# CS 405 Project 3 Report

Ege Yurtsever
32253

## Task 1

Task 1 was to implement the draw method in the SceneNode class.

MatrixMult and getTransformationMatrix functions were given in the utilities, so it was an easy implementation by using them directly. A single matrix multiplication with the transformMatrix for all the parameters was enough.

Furthermore, the draw function had to recursively draw all the childs, while passing the parameters of the parent recursively.

You can see the implementation of the draw function below:

```
26      draw(mvp, modelView, normalMatrix, modelMatrix) {
27          /**
28           * @Task1 : Implement the draw function for the SceneNode class.
29           */
30
31          var transformMatrix = this.trs.getTransformationMatrix();
32
33          var transformedModel = MatrixMult(modelMatrix, transformMatrix);
34          var transformedModelView = MatrixMult(modelView, transformMatrix);
35          var transformedMvp = MatrixMult(mvp, transformMatrix);
36          var transformedNormals = MatrixMult(normalMatrix, transformMatrix);
37
38          // Draw the MeshDrawer
39          if (this.meshDrawer) {
40              this.meshDrawer.draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);
41          }
42
43          for (var i = 0; i < this.children.length; i++) {
44              this.children[i].draw(transformedMvp, transformedModelView, transformedNormals, transformedModel);
45          }
46      }
```

## Task 2

Task 2 was to implement a diffuse and specular lighting in the fragment shader.

Diffuse light depends on the light direction and the normal direction of the surface, a simple dot product is sufficient to implement it. Also we make sure it’s not negative by taking the max of the dot product and a 0 float.

Specular light can be calculated in world space or view space, but view space option is favorable here as we are not allowed to change any other part of the fragment shader and camera position/direction is not included in the file. Also the implementation of the specular lighting is fairly standard, I've set the power to 16.0 in order to make it visible enough but not overwhelm the diffuse light.

You can see the implementation of the diffuse light and specular light below:

```
void main()
{
    vec3 normal = normalize(vNormal); // Normalize the normal
    vec3 lightPos = vec3(0.0, 0.0, 5.0); // Position of the light source
    vec3 lightdir = normalize(lightPos - fragPos); // Normalize the light direction

    float ambient = 0.35;
    float diff = 0.0;
    float spec = 0.0;
    float phongExp = 8.0;

    ////////////////////////////////////////////////////////////////////////////
    // PLEASE DO NOT CHANGE ANYTHING ABOVE !!!
    // Calculate the diffuse and specular lighting below.

    diff = max(dot(normal, lightdir), 0.0);

    float specularStrength = 0.8;
    vec3 viewDir = normalize(-fragPos);
    vec3 reflectDir = reflect(-lightdir, normal);
    spec = pow(max(dot(viewDir, reflectDir), 0.0), 16.0);
    spec = specularStrength * spec;

    // PLEASE DO NOT CHANGE ANYTHING BELOW !!!
    ////////////////////////////////////////////////////////////////////////////

    if (isLightSource) {
        gl_FragColor = texture2D(tex, vTexCoord) * vec4(1.0, 1.0, 1.0, 1.0);
    } else {
        gl_FragColor =  texture2D(tex, vTexCoord) * ( ambient + diff + spec ); // Set the fragment color
    }
}
```

## Task 3

Task 3 was to add Mars to the scene.

Adding Mars to the scene was fairly simple considering all the other planets are already added, and the procedure being the same for Mars.

A new MeshDrawer object was created for Mars and the mesh was set using the existing sphereBuffers. Also the image was set using the given link. -6 unit translation via x axis is applied and 0.35 scale from all dimensions. Finally a new SceneNode call with the parent set to SunNode is enough to add Mars to the Scene.

You can see the implementation of adding Mars to the scene below:

```
marsNode.trs.setRotation(0,0, zRotation * 1.5);
```

```
   marsMeshDrawer = new MeshDrawer();
```

```
marsMeshDrawer.setMesh(sphereBuffers.positionBuffer, sphereBuffers.texCoordBuffer, sphereBuffers.normalBuffer);
setTextureImg(marsMeshDrawer, "https://i.imgur.com/Mwsa16j.jpeg");
marsTrs = new TRS();
marsTrs.setTranslation(-6, 0, 0);
marsTrs.setScale(0.35, 0.35, 0.35);
marsNode = new SceneNode(marsMeshDrawer, marsTrs, sunNode);
```