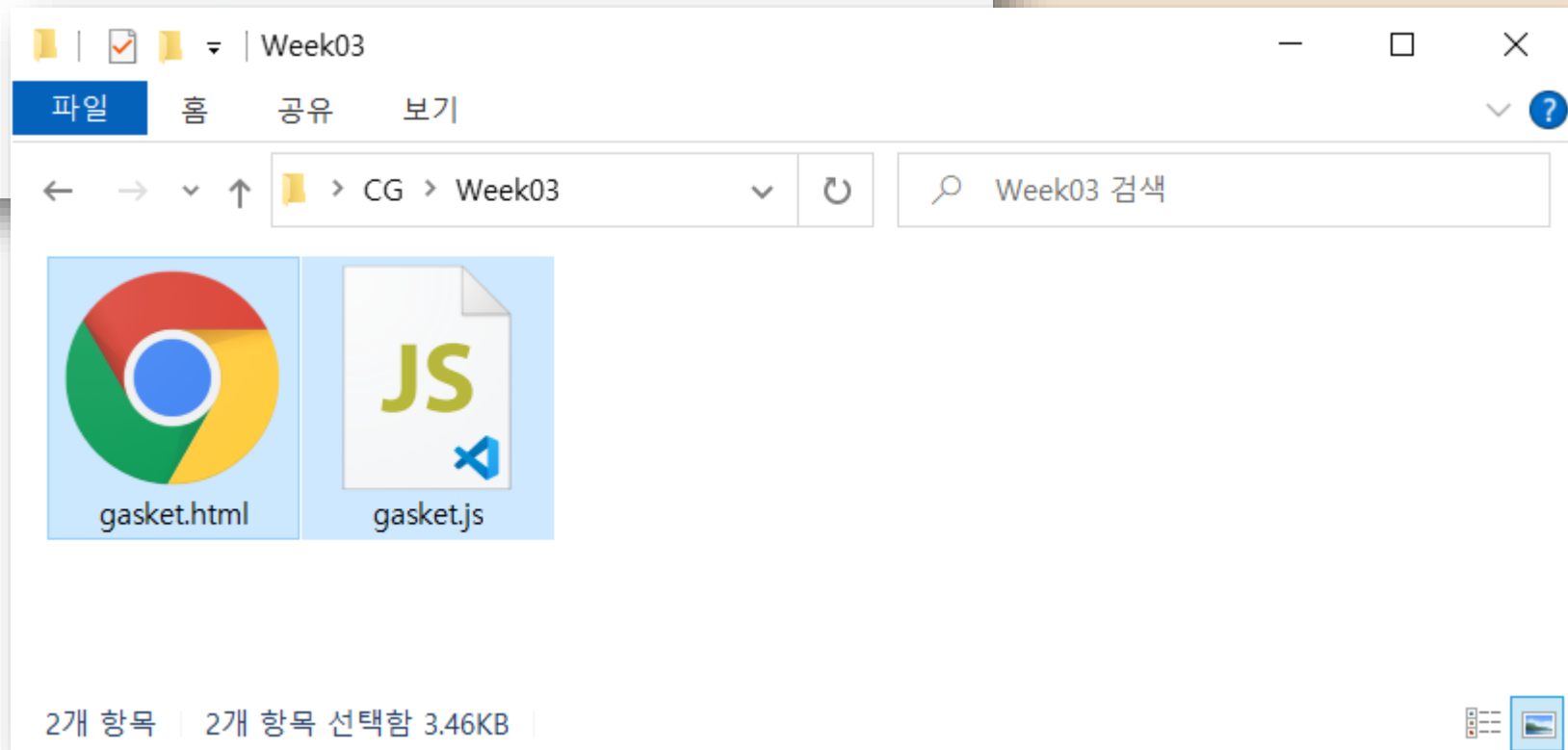
  반복적인

  프렉탈의 한 종류

The Sierpinski Gasket (2)

- How to…
  - Start with three vertices in the plane
  1. Pick an initial point at random inside the triangle
  2. Select one of the three vertices at random
  3. Find the point halfway between the initial point and the randomly selected vertex
  4. Display this new point by putting some sort of marker, such as a small circle, at its location
  5. Replace the initial point with this new point
  6. Return to step 2

The Sierpinski Gasket (2)

- How to...

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Sierpinski Gasket</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;

        void main() {
            gl_PointSize = 2.0;
            gl_Position = vPosition;
        }
        </script>

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

        void main() {
            gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);
        }
        </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="gasket.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
    </body>
</html>
```

6

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS gasket.js > ...

```javascript
var gl;
var points;
var numPoints = 5000;

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

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

    // Sierpinski Gasket
    generatePoints();

    // 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, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    render();
```
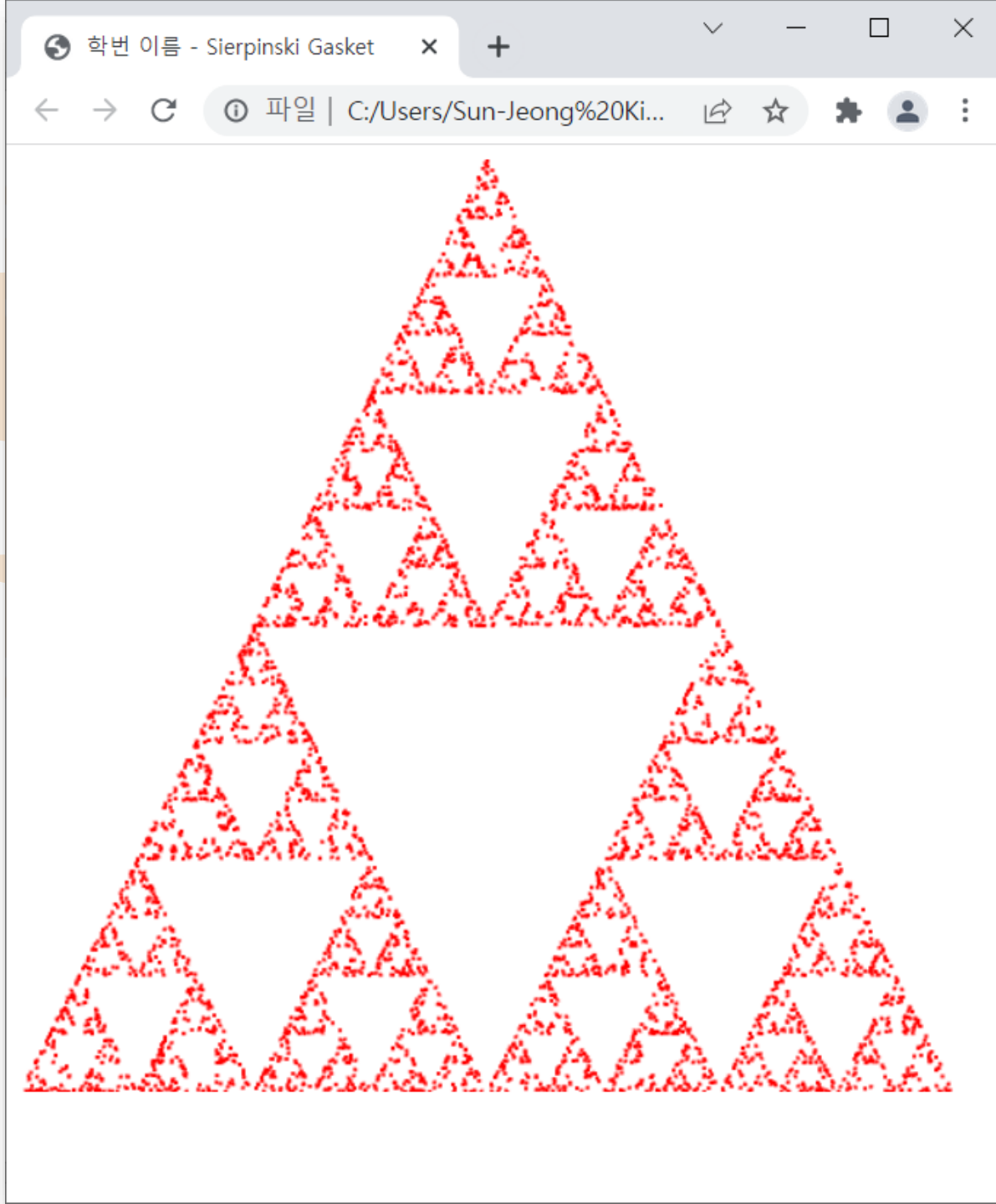
7

```js
36    };
37
38    function render() {
39        gl.clear(gl.COLOR_BUFFER_BIT);
40        gl.drawArrays(gl.POINTS, 0, points.length);
41    }
42
43    function generatePoints() {
44        // Initialize the data for the Sierpinski Gasket
45        // First, initialize the corners of a gasket with three points
46        var vertices = [
47            vec2(-1, -1),
48            vec2(0, 1),
49            vec2(1, -1)
50        ];
51
52        // Specify a starting point p for iterations
53        // p must lie inside any set of three vertices
54        var u = add(vertices[0], vertices[1]);
55        var v = add(vertices[0], vertices[2]);
56        var p = scale(0.25, add(u, v));
57
58        // Add an initial point into the array of points
59        points = [p];
60
61        // Compute the new points
62        // Each new point is located midway between last point and a randomly chosen vertex
63        for (var i=0; points.length<numPoints; i++) {
64            var j = Math.floor(Math.random() * 3);
65            p = add(points[i], vertices[j]);
66            p = scale(0.5, p);
67            points.push(p);
68        }
69    }
70
```

8

연습문제 (1)

- Sierpinski Gasket에서 점의 개수를 사용자로부터 입력 받아 그리시오.

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - Sierpinski Gasket</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;

        void main() {
            gl_PointSize = 2.0;
            gl_Position = vPosition;
        }
        </script>

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

        void main() {
            gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);
        }
        </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="gasket.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
        <p>점의 개수: <input type="text" id="numPoints" value="5000" onchange="drawGasket()"></p>
    </body>
</html>
```

11

```javascript
52          // Specify a starting point p for iterations
53          // p must lie inside any set of three vertices
54          var u = add(vertices[0], vertices[1]);
55          var v = add(vertices[0], vertices[2]);
56          var p = scale(0.25, add(u, v));
57
58          // Add an initial point into the array of points
59          points = [p];
60
61          // Compute the new points
62          // Each new point is located midway between last point and a randomly chosen vertex
63          for (var i=0; points.length<numPoints; i++) {
64              var j = Math.floor(Math.random() * 3);
65              p = add(points[i], vertices[j]);
66              p = scale(0.5, p);
67              points.push(p);
68          }
69      }
70
71  function drawGasket() {
72          numPoints = parseInt(document.getElementById("numPoints").value);
73
74          if(numPoints > 0 && numPoints <= 50000) {
75              generatePoints();
76
77              gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
78
79              render();
80          }
81          else {
82              alert("점의 개수는 0보다 크고 50,000보다 작거나 같아야 합니다.");
83          }
84      }
85
```
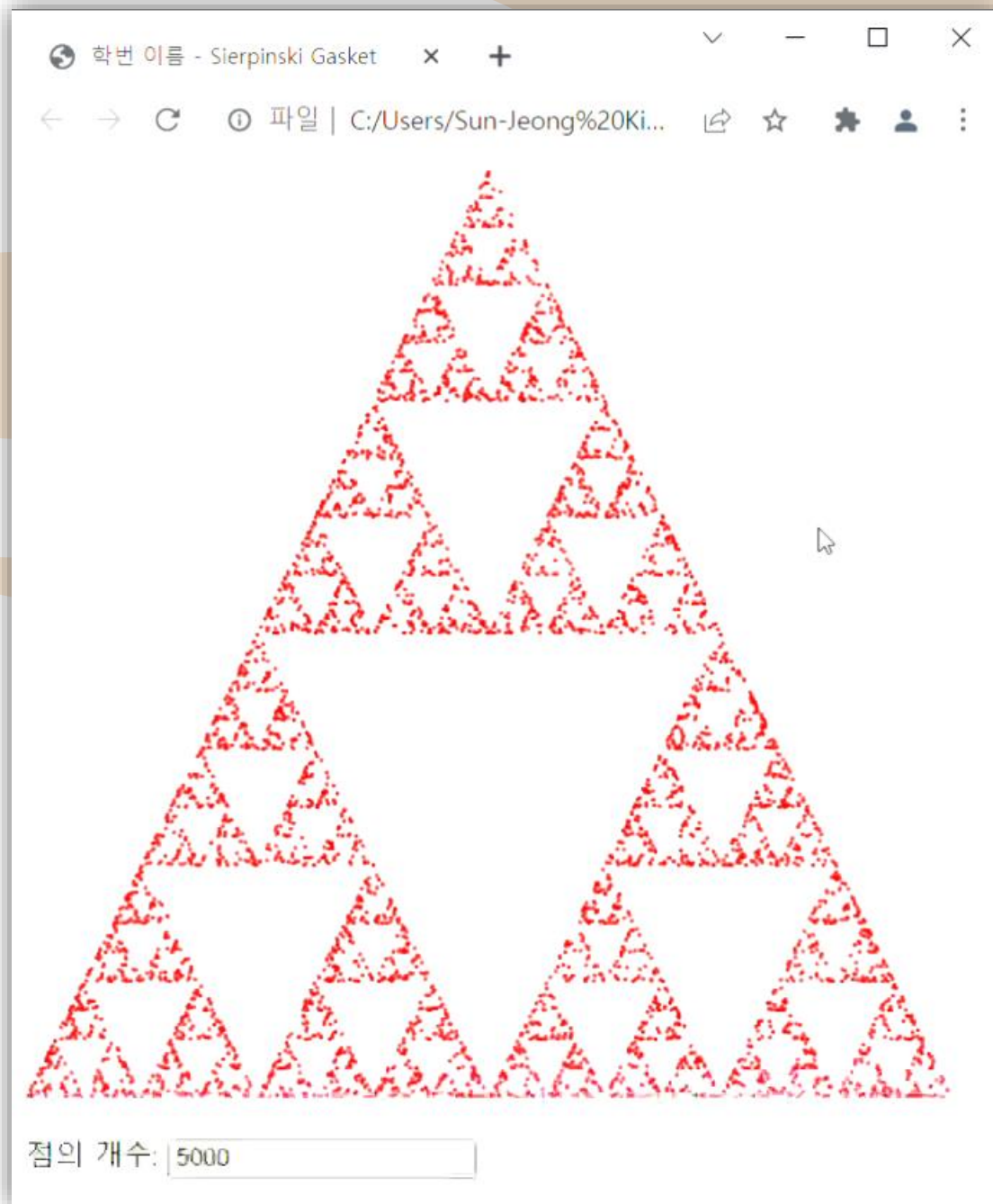
Polygons and Recursion

- Start with a triangle

- Connect bisectors of sides and remove central triangle

- Repeat

Example

- Five subdivisions

```html
<!DOCTYPE html>
<html>
    <head>
        <title>학번 이름 - 2D Sierpinski Gasket</title>
        <script id="vertex-shader" type="x-shader/x-vertex">
        attribute vec4 vPosition;

        void main() {
            //gl_PointSize = 2.0;
            gl_Position = vPosition;
        }
        </script>

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

        void main() {
            gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);
        }
        </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="gasket2d.js"></script>
    </head>
    <body>
        <canvas id="gl-canvas" width="512" height="512">
            Oops... your browser doesn't support the HTML5 canvas element!
        </canvas>
        <!--p>점의 개수: <input type="text" id="numPoints" value="5000" onchange="drawGasket()"></p-->
    </body>
</html>
```

17

```
1    var gl;
2    var points;
3    var numTimes = 5;
4
5    window.onload = function init()
6    {
7        var canvas = document.getElementById("gl-canvas");
8
9        gl = WebGLUtils.setupWebGL(canvas);
10       if( !gl ) {
11           alert("WebGL isn't available!");
12       }
13
14       // 2D Sierpinski Gasket
15       generateTriangles();
16
17       // Configure WebGL
18       gl.viewport(0, 0, canvas.width, canvas.height);
19       gl.clearColor(1.0, 1.0, 1.0, 1.0);
20
21       // Load shaders and initialize attribute buffers
22       var program = initShaders(gl, "vertex-shader", "fragment-shader");
23       gl.useProgram(program);
24
25       // Load the data into the GPU
26       var bufferId = gl.createBuffer();
27       gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
28       gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
29
30       // Associate our shader variables with our data buffer
31       var vPosition = gl.getAttribLocation(program, "vPosition");
32       gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
33       gl.enableVertexAttribArray(vPosition);
34
35       render();
```

18

File   Edit   Selection   View   Go   Run   Terminal   Help

<> gasket.html   JS gasket.js   <> gasket2d.html   JS gasket2d.js ✕

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS gasket2d.js > ⬡ divideTriangle

```javascript
22        var program = initShaders(gl, "vertex-shader", "fragment-shader");
23        gl.useProgram(program);
24
25        // Load the data into the GPU
26        var bufferId = gl.createBuffer();
27        gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
28        gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
29
30        // Associate our shader variables with our data buffer
31        var vPosition = gl.getAttribLocation(program, "vPosition");
32        gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
33        gl.enableVertexAttribArray(vPosition);
34
35        render();
36    };
37
38    function render() {
39        gl.clear(gl.COLOR_BUFFER_BIT);
40        gl.drawArrays(gl.TRIANGLES, 0, points.length);
41    }
42
43    function generateTriangles() {
44        // Initialize the data for the Sierpinski Gasket
45        // First, initialize the corners of a gasket with three points
46        var vertices = [
47            vec2(-1, -1),
48            vec2(0, 1),
49            vec2(1, -1)
50        ];
51
52        points = [];
53
54        divideTriangle(vertices[0], vertices[1], vertices[2], numTimes);
55    }
56
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 73, Col 42   Spaces: 4   UTF-8   CRLF   {} JavaScript

19

File Edit Selection View Go Run Terminal Help

<> gasket.html   JS gasket.js   <> gasket2d.html   JS gasket2d.js ✕

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS gasket2d.js > ⬡ divideTriangle

```javascript
43    function generateTriangles() {
44        // Initialize the data for the Sierpinski Gasket
45        // First, initialize the corners of a gasket with three points
46        var vertices = [
47            vec2(-1, -1),
48            vec2(0, 1),
49            vec2(1, -1)
50        ];
51
52        points = [];
53
54        divideTriangle(vertices[0], vertices[1], vertices[2], numTimes);
55    }
56
57    function divideTriangle(a, b, c, count) {
58        // check for the end of recursion
59        if (count == 0) {
60            points.push(a, b, c);
61        }
62        else {
63            // bisect the sides
64            var ab = mix(a, b, 0.5);
65            var bc = mix(b, c, 0.5);
66            var ca = mix(c, a, 0.5);
67
68            count--;
69
70            // three new triangles
71            divideTriangle(a, ab, ca, count);
72            divideTriangle(b, bc, ab, count);
73            divideTriangle(c, ca, bc, count);
74        }
75    }
76
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 73, Col 42   Spaces: 4   UTF-8   CRLF   {} JavaScript
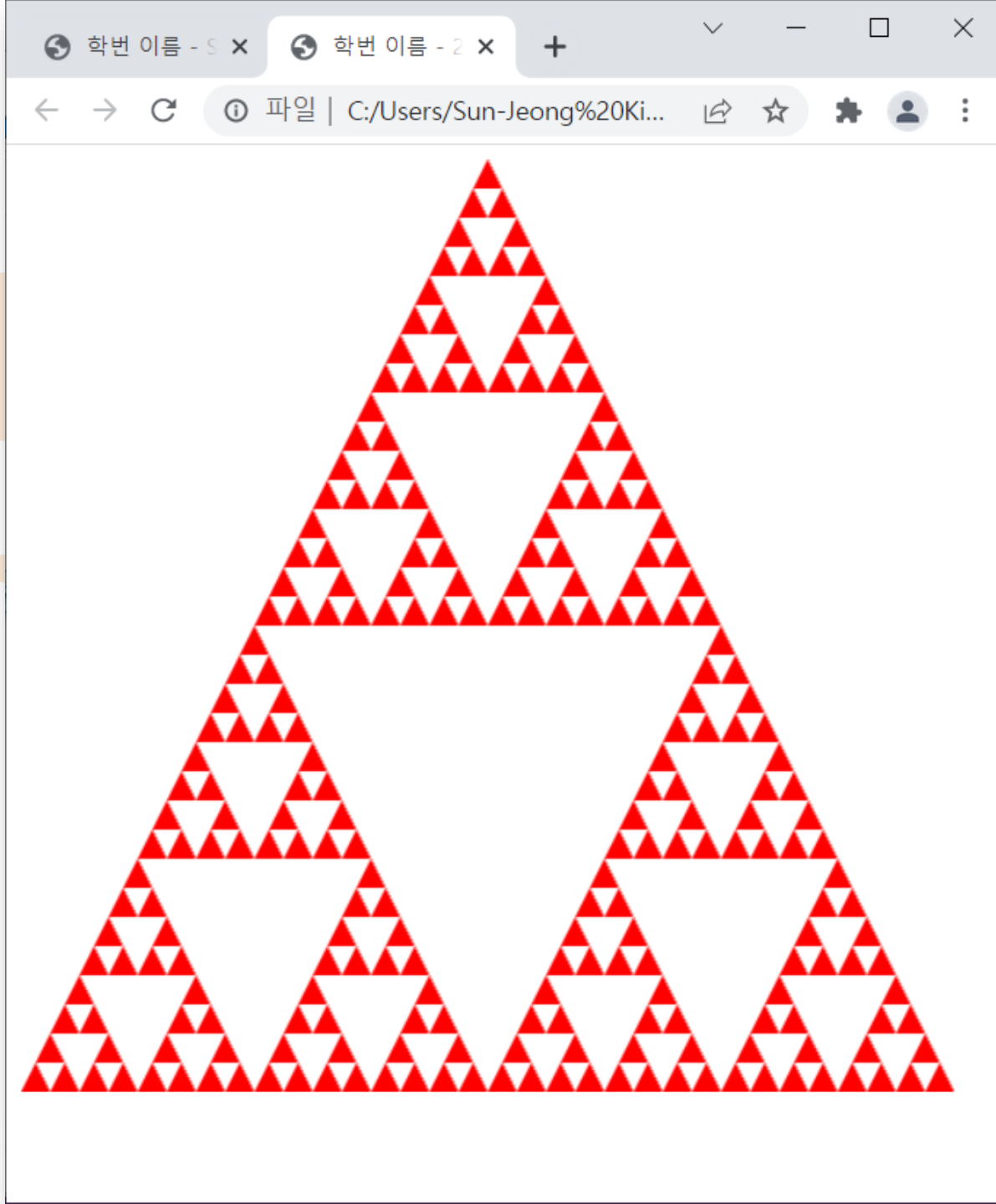
연습문제 (2)

- Sierpinski Gasket 2D에서 Subdivision 레벨을 사용자로부터 입력 받아 그리시오.
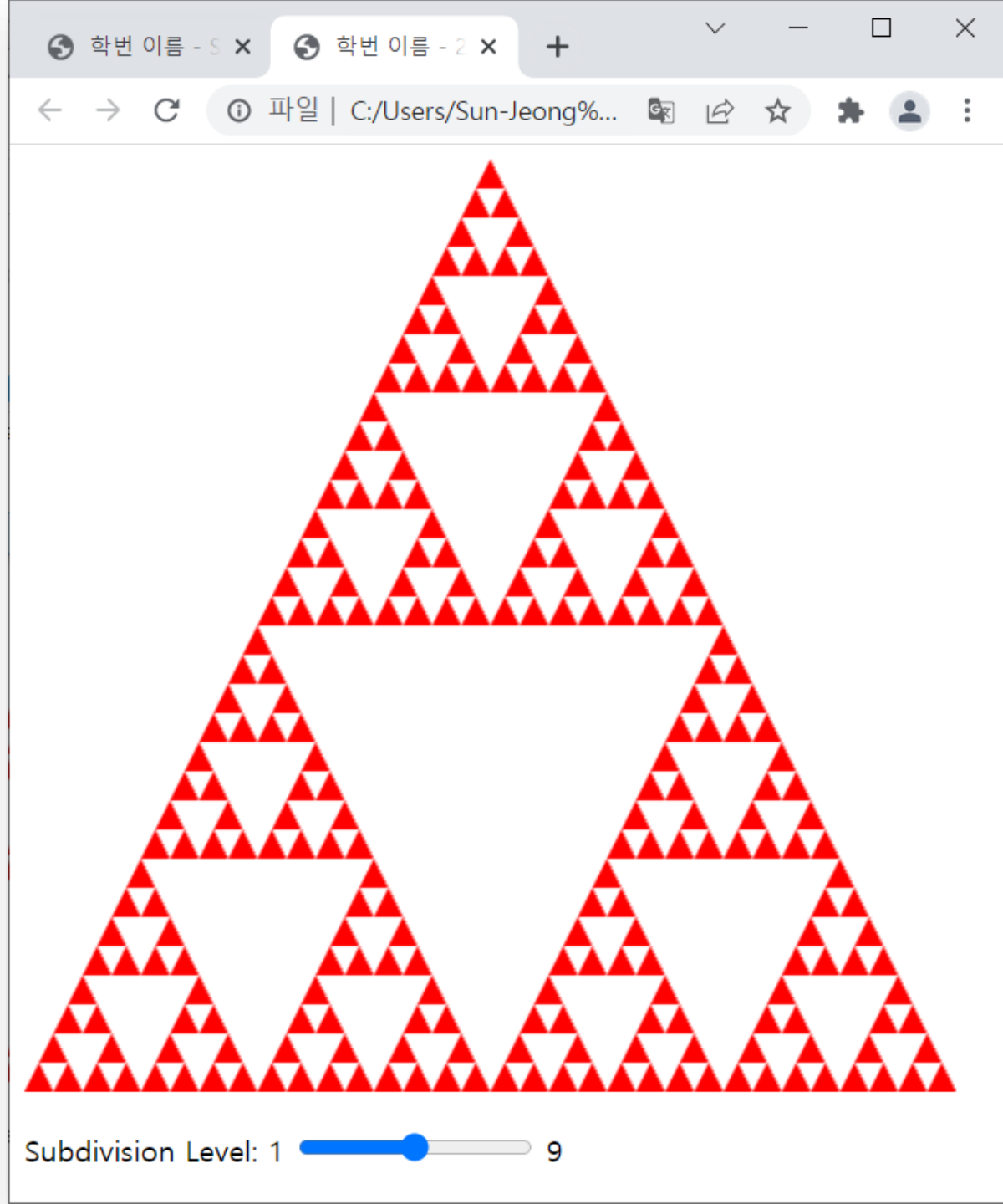
`<> gasket.html`   `JS gasket.js`   `<> gasket2d.html ✕`   `JS gasket2d.js`

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > <> gasket2d.html > ⊘ html > ⊘ body > ⊘ p

```html
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <title>학번 이름 - 2D Sierpinski Gasket</title>
5           <script id="vertex-shader" type="x-shader/x-vertex">
6           attribute vec4 vPosition;
7
8           void main() {
9               //gl_PointSize = 2.0;
10              gl_Position = vPosition;
11          }
12          </script>
13
14          <script id="fragment-shader" type="x-shader/x-fragment">
15          precision mediump float;
16
17          void main() {
18              gl_FragColor = vec4(1.0, 0.0, 0.0, 1.0);
19          }
20          </script>
21
22          <script type="text/javascript" src="../Common/webgl-utils.js"></script>
23          <script type="text/javascript" src="../Common/initShaders.js"></script>
24          <script type="text/javascript" src="../Common/MV.js"></script>
25          <script type="text/javascript" src="gasket2d.js"></script>
26      </head>
27      <body>
28          <canvas id="gl-canvas" width="512" height="512">
29              Oops... your browser doesn't support the HTML5 canvas element!
30          </canvas>
31          <p>Subdivision Level: 1 <input type="range" id="level" value="5" min="1" max="9" step="1" value="5"> 9</p>
32      </body>
33  </html>
```

23

```
gasket.html        gasket.js        gasket2d.html        gasket2d.js ✕
```

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > gasket2d.js > init > onchange

```javascript
 1    var gl;
 2    var points;
 3    var numTimes = 5;
 4
 5    window.onload = function init()
 6    {
 7        var canvas = document.getElementById("gl-canvas");
 8
 9        gl = WebGLUtils.setupWebGL(canvas);
10        if( !gl ) {
11            alert("WebGL isn't available!");
12        }
13
14        document.getElementById("level").onchange = function(event) {
15            numTimes = event.target.value;
16
17            generateTriangles();
18
19            gl.bufferData(gl.ARRAY_BUFFER, flatten(points), gl.STATIC_DRAW);
20
21            render();
22        }
23
24        // 2D Sierpinski Gasket
25        generateTriangles();
26
27        // Configure WebGL
28        gl.viewport(0, 0, canvas.width, canvas.height);
29        gl.clearColor(1.0, 1.0, 1.0, 1.0);
30
31        // Load shaders and initialize attribute buffers
32        var program = initShaders(gl, "vertex-shader", "fragment-shader");
33        gl.useProgram(program);
34
35        // Load the data into the GPU
```

24

Subdivision Level: 1 ━━━●━━━━━ 9
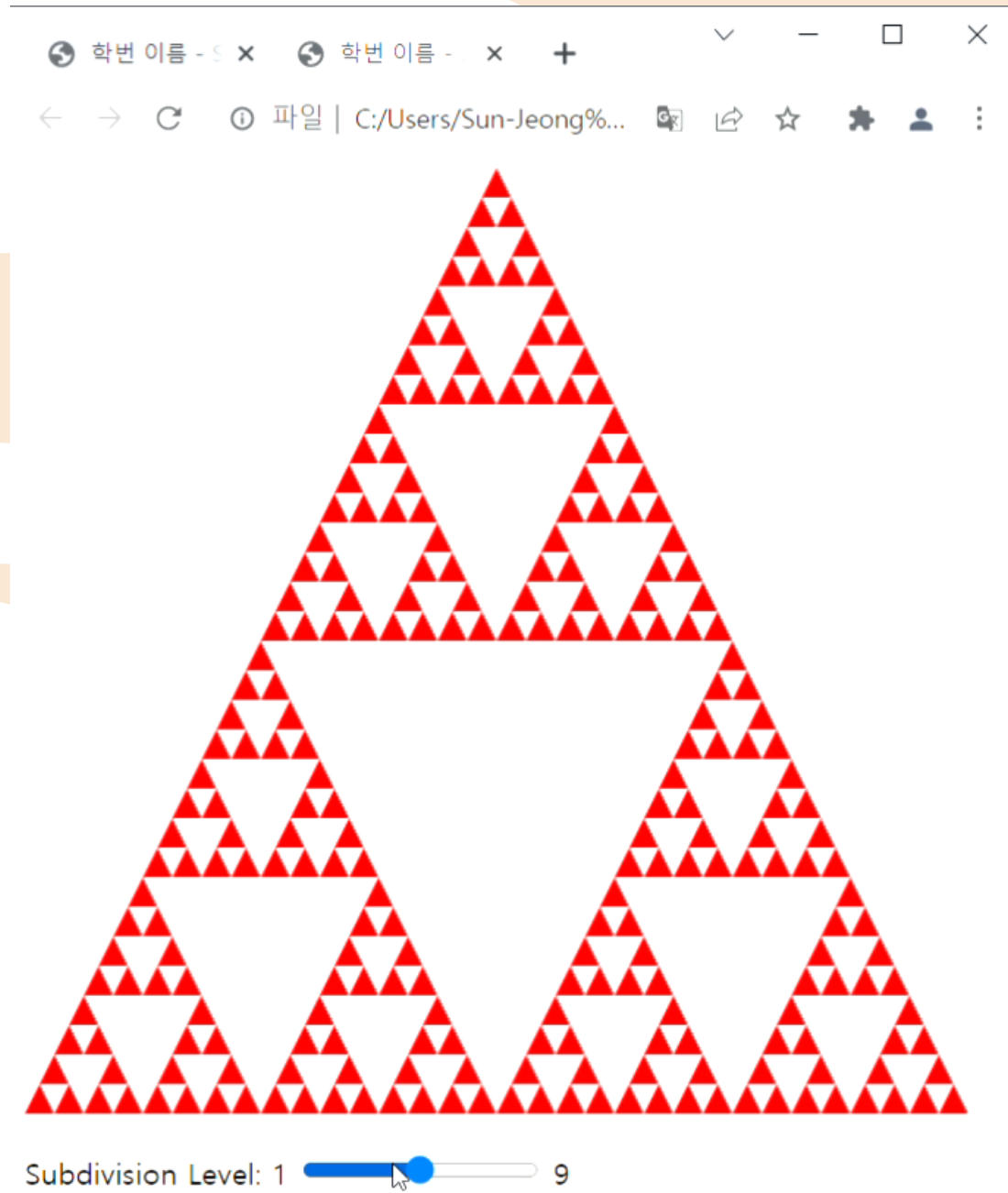
Attributes

- Properties that determines how to render a geometric primitive (appearance of objects)
  - <u>Color</u> (points, lines, polygons)
  - <u>Size</u> and width (points, lines)
  - Stipple pattern (lines, polygons)
  - Polygon mode
    - Display as filled: solid color or stipple pattern
    - Display edges
    - Display vertices
- Only a few (`gl_PointSize`) are supported by OpenGL functions

Color

- Three color theory
  - Our brains do not receive the entire color distribution but rather than three values
- <u>Additive</u> color – ex) CRT
- <u>Subtractive</u> color – ex) printing
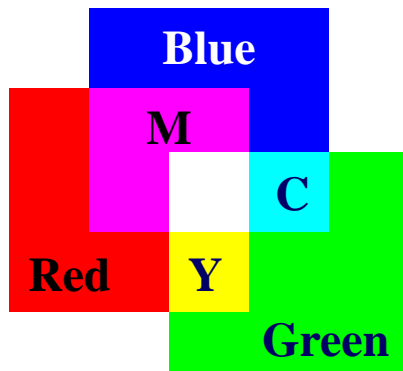
Additive Color
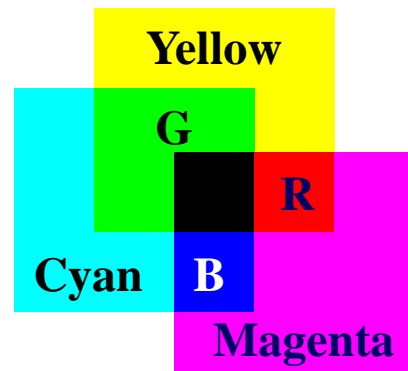
Subtractive Color
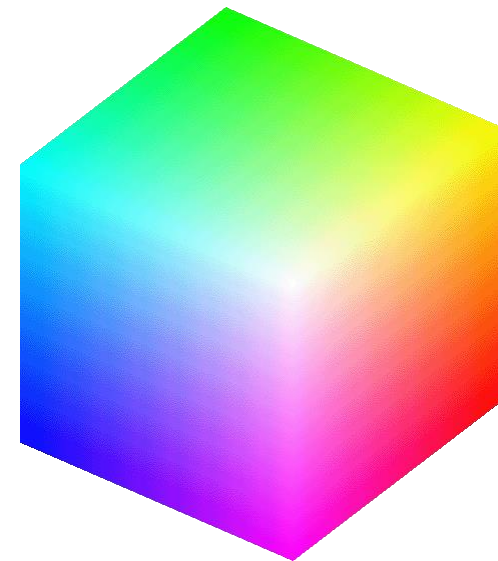
Color Solid

RGB Color

- Each color component is stored separately in the frame buffer
  - Usually 8 bits per component in buffer
  - Color values can range from 0.0 (none) to 1.0 (all) using floats or over the range from 0 to 255 using unsigned bytes

Frame buffer
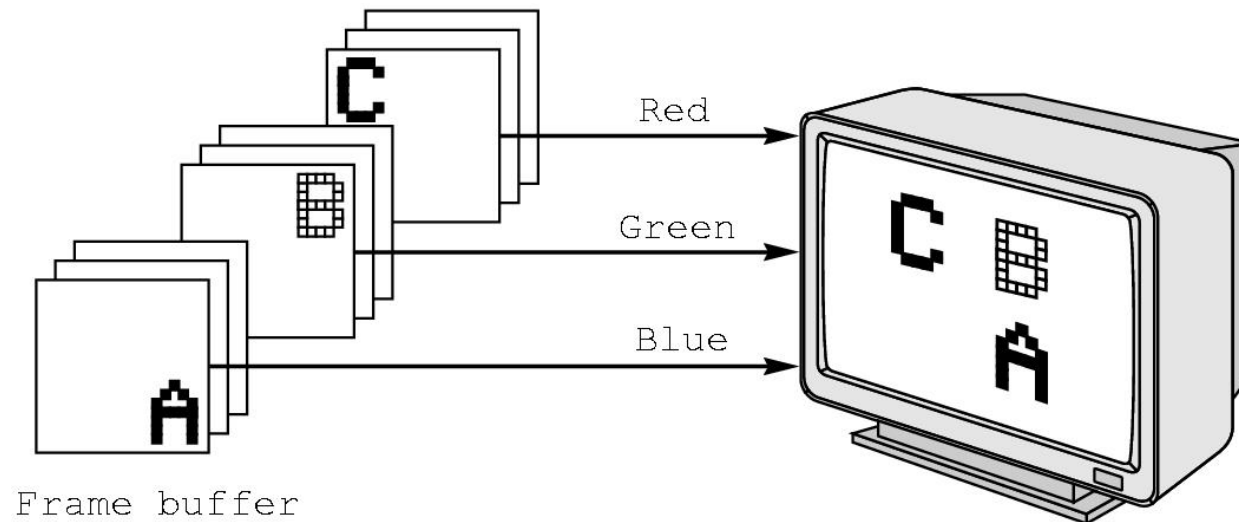
Smooth Color

- Default is <u>smooth</u> shading
  - OpenGL <u>interpolates</u> vertex colors across visible polygons
- Alternative is flat shading
  - Color of first vertex determines fill color
  - Handle in shader

Setting Colors

에서편집
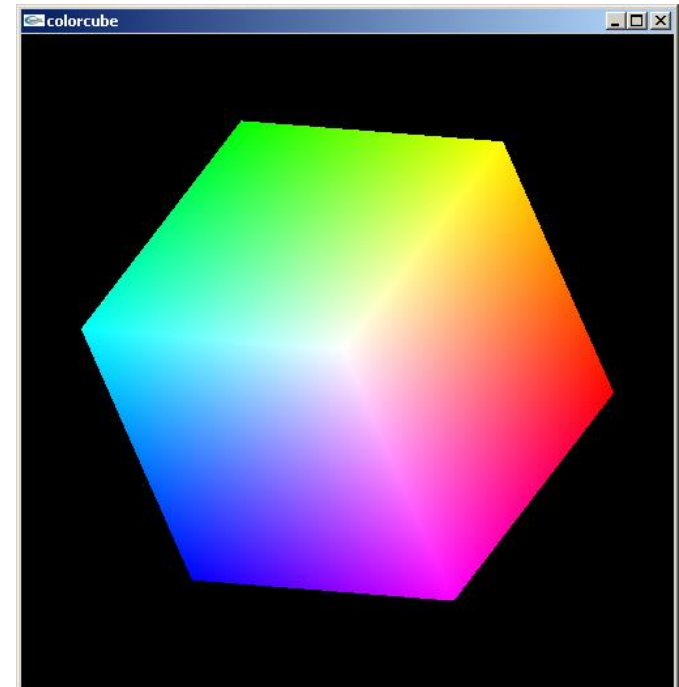
- Colors are ultimately set in the <u>fragment shader</u> but can be determined in either shader or in the application

- Application color: pass to vertex shader as a uniform variable or as a vertex attribute

- Vertex shader color: pass to fragment shader as varying variable

- Fragment color: can alter via shader code
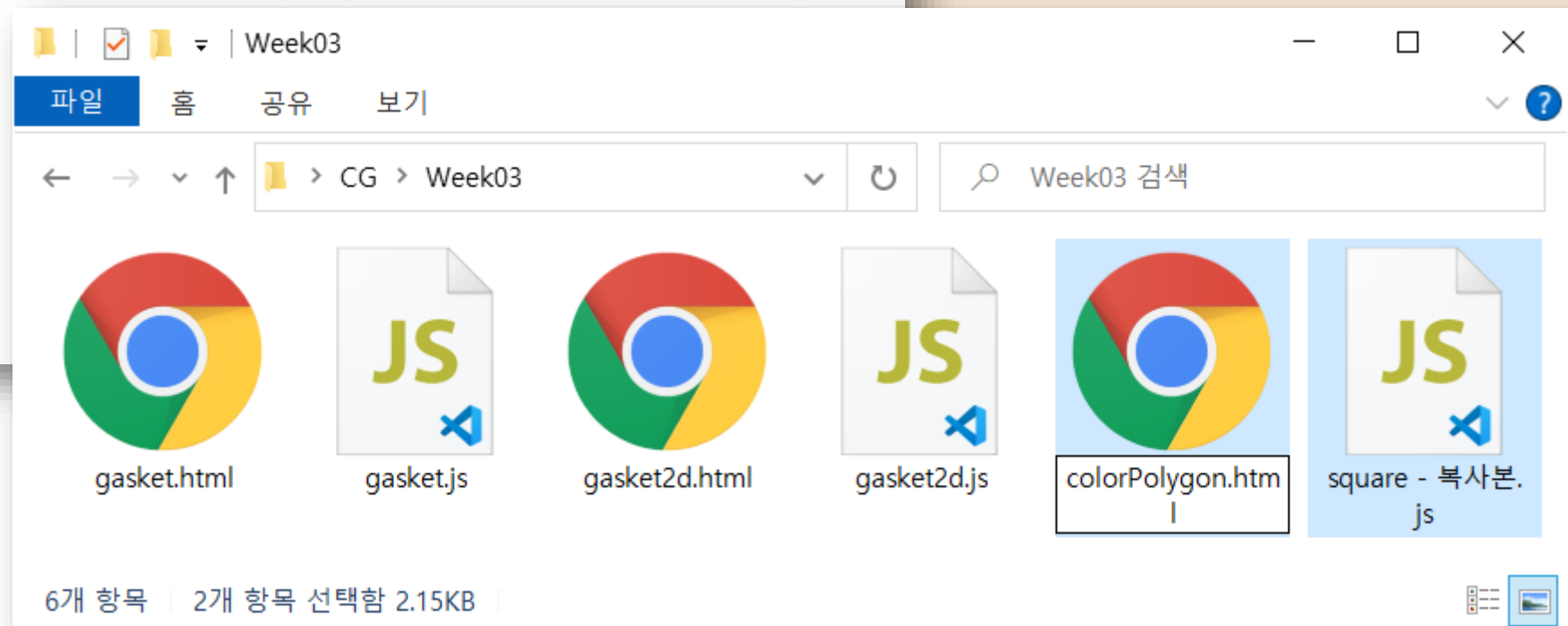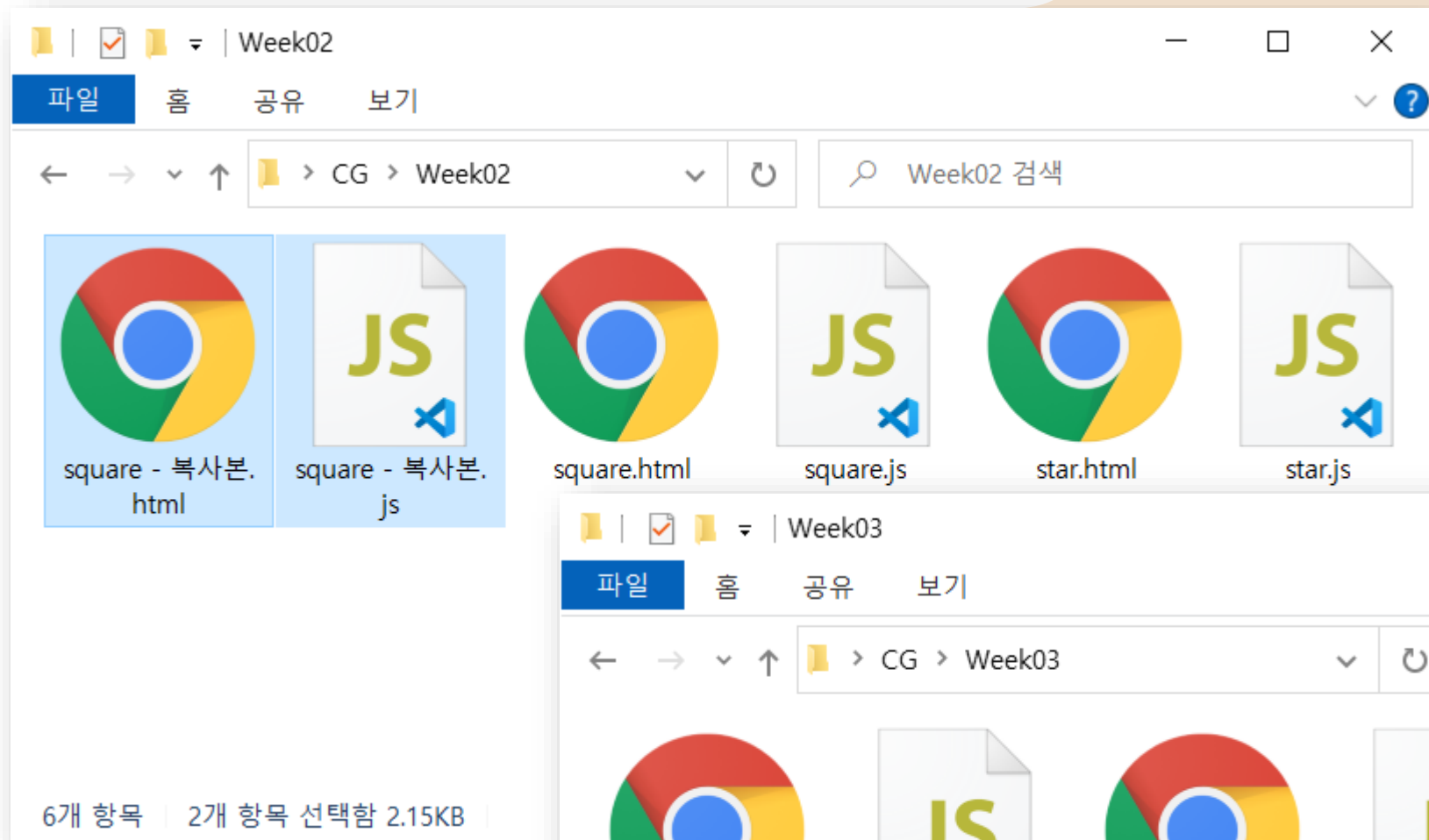
File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html   JS gasket.js   gasket2d.html   JS gasket2d.js   colorPolygon.html ✕   JS colorPolygon.js

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > colorPolygon.html > html > head > script

```html
1    <!DOCTYPE html>
2    <html>
3        <head>
4            <title>2022 Computer Graphics</title>
5
6            <script id="vertex-shader" type="x-shader/x-vertex">
7            attribute vec4 vPosition;
8
9            void main() {
10               //gl_PointSize = 5.0;
11               gl_Position = vPosition;
12           }
13           </script>
14
15           <script id="fragment-shader" type="x-shader/x-fragment">
16           precision mediump float;
17
18           void main() {
19               gl_FragColor = vec4(1.0, 1.0, 0.0, 1.0);
20           }
21           </script>
22
23           <script type="text/javascript" src="../Common/webgl-utils.js"></script>
24           <script type="text/javascript" src="../Common/initShaders.js"></script>
25           <script type="text/javascript" src="../Common/MV.js"></script>
26           <script type="text/javascript" src="colorPolygon.js"></script>
27       </head>
28       <body>
29           <canvas id="gl-canvas" width="512" height="512">
30               Oops... your browser doesn't support the HTML5 canvas element!
31           </canvas>
32       </body>
33   </html>
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 26, Col 60   Spaces: 4   UTF-8   CRLF   HTML

32

File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html   gasket.js   gasket2d.html   gasket2d.js   colorPolygon.html   colorPolygon.js ✕

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS colorPolygon.js > ⬡ render

```javascript
1    var gl;
2
3    window.onload = function init()
4    {
5        var canvas = document.getElementById("gl-canvas");
6
7        gl = WebGLUtils.setupWebGL(canvas);
8        if( !gl ) {
9            alert("WebGL isn't available!");
10       }
11
12       var vertices = [
13           vec2(-0.5, -0.5),
14           vec2(0.5, -0.5),
15           vec2(0, 0.5)
16       ];
17
18       // Configure WebGL
19       gl.viewport(0, 0, canvas.width, canvas.height);
20       gl.clearColor(1.0, 1.0, 1.0, 1.0);
21
22       // Load shaders and initialize attribute buffers
23       var program = initShaders(gl, "vertex-shader", "fragment-shader");
24       gl.useProgram(program);
25
26       // Load the data into the GPU
27       var bufferId = gl.createBuffer();
28       gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
29       gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);
30
31       // Associate our shader variables with our data buffer
32       var vPosition = gl.getAttribLocation(program, "vPosition");
33       gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
34       gl.enableVertexAttribArray(vPosition);
35
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 42, Col 31   Spaces: 4   UTF-8   CRLF   {} JavaScript

33

```js
    var vertices = [
        vec2(-0.5, -0.5),
        vec2(0.5, -0.5),
        vec2(0, 0.5)
    ];

    // 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(vertices), gl.STATIC_DRAW);

    // Associate our shader variables with our data buffer
    var vPosition = gl.getAttribLocation(program, "vPosition");
    gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
    gl.enableVertexAttribArray(vPosition);

    render();
};

function render()
{
    gl.clear(gl.COLOR_BUFFER_BIT);
    gl.drawArrays(gl.TRIANGLES, 0, 3);
}
```

34
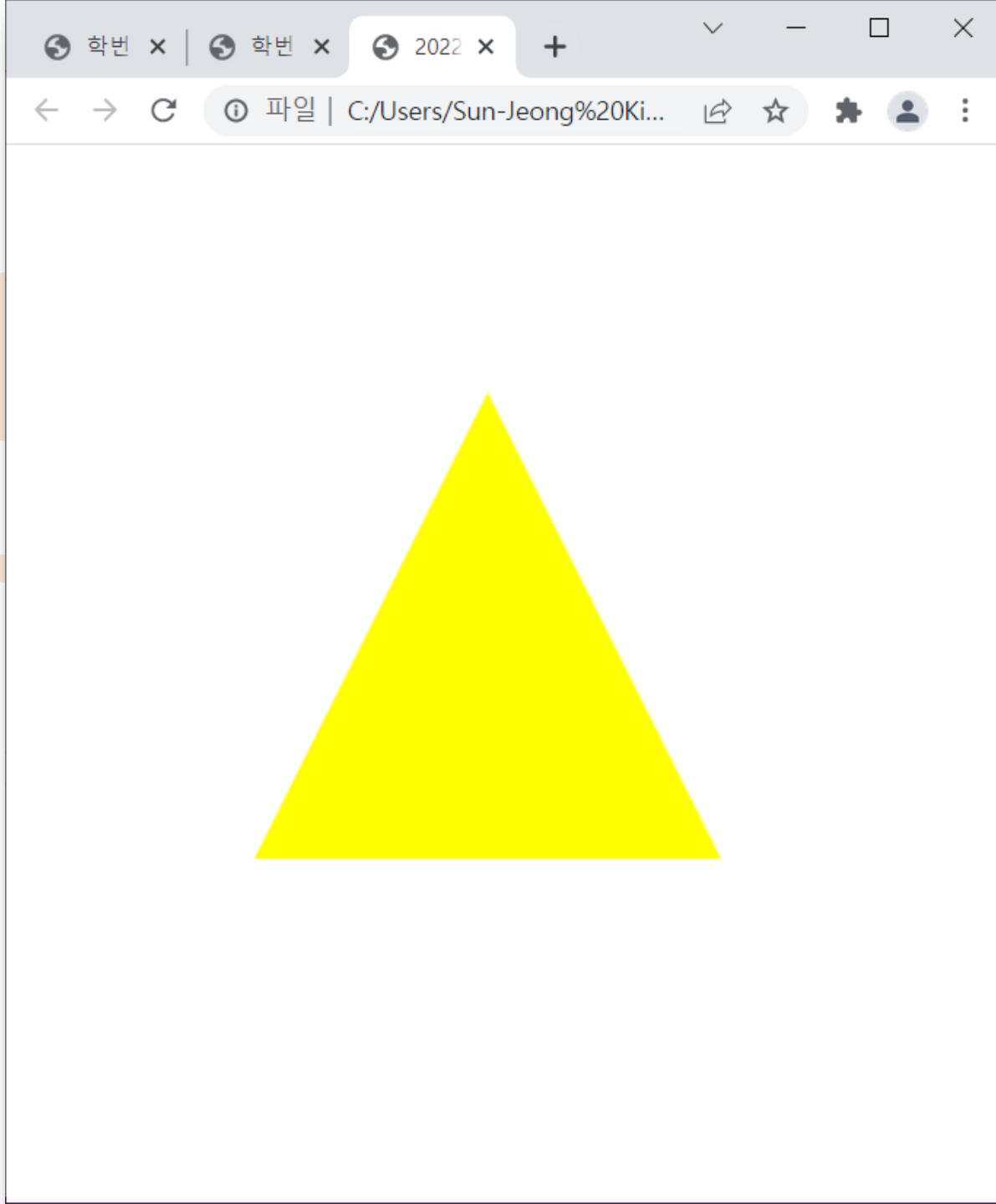
File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html | gasket.js | gasket2d.html | gasket2d.js | colorPolygon.html × | colorPolygon.js

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > <> colorPolygon.html > ⬡ html > ⬡ head > ⬡ script#fragment-shader

```html
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <title>2022 Computer Graphics</title>
5
6           <script id="vertex-shader" type="x-shader/x-vertex">
7                   attribute vec4 vPosition;
8                   attribute vec4 vColor;
9                   varying vec4 fColor;
10
11              void main() {
12                  //gl_PointSize = 5.0;
13                  gl_Position = vPosition;
14                  fColor = vColor;
15              }
16          </script>
17
18          <script id="fragment-shader" type="x-shader/x-fragment">
19          precision mediump float;
20              varying vec4 fColor;
21
22              void main() {
23                  gl_FragColor = fColor;
24              }
25          </script>
26
27      <script type="text/javascript" src="../Common/webgl-utils.js"></script>
28      <script type="text/javascript" src="../Common/initShaders.js"></script>
29      <script type="text/javascript" src="../Common/MV.js"></script>
30      <script type="text/javascript" src="colorPolygon.js"></script>
31      </head>
32      <body>
33          <canvas id="gl-canvas" width="512" height="512">
34              Oops... your browser doesn't support the HTML5 canvas element!
35          </canvas>
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 23, Col 39   Spaces: 4   UTF-8   CRLF   HTML

36

File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html   gasket.js   gasket2d.html   gasket2d.js   colorPolygon.html   colorPolygon.js ✕

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS colorPolygon.js > ✿ init

```javascript
1    var gl;
2
3    window.onload = function init()
4    {
5        var canvas = document.getElementById("gl-canvas");
6
7        gl = WebGLUtils.setupWebGL(canvas);
8        if( !gl ) {
9            alert("WebGL isn't available!");
10       }
11
12       var vertices = [
13           vec2(-0.5, -0.5),
14           vec2(0.5, -0.5),
15           vec2(0, 0.5)
16       ];
17
18       var colors = [
19           vec4(1, 0, 0, 1),
20           vec4(0, 1, 0, 1),
21           vec4(0, 0, 1, 1)
22       ];
23
24       // Configure WebGL
25       gl.viewport(0, 0, canvas.width, canvas.height);
26       gl.clearColor(1.0, 1.0, 1.0, 1.0);
27
28       // Load shaders and initialize attribute buffers
29       var program = initShaders(gl, "vertex-shader", "fragment-shader");
30       gl.useProgram(program);
31
32       // Load the data into the GPU
33       var bufferId = gl.createBuffer();
34       gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
35       gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 50, Col 40   Spaces: 4   UTF-8   CRLF   {} JavaScript

37

File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html   gasket.js   gasket2d.html   gasket2d.js   colorPolygon.html   colorPolygon.js ×

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS colorPolygon.js > init

```javascript
27
28        // Load shaders and initialize attribute buffers
29        var program = initShaders(gl, "vertex-shader", "fragment-shader");
30        gl.useProgram(program);
31
32        // Load the data into the GPU
33        var bufferId = gl.createBuffer();
34        gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
35        gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);
36
37        // Associate our shader variables with our data buffer
38        var vPosition = gl.getAttribLocation(program, "vPosition");
39        gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
40        gl.enableVertexAttribArray(vPosition);
41
42        // Create a buffer object, initialize it, and associate it with
43        // the associated attribute variable in our vertex shader
44        var cbufferId = gl.createBuffer();
45        gl.bindBuffer(gl.ARRAY_BUFFER, cbufferId);
46        gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
47
48        var vColor = gl.getAttribLocation(program, "vColor");
49        gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
50        gl.enableVertexAttribArray(vColor);
51
52        render();
53    };
54
55    function render()
56    {
57        gl.clear(gl.COLOR_BUFFER_BIT);
58        gl.drawArrays(gl.TRIANGLES, 0, 3);
59    }
60
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 50, Col 39   Spaces: 4   UTF-8   CRLF   {} JavaScript

38
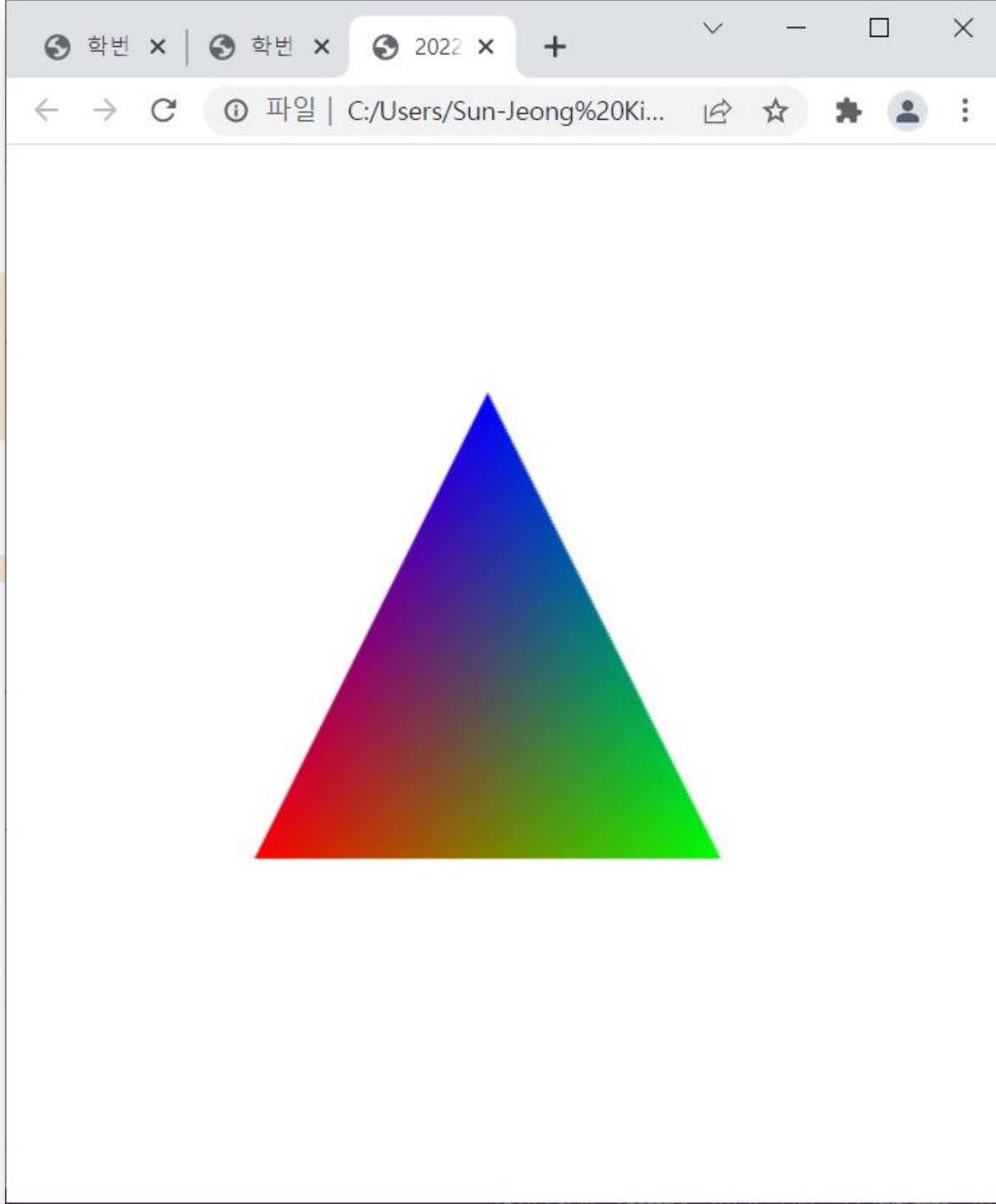
연습문제 (3)

- 오른쪽과 같은 사각형을 그리시오.

연습문제 (4)

- 오른쪽과 같은 사각형의 테두리 선을 그리시오.
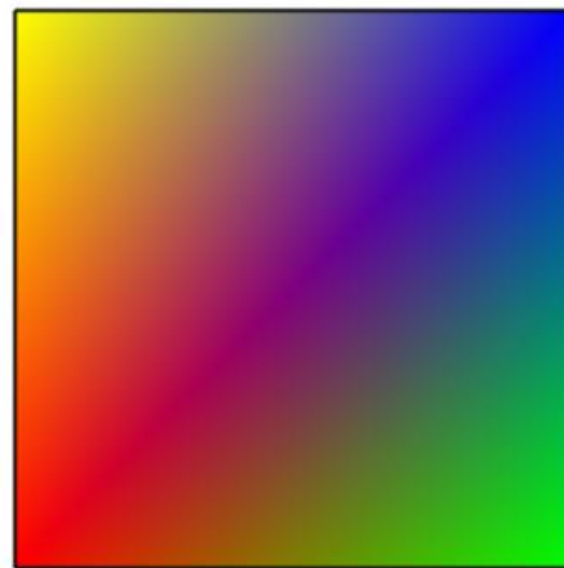
File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html   gasket.js   gasket2d.html   gasket2d.js   colorPolygon.html   colorPolygon.js ✕

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS colorPolygon.js > ⬡ render
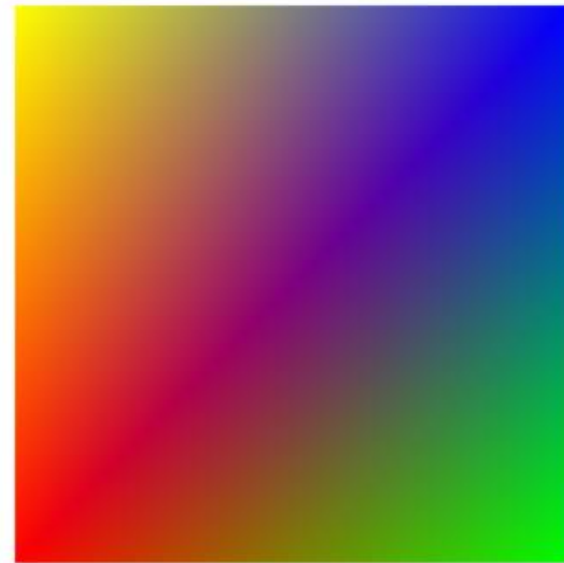
```javascript
1   var gl;
2
3   window.onload = function init()
4   {
5       var canvas = document.getElementById("gl-canvas");
6
7       gl = WebGLUtils.setupWebGL(canvas);
8       if( !gl ) {
9           alert("WebGL isn't available!");
10      }
11
12      var vertices = [
13          vec2(-0.5, -0.5),
14          vec2(0.5, -0.5),
15          vec2(0.5, 0.5),
16          vec2(-0.5, 0.5),
17
18          vec2(-0.5, -0.5),
19          vec2(0.5, -0.5),
20          vec2(0.5, 0.5),
21          vec2(-0.5, 0.5)
22      ];
23
24      var colors = [
25          vec4(1, 0, 0, 1),
26          vec4(0, 1, 0, 1),
27          vec4(0, 0, 1, 1),
28          vec4(1, 1, 0, 1),
29
30          vec4(0, 0, 0, 1),
31          vec4(0, 0, 0, 1),
32          vec4(0, 0, 0, 1),
33          vec4(0, 0, 0, 1)
34      ];
35
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 71, Col 31   Spaces: 4   UTF-8   CRLF   {} JavaScript

42

```javascript
        // 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(vertices), gl.STATIC_DRAW);

        // Associate our shader variables with our data buffer
        var vPosition = gl.getAttribLocation(program, "vPosition");
        gl.vertexAttribPointer(vPosition, 2, 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);

        render();
};

function render()
{
        gl.clear(gl.COLOR_BUFFER_BIT);
        gl.drawArrays(gl.TRIANGLE_FAN, 0, 4);
        gl.drawArrays(gl.LINE_LOOP, 4, 4);
}
```
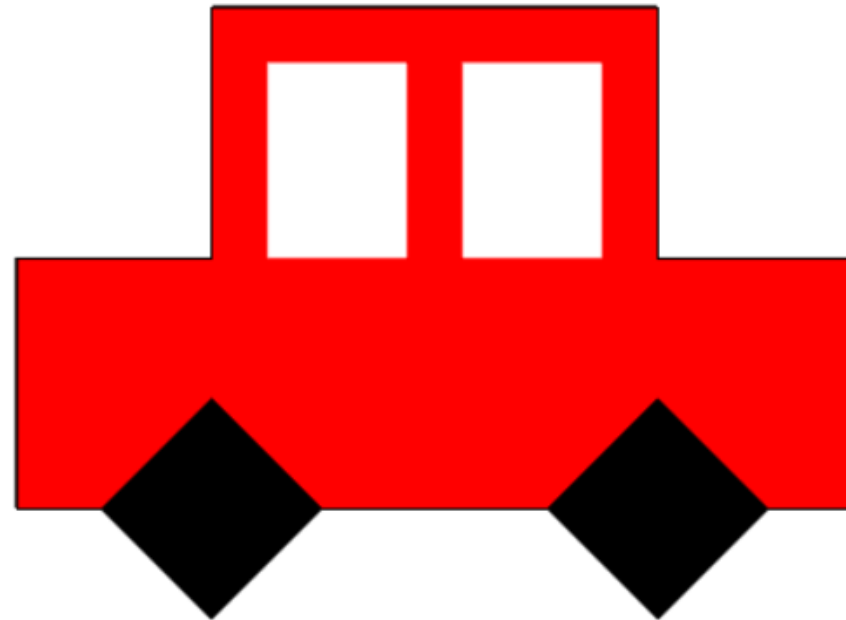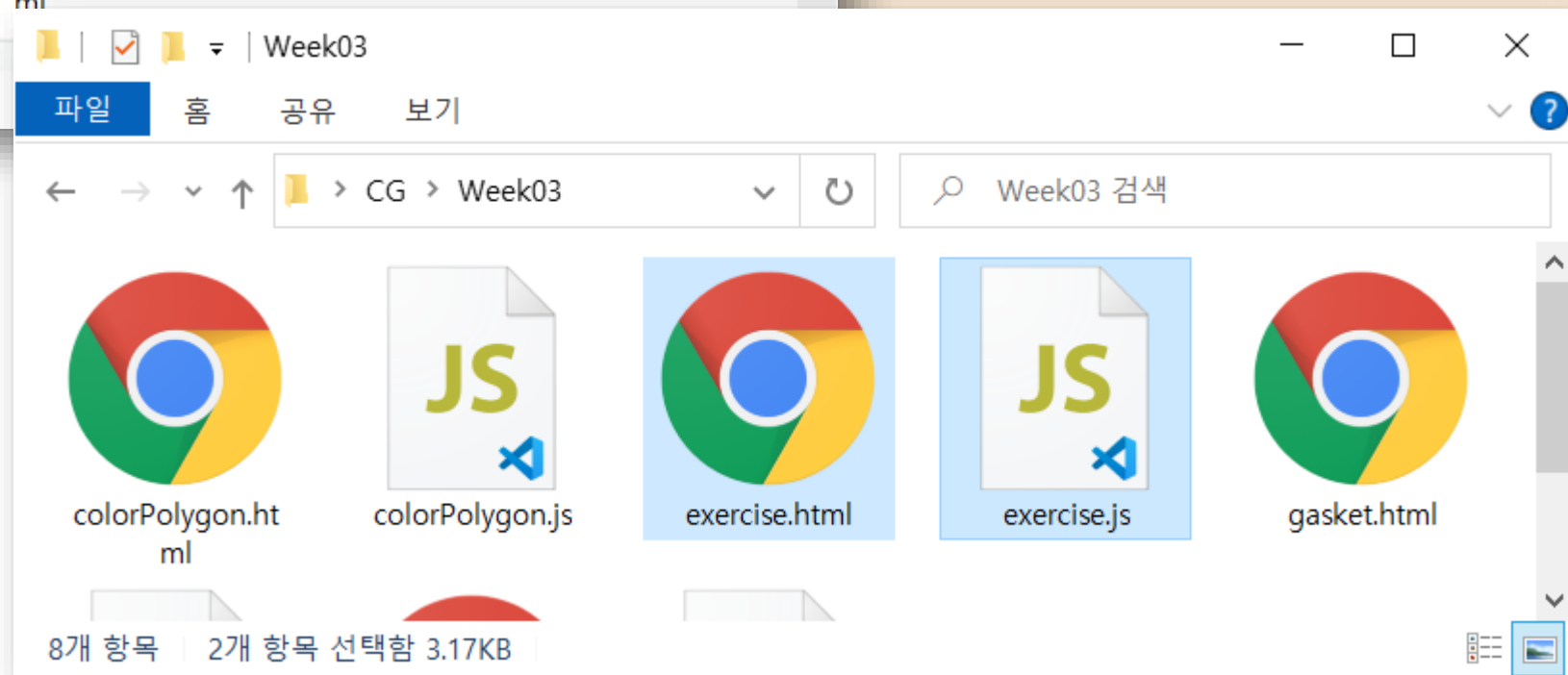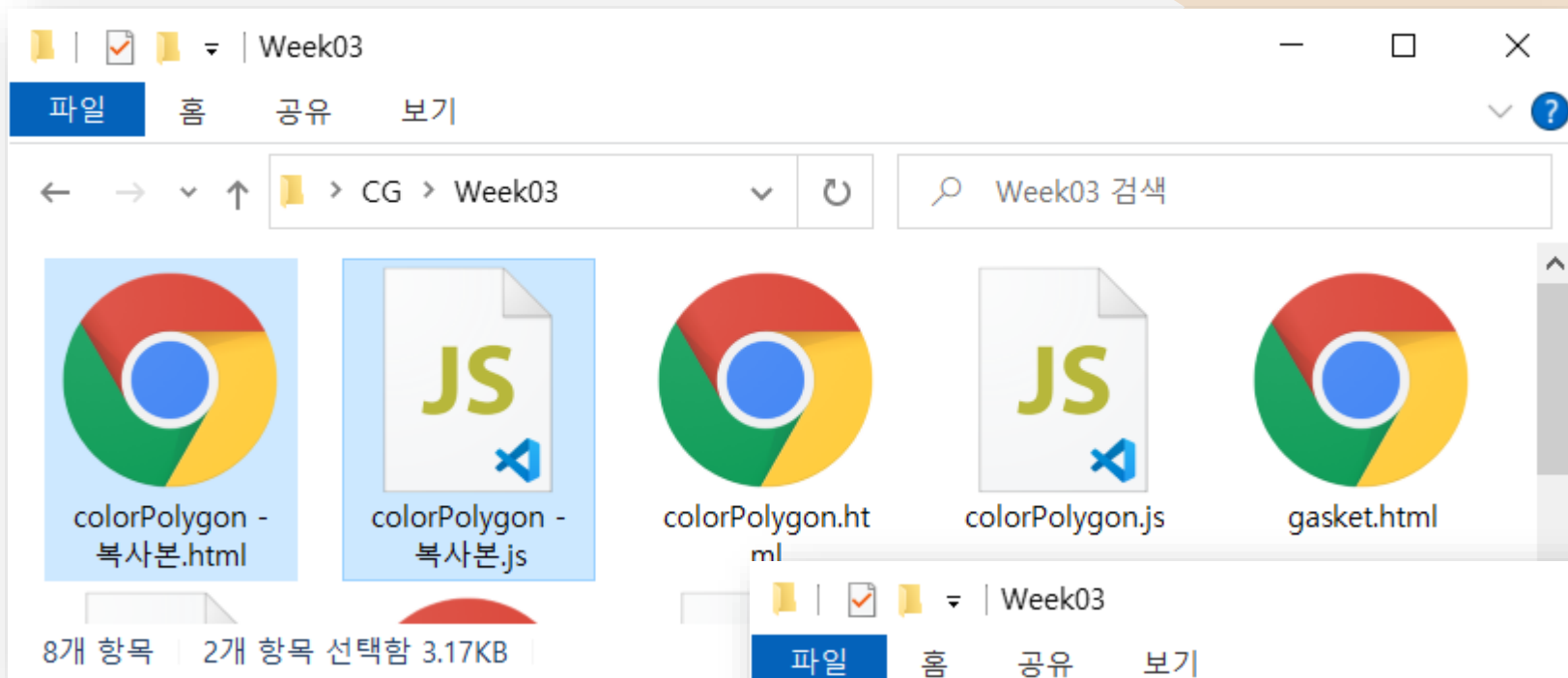
(handwritten annotations) 인덱스 0부터 4개 정상출 ; 인덱스 주의!

43

연습문제 (5)

- 자유 그리기
  - 도형 5개 이상
  - 색상 3개 이상

File   Edit   Selection   View   Go   Run   Terminal   Help

gasket.html   gasket.js   gasket2d.html   gasket2d.js   colorPolygon.html   colorPolygon.js   exercise.html ✕   exe

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > <> exercise.html > ⬡ html > ⬡ head > ⬡ script

```html
1   <!DOCTYPE html>
2   <html>
3       <head>
4           <title>2022 Computer Graphics</title>
5
6           <script id="vertex-shader" type="x-shader/x-vertex">
7               attribute vec4 vPosition;
8               attribute vec4 vColor;
9               varying vec4 fColor;
10
11              void main() {
12                  //gl_PointSize = 5.0;
13                  gl_Position = vPosition;
14                  fColor = vColor;
15              }
16          </script>
17
18          <script id="fragment-shader" type="x-shader/x-fragment">
19              precision mediump float;
20              varying vec4 fColor;
21
22              void main() {
23                  gl_FragColor = fColor;
24              }
25          </script>
26
27          <script type="text/javascript" src="../Common/webgl-utils.js"></script>
28          <script type="text/javascript" src="../Common/initShaders.js"></script>
29          <script type="text/javascript" src="../Common/MV.js"></script>
30          <script type="text/javascript" src="exercise.js"></script>
31      </head>
32      <body>
33          <canvas id="gl-canvas" width="512" height="512">
34              Oops... your browser doesn't support the HTML5 canvas element!
35          </canvas>
```

Restricted Mode   ⊗ 0 ⚠ 0   Ln 30, Col 56   Spaces: 4   UTF-8   CRLF   HTML

46

```
1    var gl;
2
3    window.onload = function init()
4    {
5        var canvas = document.getElementById("gl-canvas");
6
7        gl = WebGLUtils.setupWebGL(canvas);
8        if( !gl ) {
9            alert("WebGL isn't available!");
10       }
11
12       var vertices = [
13           vec2(-0.4, 0.75),    vec2(-0.4, 0.3),
14           vec2(0.4, 0.3),      vec2(0.4, 0.75),
15
16           vec2(-0.75, 0.3),    vec2(-0.75, -0.15),
17           vec2(0.75, -0.15),   vec2(0.75, 0.3)
18       ];
19
20       var colors = [
21           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
22           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
23
24           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
25           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1)
26       ];
27
28       // Configure WebGL
29       gl.viewport(0, 0, canvas.width, canvas.height);
30       gl.clearColor(1.0, 1.0, 1.0, 1.0);
31
32       // Load shaders and initialize attribute buffers
33       var program = initShaders(gl, "vertex-shader", "fragment-shader");
34       gl.useProgram(program);
35
```

47

File Edit Selection View Go Run Terminal Help

.html   JS gasket.js   <> gasket2d.html   JS gasket2d.js   <> colorPolygon.html   JS colorPolygon.js   <> exercise.html   JS exercise.js

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS exercise.js > ⬡ render

```javascript
36          // Load the data into the GPU
37          var bufferId = gl.createBuffer();
38          gl.bindBuffer(gl.ARRAY_BUFFER, bufferId);
39          gl.bufferData(gl.ARRAY_BUFFER, flatten(vertices), gl.STATIC_DRAW);
40
41          // Associate our shader variables with our data buffer
42          var vPosition = gl.getAttribLocation(program, "vPosition");
43          gl.vertexAttribPointer(vPosition, 2, gl.FLOAT, false, 0, 0);
44          gl.enableVertexAttribArray(vPosition);
45
46          // Create a buffer object, initialize it, and associate it with
47          // the associated attribute variable in our vertex shader
48          var cbufferId = gl.createBuffer();
49          gl.bindBuffer(gl.ARRAY_BUFFER, cbufferId);
50          gl.bufferData(gl.ARRAY_BUFFER, flatten(colors), gl.STATIC_DRAW);
51
52          var vColor = gl.getAttribLocation(program, "vColor");
53          gl.vertexAttribPointer(vColor, 4, gl.FLOAT, false, 0, 0);
54          gl.enableVertexAttribArray(vColor);
55
56          render();
57      };
58
59      function render()
60      {
61          gl.clear(gl.COLOR_BUFFER_BIT);
62          gl.drawArrays(gl.TRIANGLE_FAN, 0, 4);
63          gl.drawArrays(gl.TRIANGLE_FAN, 4, 4);
64      }
65
```

File   Edit   Selection   View   Go   Run   Terminal   Help
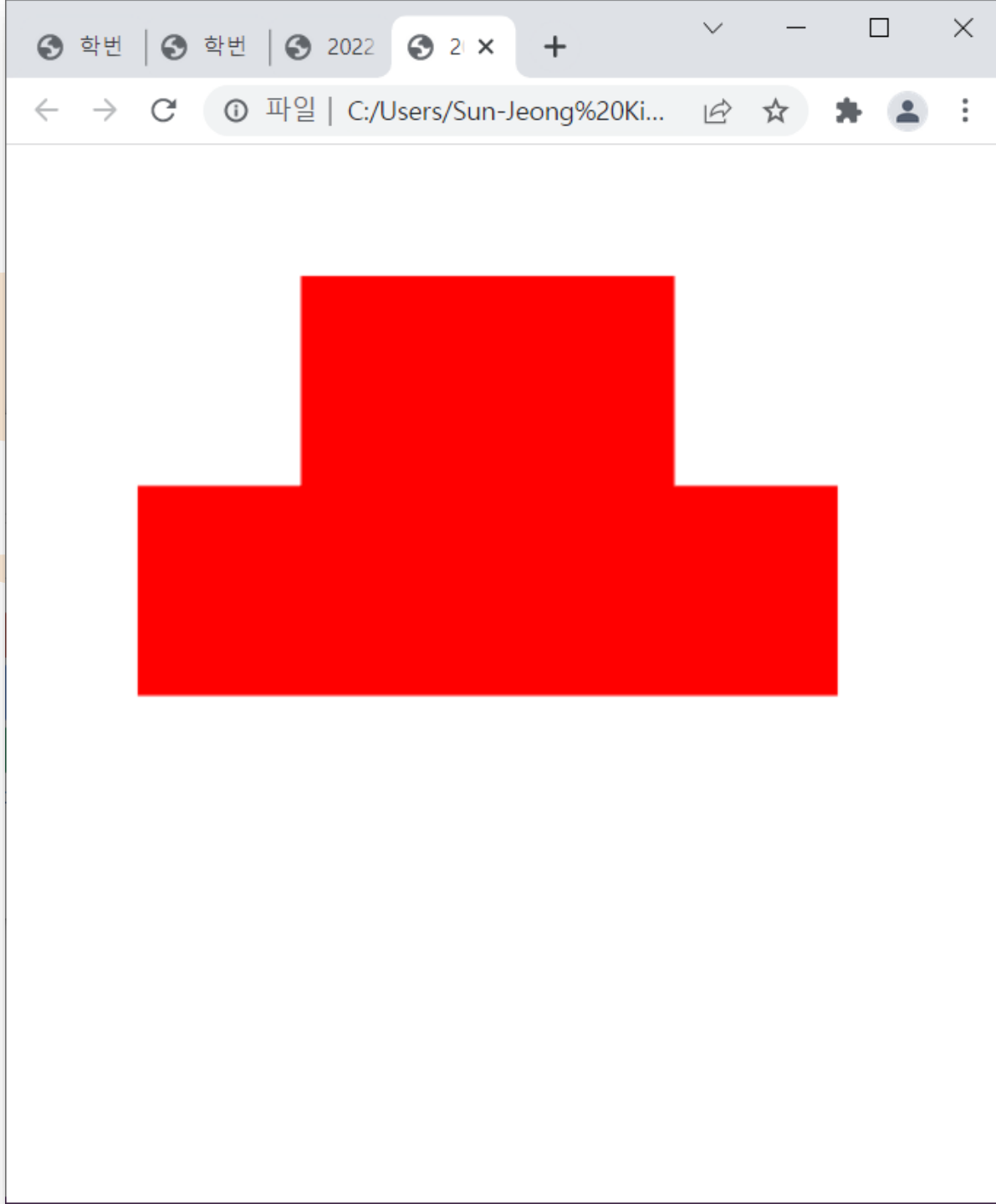
.html   JS gasket.js   <> gasket2d.html   JS gasket2d.js   <> colorPolygon.html   JS colorPolygon.js   <> exercise.html   JS exercise.js ✕

C: > Users > Sun-Jeong Kim > Desktop > CG > Week03 > JS exercise.js > 🏷 render

```javascript
1    var gl;
2
3    window.onload = function init()
4    {
5        var canvas = document.getElementById("gl-canvas");
6
7        gl = WebGLUtils.setupWebGL(canvas);
8        if( !gl ) {
9            alert("WebGL isn't available!");
10       }
11
12       var vertices = [
13           vec2(-0.4, 0.75),    vec2(-0.4, 0.3),
14           vec2(0.4, 0.3),      vec2(0.4, 0.75),
15
16           vec2(-0.75, 0.3),    vec2(-0.75, -0.15),
17           vec2(0.75, -0.15),   vec2(0.75, 0.3),
18
19           vec2(-0.4, 0.75),    vec2(-0.4, 0.3),      vec2(-0.75, 0.3),    vec2(-0.75, -0.15),
20           vec2(0.75, -0.15),   vec2(0.75, 0.3),      vec2(0.4, 0.3),      vec2(0.4, 0.75)
21       ];
22
23       var colors = [
24           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
25           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
26
27           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
28           vec4(1, 0, 0, 1),    vec4(1, 0, 0, 1),
29
30           vec4(0, 0, 0, 1),    vec4(0, 0, 0, 1),    vec4(0, 0, 0, 1),    vec4(0, 0, 0, 1),
31           vec4(0, 0, 0, 1),    vec4(0, 0, 0, 1),    vec4(0, 0, 0, 1),    vec4(0, 0, 0, 1)
32       ];
33
34       // Configure WebGL
35       gl.viewport(0, 0, canvas.width, canvas.height);
```

🛡 Restricted Mode   ⊗ 0 ⚠ 0   Ln 70, Col 37   Spaces: 4   UTF-8   CRLF   {} JavaScript

50

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

        // Associate our shader variables with our data buffer
        var vPosition = gl.getAttribLocation(program, "vPosition");
        gl.vertexAttribPointer(vPosition, 2, 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);

        render();
    };

    function render()
    {
        gl.clear(gl.COLOR_BUFFER_BIT);
        gl.drawArrays(gl.TRIANGLE_FAN, 0, 4);
        gl.drawArrays(gl.TRIANGLE_FAN, 4, 4);
        gl.drawArrays(gl.LINE_LOOP, 8, 8);
    }
```