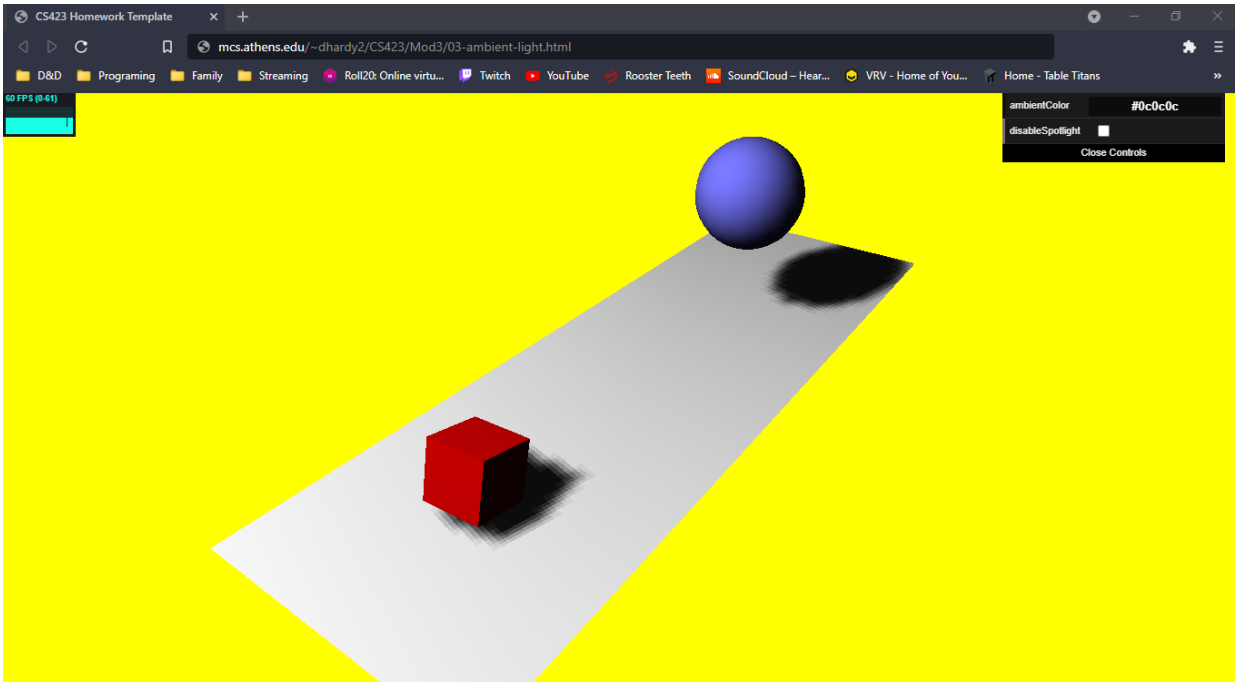
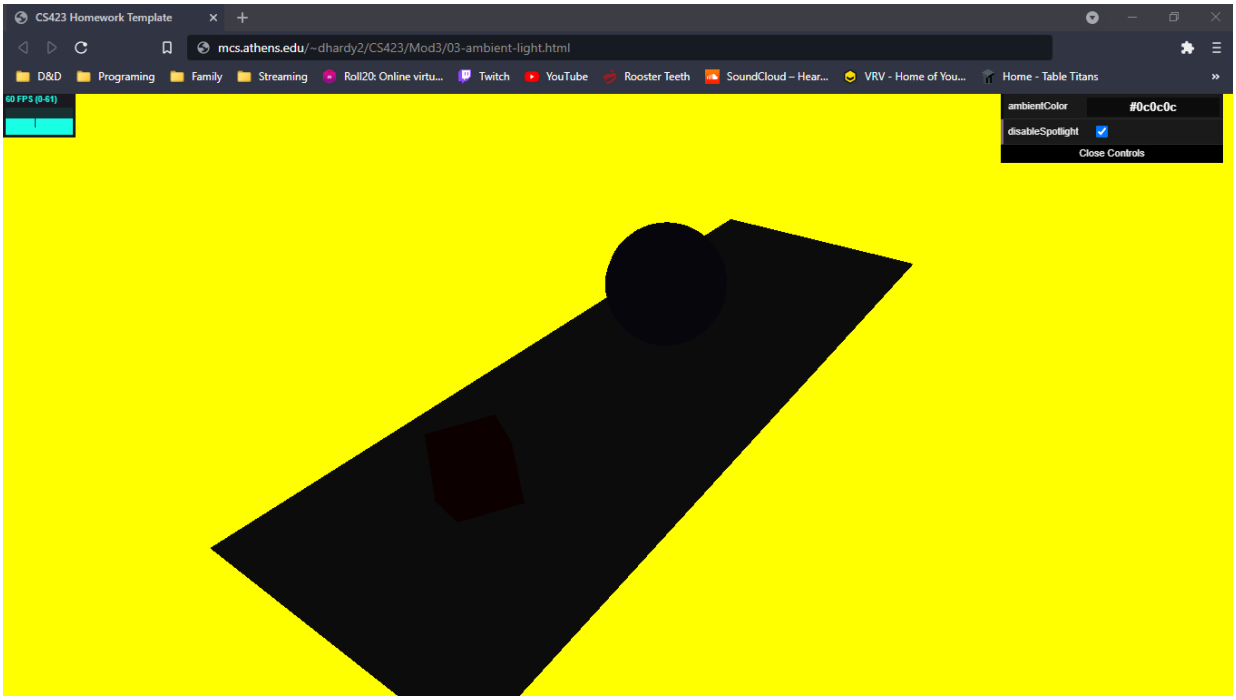
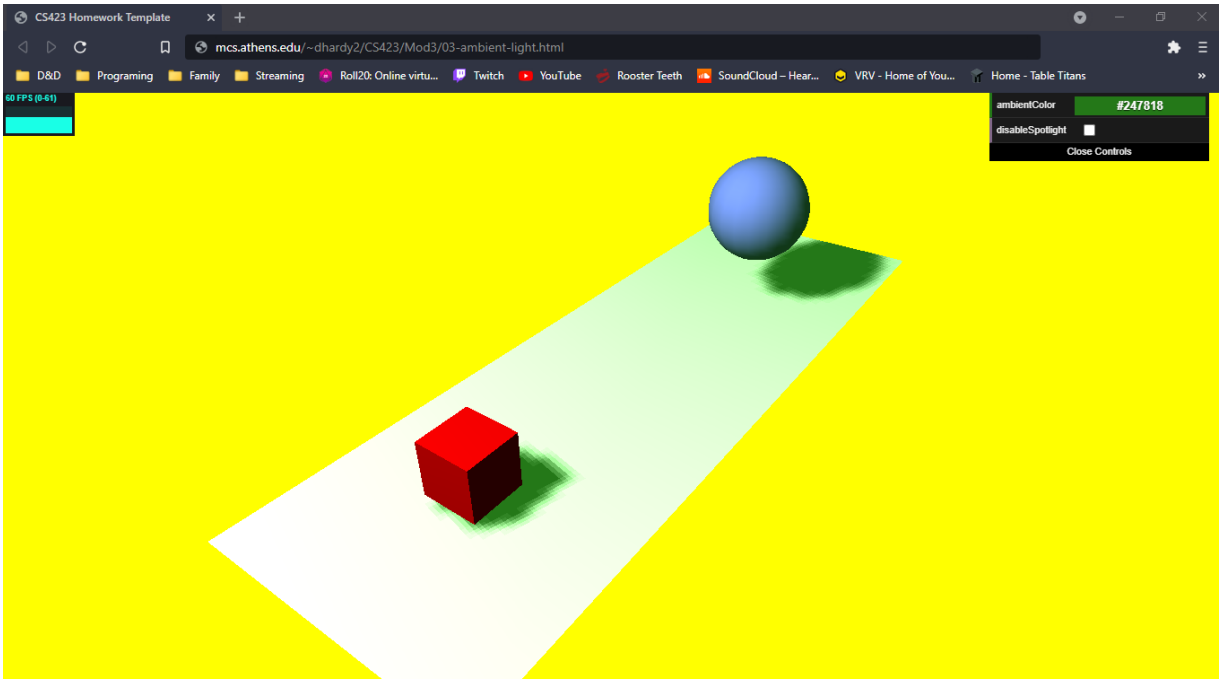
Devin Hardy

00076619

CS423







2.3 Your turn

1. The color change on the ambient lighting causes the material being used to reflect some of the color of the light.

2. When the material is basic it does not reflect light so it stays the color it has been set to.

```
<!DOCTYPE html>
<HTML>
        <HEAD>
         <TITLE>CS423 Homework Template</TITLE>
         <SCRIPT TYPE="text/javascript" SRC="../libs/three.js"></SCRIPT>
         <SCRIPT TYPE="text/javascript" SRC="../libs/stats.min.js"></SCRIPT>
         <SCRIPT TYPE="text/javascript" SRC="../libs/dat.gui.min.js"></SCRIPT>
         <STYLE>
          body {
          margin: 0;
          overflow: hidden;
          }
         </STYLE>
        </HEAD>
        <BODY>
         <DIV ID="Stats-output">
         </DIV>
         <DIV ID="WebGL-output">
         </DIV>
         <!-- Scripts that we use for running things -->
         <SCRIPT TYPE="text/javascript" SRC="03-ambient-light.js"></SCRIPT>
        </BODY>
</HTML>

function init() {

   var stats = initStats();

   // create a scene, that will hold all our elements such as objects, cameras and lights.
```

```
var scene = new THREE.Scene();

// create a camera, which defines where we're looking at.
var camera = new THREE.PerspectiveCamera(45, window.innerWidth / window.innerHeight, 0.1, 1000);

// create a render and set the size
var renderer = new THREE.WebGLRenderer();

renderer.setClearColor(new THREE.Color(0xEEEEEE, 1.0));
renderer.setSize(window.innerWidth, window.innerHeight);
renderer.shadowMapEnabled = true;

// create the ground plane
var planeGeometry = new THREE.PlaneGeometry(60, 20, 1, 1);
var planeMaterial = new THREE.MeshLambertMaterial({color: 0xffffff});
var plane = new THREE.Mesh(planeGeometry, planeMaterial);
plane.receiveShadow = true;

// rotate and position the plane
plane.rotation.x = -0.5 * Math.PI;
plane.position.x = 15;
plane.position.y = 0;
plane.position.z = 0;

// add the plane to the scene
scene.add(plane);

// create a cube
```

```
var cubeGeometry = new THREE.BoxGeometry(4, 4, 4);
var cubeMaterial = new THREE.MeshLambertMaterial({color: 0xff0000});
var cube = new THREE.Mesh(cubeGeometry, cubeMaterial);
cube.castShadow = true;

// position the cube
cube.position.x = -4;
cube.position.y = 3;
cube.position.z = 0;

// add the cube to the scene
scene.add(cube);

var sphereGeometry = new THREE.SphereGeometry(4, 20, 20);
var sphereMaterial = new THREE.MeshLambertMaterial({color: 0x7777ff});
var sphere = new THREE.Mesh(sphereGeometry, sphereMaterial);

// position the sphere
sphere.position.x = 20;
sphere.position.y = 0;
sphere.position.z = 2;
sphere.castShadow = true;

// add the sphere to the scene
scene.add(sphere);

// position and point the camera to the center of the scene
camera.position.x = -25;
camera.position.y = 30;
```

```
camera.position.z = 25;
camera.lookAt(new THREE.Vector3(10, 0, 0));

// add subtle ambient lighting
var ambiColor = "#0c0c0c";
var ambientLight = new THREE.AmbientLight(ambiColor);
scene.add(ambientLight);

// add spotlight for the shadows
var spotLight = new THREE.SpotLight(0xffffff);
spotLight.position.set(-40, 60, -10);
spotLight.castShadow = true;
scene.add(spotLight);

// add the output of the renderer to the html element
document.getElementById("WebGL-output").appendChild(renderer.domElement);

// call the render function
var step = 0;

var controls = new function () {
   this.rotationSpeed = 0.02;
   this.bouncingSpeed = 0.03;
   this.ambientColor = ambiColor;
   this.disableSpotlight = false;
};

var gui = new dat.GUI();
gui.addColor(controls, 'ambientColor').onChange(function (e) {
```

```
    ambientLight.color = new THREE.Color(e);
});
gui.add(controls, 'disableSpotlight').onChange(function (e) {
    spotLight.visible = !e;
});



render();

function render() {
    stats.update();
    // rotate the cube around its axes
    cube.rotation.x += controls.rotationSpeed;
    cube.rotation.y += controls.rotationSpeed;
    cube.rotation.z += controls.rotationSpeed;

    // bounce the sphere up and down
    step += controls.bouncingSpeed;
    sphere.position.x = 20 + ( 10 * (Math.cos(step)));
    sphere.position.y = 2 + ( 10 * Math.abs(Math.sin(step)));

    // render using requestAnimationFrame
    requestAnimationFrame(render);
    renderer.render(scene, camera);
}

function initStats() {

    var stats = new Stats();
```

```
        stats.setMode(0); // 0: fps, 1: ms

        // Align top-left
        stats.domElement.style.position = 'absolute';
        stats.domElement.style.left = '0px';
        stats.domElement.style.top = '0px';

        document.getElementById("Stats-output").appendChild(stats.domElement);

        return stats;
    }
}

window.onload = init
```