Computing Science CMPT 361 **Spring 2019**
Instructor: KangKang Yin **Simon Fraser University**

Assignment #2 (20 marks)

Programming part due: Wednesday, March 20, at 11:59 p.m.

Plagiarism is a serious academic offense: passing off someone else's work or ideas as your own in order to get a higher mark. Plagiarism is treated very seriously. The assignment you hand in must represent your own work. Submitting source code downloaded/copied from the WWW or your classmates' solutions as your own is deemed cheating, and an F grade will be awarded. However, reading and studying code from the web and discussing ideas with your classmates is allowed, but you must acknowledge their help, and still do your own work. In the README file you submit with your source code, you must list all the open source code that you have studied, and all the people's names with whom you have discussed your assignment.

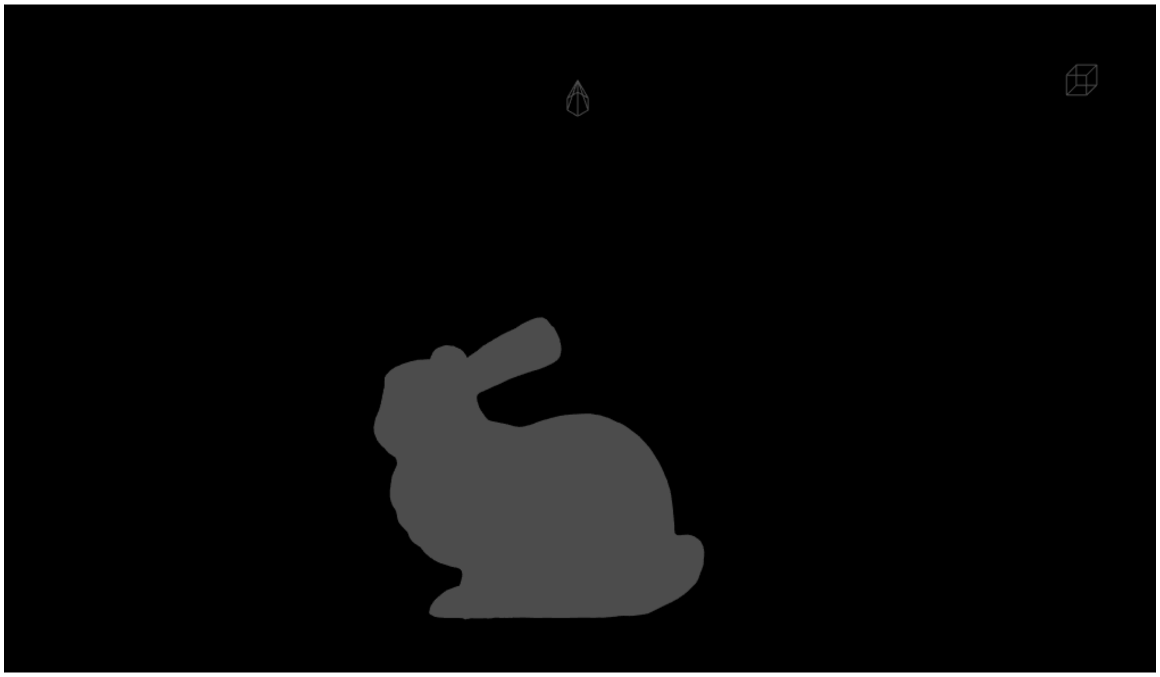
Late Policy: Late submissions are possible, but they will be penalized. One day late: 10% penalty; Two days late: 20% penalty; Three days late: 40% penalty; Four days late: 60% penalty; Five days late: 80% penalty; Six or more days late: 100% penalty.

Problem 1 (20 marks): Shading Stanford Bunny

In this assignment you are going to render the Stanford bunny with two light sources. Do not use any other third party libraries other than bunny.js and the Common folder.

(a) [3 marks] Drawing the bunny using perspective projection

There are two functions in the bunny.js provided with this assignment. get_vertices returns an array of vertex positions; get_faces returns an array of vertex indices starting from 1. The perspective camera is at (0, 0, 10), and the initial position of the bunny is at (0, 0, 0). You can assign an arbitrary uniform color for each vertex for now.



(b) [2 marks] Interactively translating the bunny

Implement translation of the bunny along the global X, Y and Z axes. The horizontal and vertical movement of the mouse when the left mouse button is pressed should correspond to X and Y translation. The Z translation can be implemented by either the up and down arrows on your keyboard, or the wheel on your mouse if you have one.

(c) [4 marks] Interactively rotating the bunny

Implement rotation of the bunny around the global X and Y axes. The horizontal and vertical movement of the mouse when the right mouse button is pressed should correspond to Y and X rotations respectively. Also implement a reset function so that whenever the user press key “r”, the bunny returns to its initial location and orientation.

(d) [2 marks] Draw an auto rotating point light as a wireframe cube

The point light is at (5,5,0) initially. The point light auto rotates counter-clockwise above the bunny in a constant speed. The rotation of the light can be turned on and off by pressing key “p”.

(e) [3 marks] draw an auto panning spot light as a wireframe cone

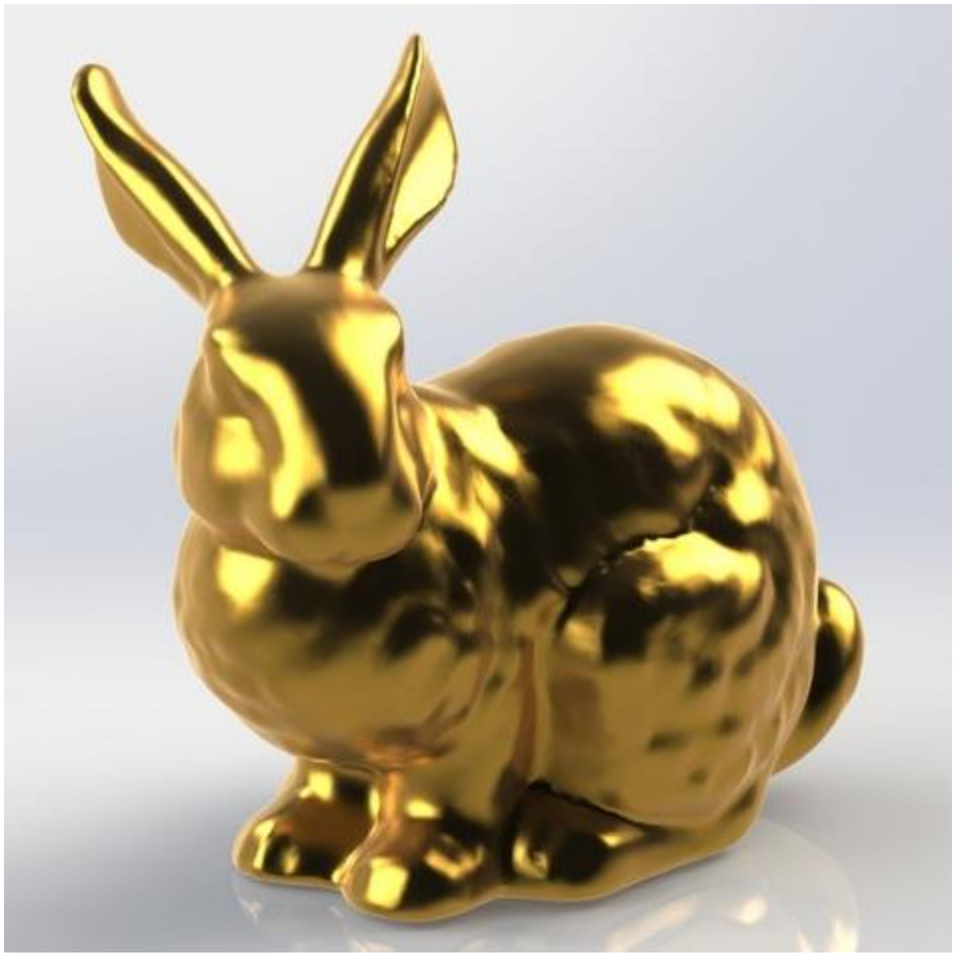
The spot light is located at (0, 4, 2) and aims at (0,0,0) initially. The spotlight auto pans side to side (i.e., from left to right and then from right to left continuously). The panning of the light can be turned on and off by pressing key “s”. The spot light has a cutoff angle of 30 degrees and an angular attenuation coefficient *e*=1.

(f) [3 marks] Phong reflection and shading

Implement the Phong reflection model and Phone Shading so that the bunny looks realistic and 3D. You need to compute vectors such as normals yourself.

(g) [3 marks] Tuning the light and material parameters

Experiment with different material and light parameters so that your bunny looks similar to the bunny below. Save one screenshot of your whole scene in which your bunny looks the closest to the bunny below. Don't worry about soft shadows and reflections. Just worry about how the bunny looks like.



Note that the above steps build on top of each other, in order. You need not submit individual programs to correspond to these steps. If you can implement all the required parts, a single, complete program is sufficient. **No skeleton code** is provided.

Submission: Please submit a zip file with student number and your name (i.e., `300000001_TerryFox.zip`). The zip file contains an HTML, a JavaScript, the best rendered image so far in jpeg format .jpg, and a README file. The README should acknowledge any help you have received and any discussion you have participated, and document any steps not completed, additional features, and any extra instructions for the TA to mark your assignment.