REPORT

과 목 명 : 컴퓨터그래픽스
담담교수 : 송인식 교수님
소　　속 : 소프트웨어학과
학　　번 : 32151671
이　　름 : 박민혁

단국대학교
Dankook University

1. homework1.html

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;charset=utf-8" >
<title>2D Sierpinski Gasket</title>


<script id="vertex-shader" type="x-shader/x-vertex">
attribute vec4 vPosition;

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

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

void
main()
{
    gl_FragColor = vec4( 0.1, 0.5, 0.4 ,1.0 );
}
</script>

<script type="text/javascript" src="webgl-utils.js"></script>
<script type="text/javascript" src="initShaders.js"></script>
<script type="text/javascript" src="MV.js"></script>
<script type="text/javascript" src="homework1.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>
```

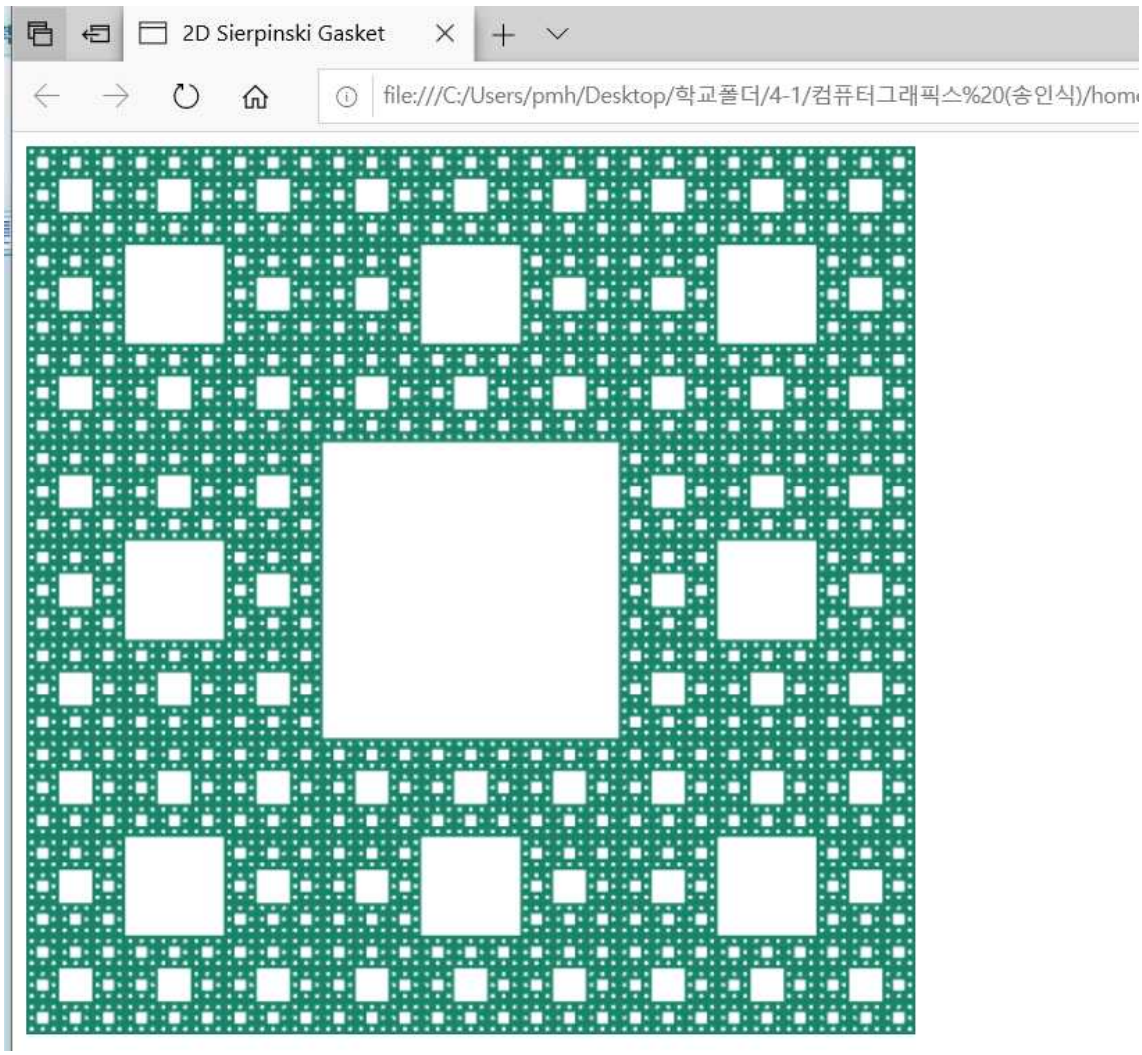
2. homework1.js

```
"use strict";
var canvas;
var gl;
var points = [];
var NumTimesToSubdivide = 5;
window.onload = function init()
{    canvas = document.getElementById( "gl-canvas" );
    gl = WebGLUtils.setupWebGL( canvas );
    if ( !gl ) { alert( "WebGL isn't available" ); }
    //
    //  Initialize our data for the Sierpinski Gasket
    //
    // First, initialize the corners of our sierpinski carpet with four points.
    var vertices = [
        vec2( -1, -1 ),
        vec2( -1,  1 ),
        vec2(  1,  1 ),
        vec2(  1, -1 ),
    ];
   divideSquare( vertices[0], vertices[1], vertices[2],vertices[3],
                        NumTimesToSubdivide);
    //
    //  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 out 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 Square( x, y, z, w)
{       points.push( x, y, z );
        points.push( x, w, z );
}function divideSquare( x, y, z,w, count )
{       // check for end of recursion
        if ( count === 0 ) {
                Square( x, y, z ,w);
        }
        else {
                //bisect the sides
                var xy = mix( x, y, 1/3 );
                var xy1 = mix(x, y, 2/3)
                var yz = mix( y, z, 1/3 );
                var yz1 = mix( y, z, 2/3 );
                var zw = mix( z, w, 1/3 );
                var zw1 = mix( z, w, 2/3 );
                var wx = mix( w, x, 1/3 );
                var wx1 = mix( w, x, 2/3 );
                var wxyz1 = mix( wx1, yz, 1/3 );
                var wxyz2 = mix( wx1, yz, 2/3 );
                var wxyz3 = mix( wx, yz1, 1/3 );
                var wxyz4 = mix( wx, yz1, 2/3 );
                --count;
                // Eight new Square
                divideSquare( wxyz4, yz1, z, zw, count );
                divideSquare( wxyz3, wxyz4, zw,zw1, count );
                divideSquare( wx,wxyz3, zw1, w, count );
                divideSquare( wx1,wxyz1, wxyz3, wx, count );
                divideSquare( x, xy, wxyz1,wx1, count );
                divideSquare( xy, xy1, wxyz2, wxyz1, count );
                divideSquare( xy1,y, yz, wxyz2, count );
                divideSquare( wxyz2, yz, yz1, wxyz4, count );

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

3. Result


2D Sierpinski Gasket
file:///C:/Users/pmh/Desktop/학교폴더/4-1/컴퓨터그래픽스%20(송인식)/hom