REPORT

17290079.html:

```
</script>
<!-- Fragment Shader -->
<script id = "fragment-shader" type = "x-shader/x-fragment">
    precision mediump float;
        uniform vec4 fColor;

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

17290079.js:

```
    fColorLocation = gl.getUniformLocation(program, "fColor");
```

in render() function:

```
  gl.uniform4f(fColorLocation, RED, GREEN, BLUE, 1.0);
```

CREATING LETTERS

Letter color:

I create fragment shader in html file to determine the letter color . In javascript file I create fColorLocation variable as a globel variable(because I use this variable in render fuction). Then in render function I use gl.Uniform4f(fcolorLocation,RED,BLUE,GREEN,1). To use in this function RED,BLUE,GREEN I define these variable
I put some default value into these variables to create starting letter color.

17290079.html

```
<!-- Vertex Shader -->
<script id = "vertex-shader" type = "x-shader/x-vertex">
    uniform vec4 translation;
    uniform mat4 scale;
    attribute vec4 vPosition;

    uniform float theta;
    void main(){


        gl_Position.x =(vPosition.x * cos(theta) - vPosition.y * sin(theta)+translation.x)*scale[0][0];

        gl_Position.y = (vPosition.x * sin(theta) + vPosition.y * cos(theta)+translation.y)*scale[1][1];
```

Letter Position:

I create vertex shader in html file and I define uniforms and an attribute. “vPositon” vector is used to define current position on cordinate system. We will use it to change the rotation, location and scale size. In js file vPosition variable which get the position value from vertex shader in html file. Finally using gl.enableVertexAttribArray(vPosition) function , we make vPosition enable to use.

```
        gl_Position.z = 0.0 ;

        gl_Position.w = 1.0 ;
 }
</script>

17290079.js:
vPosition = gl.getAttribLocation(program,
"vPosition");
    gl.vertexAttribPointer(vPosition, 2,
gl.FLOAT, false, 0, 0);
 gl.enableVertexAttribArray(vPosition);
```

```
17290079.js

vertices= [vec2( -0.6,0.8 ),
    vec2( -0.6,0.7 ),
    vec2( -0.1,0.8 ),
            .
            .
            .
]


const bufferID = gl.createBuffer();
    gl.bindBuffer( gl.ARRAY_BUFFER,
bufferID ); // bind this buffer to this canvas
    /* put coordinates into the buffer */
    gl.bufferData( gl.ARRAY_BUFFER,
flatten(vertices), gl.STATIC_DRAW );

render()  function
gl.drawArrays( gl.TRIANGLES, 0, 72);
```

Creating Letter Forms

I determined points(vec2(x,y)) on the coordinate system by using vertices vector and connected these points to form triangles. I create a buffer and then I bind the buffer and then finally I put the vertices which I created into buffer data. To make triangles I use gl.drawArrays(gl.TRIANGLES,0,72) function. WhenThese triangles are created ,it looks like 2 letters Which are Ş and Ğ on the screen.

17290079.html

```
<select id="color" size="6">
<option value="0">
          change color to red
</option>
<option value="1">change color to
green</option>
<option value="2">change color to
blue</option>
<option value="3">change color to
yellow</option>
<option value="5">change color to
purple</option>
<option value="6">change color to
white</option>
</select>

var m = document.getElementById("color");
   m.addEventListener("click", function() {
       switch (m.selectedIndex) {
           case 0:
  RED=1.0;GREEN=0.0;BLUE=0.0;
              break;
           case 1:
RED=0.0;GREEN=0.250;BLUE=0.150;
              break;
           case 2:
 RED=0.0;GREEN=0.0;BLUE=1.0;
              break;
           case 3:
          RED=1.0;GREEN=1.0;BLUE=0.0;
                 break;
           case 4:
   RED=0.52;GREEN=0.0;BLUE=1.0;
                    break;
            case 5:
  RED=1.0;GREEN=1.0;BLUE=1.0;
                  break;
                 }
    });

fColorLocation =
gl.getUniformLocation(program, "fColor");

render() function:
gl.uniform4f(fColorLocation, RED, GREEN,
BLUE, 1.0);
```

CHANGING COLOR

I create selection table in html file. Put them some color options like red, blue, green, yellow, white, purple into this table. To active the table I also create m variable to get the color id when user click , then by using this color id program know the which case it should run.

In these case include green, blue, red value to make the letters desired color . I mentioned about fragment shader , I need reach the fragment shader to determine letter color. Therefore,

I use fColorLocation variable which include fcolor vector come from fragment shader.

Finally in render function I have gl.Uniform4f(fcolorLocation,RED,BLUE,GREEN,1) function and it allow us to change the color that user want.

17290079.html:

```
<!-- Vertex Shader -->
<script id = "vertex-shader" type = "x-
shader/x-vertex">
   uniform vec4 translation;
   uniform mat4 scale;
   attribute vec4 vPosition;

   uniform float theta;
   void main(){


  gl_Position.x =(vPosition.x * cos(theta) -
vPosition.y *
sin(theta)+translation.x)*scale[0][0];

       gl_Position.y = (vPosition.x * sin(theta) +
vPosition.y *
cos(theta)+translation.y)*scale[1][1];

gl_Position.z = 0.0 ;
gl_Position.w = 1.0 ;
}
</script>
```

17290079.js

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


render() function:
 if(rotate!=0){
   theta += (CheckDirection ? -0.1 : 0.1);

    gl.uniform1f( thetaLoc, theta );// rotate=0;
}
```

CHANGING ROTATION

When I create letter, I used vertex shader. I also use it to rotate the letter. I create the theta variable as an angle. I use this angle to determine next x and y coordinate of the next position after it is rotated. To calculate the position I make some algebric calculation in vertex shader.

I define theta variable with 0.0 default value.
I also define thetaLoc pointer it gets theta from vertex shader in html file. In render function I change the theta value and put the new value into thetaLoc pointer and current angle value is used in vertex shader.

17290079.html:

```
    <div>
       rotating_speed 100 <input id="slide"
type="range"
       min="0" max="1000" step="100"
value="500" />
       0
    </div>
```

17290079.js:

```
document.getElementById("slide").onchange
= function() {delay = this.value; }
```

Render() function:

```
function render(){
setTimeout( function(){
.
.
.
},delay);

}
```

Rotating_speed

I create a slider to arrange the speed of rotation. In html file I use a value variable it take the value from user and program use it as a delay value in js file. This delay value determine the speed of rotation and it used in render function(setTimeout(function(){..}),delay).

17290079.html:

```
   <button id = "startButton">
                    start Rotation
    </button>
```

17290079.js

```
  var myButton2 =
document.getElementById("startButton");
   myButton2.addEventListener("click",
function() {rotate=1;});
```

render() function:

```
 if(rotate!=0){
  theta += (CheckDirection ? -0.1 : 0.1);

   gl.uniform1f( thetaLoc, theta );// rotate=0;
}
```

Start Rotation

There is a start rotation button which allows us to start rotating. In html file I define syntax of button and I determine the id as startButton. I also define syntax of this button and rotate variable with the default value of 0 in js file. I use the rotate variable to control for rotation. When user click the button, program change the value of rotation as 1 and if statement allows us to use code stuff inside if scopes.

17290079.html:

```
   <button id = "stopButton">
                  stop Rotation
    </button>
```

17290079.js

```
  var myButton3 =
document.getElementById("stopButton");
   myButton3.addEventListener("click",
function() {rotate=0;});


render() function:
 if(rotate!=0){
  theta += (CheckDirection ? -0.1 : 0.1);

   gl.uniform1f( thetaLoc, theta );// rotate=0;
}
```

Stop Rotation

There is a start rotation button which allows us to stop rotating. In html file I define syntax of button and I determine the id as stopButton. In js file I define syntax of stop button. I used rotate variable as I mentioned and this variable is also used for stopping rotation. . When user click the button, program change the value of rotation as 0 and if statement doesn't allow us to use code stuff inside if scopes and rotation stops.

17290079.html:

```
   <button id = "DirectionButton">
                  Change Rotation Direction
    </button>
```

17290079.js

```
    var myButton =
document.getElementById("DirectionButton");
   myButton.addEventListener("click",
function() {CheckDirection =
!CheckDirection;});


render() function:
 if(rotate!=0){
  theta += (CheckDirection ? -0.1 : 0.1);

   gl.uniform1f( thetaLoc, theta );// rotate=0;
}
```

Change Rotating Direction

There is a Change Rotation Direction button which allows us change rotating direnction. I define syntax of the button in html and js file. In js file I define a boolean variable which is CheckDirection. This variable determine the direction of the rotation in render function.

17290079.html

```
<!-- Vertex Shader -->
<script id = "vertex-shader" type = "x-
shader/x-vertex">
   uniform vec4 translation;
   uniform mat4 scale;
   attribute vec4 vPosition;

   uniform float theta;
   void main(){
      gl_Position.x =(vPosition.x * cos(theta) -
vPosition.y *
sin(theta)+translation.x)*scale[0][0];

      gl_Position.y = (vPosition.x * sin(theta) +
vPosition.y *
cos(theta)+translation.y)*scale[1][1];
gl_Position.z = 0.0 ;
gl_Position.w = 1.0 ;
}
           </script>


<button id = "Lscale">
                  scale larger
    </button>

<button id = "Sscale">
                  scale smaller
    </button>
```

17290079.js

```
   var scaleLargeButton =
document.getElementById("Lscale");
   scaleLargeButton.addEventListener("click",
function() { Sx+=0.1,Sy+=0.1;});

   var scaleSmallButton =
document.getElementById("Sscale");
   scaleSmallButton.addEventListener("click",
function() { Sx-=0.1,Sy-=0.1;});

Scale = gl.getUniformLocation(program,
'scale');
```

SCALING

There are 2 button to arrange scaling which are named scale larger and scale smaller button.
In vertex shader I define a uniform matrix called scale and then I use it when I calculate the position. I get the information from js file. In js file Sx,SY,Sz variables are defined to determine how much the size of letters increase or decrease. I create a Scale variable to connect with the scale variable which come from js file. In render function I create matrix by using Sx,SY,Sz variables and put the matrix in gl.uniform4v(Scale,false,Matrix).

I mentioned about larger and scale smaller button. I define defined the syntaxes in both js and html file. When you click scale larger button I increase the values of Sx and Sy, so letters increase in size. . When you click scale smaller button program decrease the values of Sx and Sy, so letters decrease in size.

```
render() function

 Matrix= new Float32Array([
       Sx,   0.0,  0.0,  0.0,
       0.0,  Sy,   0.0,  0.0,
       0.0,  0.0,  Sz,   0.0,
       0.0,  0.0,  0.0,  1.0
    ]);
   gl.uniformMatrix4fv(Scale, false, Matrix);
```

```
17290079.html
<!-- Vertex Shader -->
<script id = "vertex-shader" type = "x-
shader/x-vertex">
   uniform vec4 translation;
   uniform mat4 scale;
   attribute vec4 vPosition;

   uniform float theta;
   void main(){
      gl_Position.x =(vPosition.x * cos(theta) -
vPosition.y *
sin(theta)+translation.x)*scale[0][0];

      gl_Position.y = (vPosition.x * sin(theta) +
vPosition.y *
cos(theta)+translation.y)*scale[1][1];
gl_Position.z = 0.0 ;
gl_Position.w = 1.0 ;
}
            </script>
```

TRANSLATION

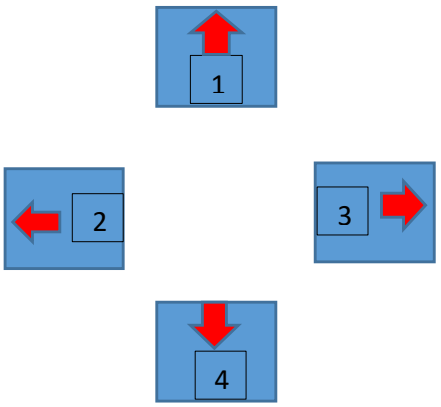
I use some keyboard key to translate letters.
The key 1 moves the letters up, the key 4 moves them down, the key 2 moves them to the left, the keys 3 to the right.
I use some algebric calculation for moving letters in vertex shader. Firstly I use a uniform vector named translation. This vector get information from js file about how far the letters are shifted in the coordinate system. In js file I create a translation variable which is connected with the translation vector coming from inside vertex shader. I define Tx, Ty, Tz variables which allows us to determine the final coordinates after the letters are shifted. Finally I use the function (gl.uniform4f(translation, Tx,Ty,Tz,0.0)) to send the information.

17290079.js

```
window.onkeydown =
   function(event) {
      var key =
String.fromCharCode(event.keyCode);

      switch (key) {
      case "1":

       Ty+=0.1;
           break;
         case "2":
          Tx-=0.1;
           break;
         case "3":
          Tx+=0.1;
           break;
         case "4":
          Ty-=0.1;
              break;
      }
   };
```




```
translation = gl.getUniformLocation(program,
'translation');

render() function

 gl.uniform4f(translation, Tx, Ty, Tz, 0.0);
```

ŞEYMA NUR ALTUNDAĞ

17290079