CPSC 314
Assignment 3

Due 4PM, Nov 25, 2005 (You can submit the programming part till Monday Nov 28 with no penalty or grace day use)

Answer the questions in the spaces provided on the question sheets. If you run out of space for an answer, use separate pages and staple them to your assignment.
Note: Some of the questions in the assignment are based on material to be covered between now and Nov 25. Answer those after the topics are covered in class.

Name: _____

Student Number: _____

Question 1	/ 10
Question 2	/ 5
Question 3	/ 5
Question 4	/ 90
TOTAL	/ 100

1. Light and shading

   (a) Given a scene with two non specular objects, one yellow ($k_a = k_d = (1, 1, 0)$) and one red ($k_a = k_d = (1, 0, 0)$), classify the following statement as true or false. Explain.

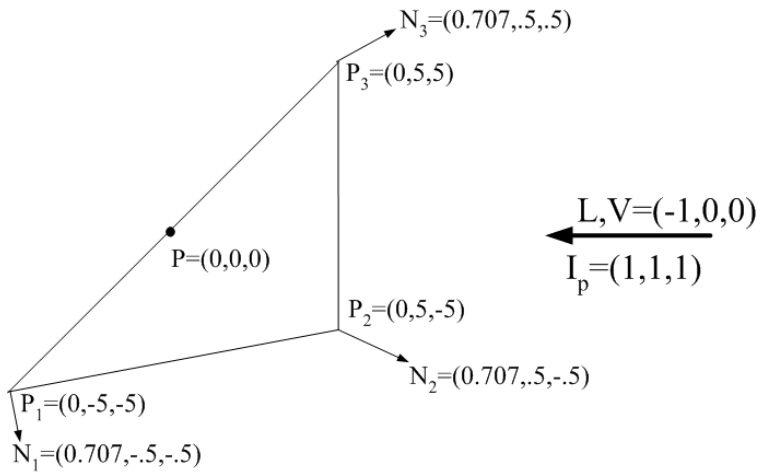
      i. (1 point) Given a single point light source with intensity $I_p = (1, 0, 0)$ the objects will have the same shading.

      ii. (1 point) Given a single ambient light source with intensity $I_a = (1, 0, 0)$ the objects will have the same shading.

   (b) (1 point) Write the openGL code for defining the following lighting scenario with three light sources: ambient light source with intensity $I_a = (0.3, 0, 0)$; directional light with direction $(1, 0, 0)$ and intensity $(0.6, 0.6, 0.6)$; point light at $(10, 0, 0)$.

   (c) (1 point) In openGL define the material properties for a triangle with $k_a = (1, .5, .5)$, $k_d = (1, .5, .5)$, $k_s = (.5, .5, .5)$ and specularity coefficient $n = 16$.

(d) In the scene below there is one directional light source at infinity $(\infty, 0, 0))$with direction $(-1, 0, 0)$. The view direction is the same as light direction $(-1, 0, 0)$. The shading coefficients for the triangle are $k_a = k_d = (1, 0, 0)$, $k_s = (0, 1, 0)$ and the specularity coefficient is $n = \infty$.



Compute the color at point P on the triangle using the following shading algorithms (use per-face or per-vertex normals as necessary):

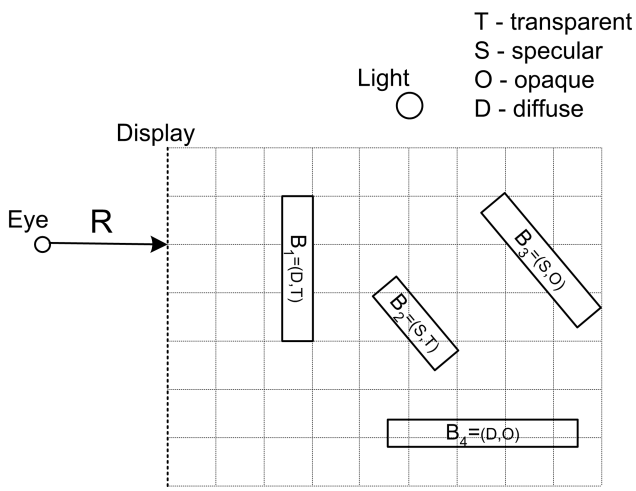
i. (2 points) Flat shading,

ii. (2 points) Gourard shading,

iii. (2 points) Phong shading.

2. Ray-Tracing

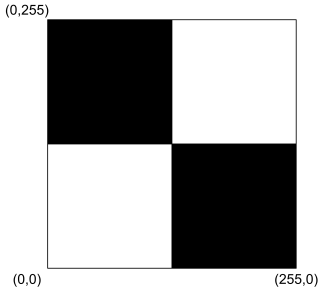
   (a) (3 points) Draw the ray tree for the ray R shown below. Assume index of refraction c_1 for air is 1 and index of refraction for all the transparent objects in the scene is $c_2 = \frac{1}{\sqrt{2}}$. Use Snell's law to obtain refraction angles.



   (b) (2 points) Assume the transparency coefficient α for the transparent objects is .5, the light intensity is $I_p = (1, 1, 1)$ (no other lights), and the diffuse/specular coefficients for the objects are $k_d^1 = (1, 0, 0), k_s^1 = (0, 0, 0), k_d^2 = (0, 0, 0), k_s^2 = (1, 1, 1), k_d^3 = (0, 0, 0), k_s^3 = (1, 1, 1), k_d^4 = (0, 1, 0), k_s^4 = (0, 0, 0)$. What is the color returned by the ray tracing algorithm for ray R?

3. Texture Mapping.

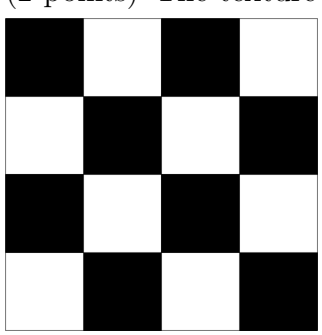
   (a) (3 points) The following texture is stored in the array *image* of size $imgx \times imgy$ (256×256).



   Draw the textured triangle produced by the following code:

```
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_S, GL_REPEAT);
glTexParameteri(GL_TEXTURE_2D, GL_TEXTURE_WRAP_T, GL_REPEAT);
glTexImage2D(GL_TEXTURE_2D,0,GL_RGBA, imgx, imgy, 0,
             GL_RGBA, GL_UNSIGNED_BYTE, image);
glEnable(GL_TEXTURE_2D);
glBegin(GL_POLYGON);
glTexCoord2f( 0, 0);
glVertex3d( 0 , 0, 0 );
glTexCoord2f( 1, 1);
glVertex3d( 1, 0, 0 );
glTexCoord2f( 0, 1);
glVertex3d( 1, 1, 0 );
glEnd();
```

   (b) (2 points) The texture below is stored in a 4×4 "texel" array.



   How will this texture look when mapped to a square of 3×3 pixels? Draw and explain.

4. (90 points) Graphics Extravaganza or Jurassic Park

In this assignment, you will be using all the OpenGL/Graphics knowledge you acquired in the course so far, to generate a fun interactive environment based on your dinosaurs from assignment 1. Your environment should contain two or more dinosaurs that interact with one another. The user should be able to control the motion of one of the dinosaurs and the rest should react to this motion. The user-guided motion should include the ability to walk the dinosaur around the scene in any direction the user wants, speed up or slow this motion, as well as the ability to perform different other motions. The other dinosaurs should react to your dino based on type of motion and proximity. Think of a logical game-like scenario for this interaction. You should also populate your scene with surrounding objects (use Google to find suitable obj format models) such as trees, houses and so on.

It is recommended that you work in groups of two. Hand in one project. In addition to assignment 1 you can use any parts of the OpenGL demo programs provided during this course as a starting point. Document what demo parts you used in your README file. If you need help regarding how to implement any particular features, do not hesitate to ask the instructor or the TAs. Be sure to develop your project in testable stages. The best projects will be glorified forever in the 314 Hall of Fame!

Grading To have your programming assignment graded out of 90, you need to implement four standard features and one advanced feature (in this case the total grade of assignment 3 will be marked out of 110 - not counting bonuses). Alternatively, you can implement all five standard features and no advanced. In this case your programming assignment will be graded out of 80 (in this case the total grade of assignment 3 will be marked out of 100 - not counting the bonus). It is recommended therefore that you implement four of the standard features first and only then turn to the advanced ones.

Standard Features Whichever of the projects below you choose to implement, it must include at least four of the five standard OpenGL features below.

- Camera motion around the scene supporting rotation, translation and zoom.
- Multiple colored light sources and multiple materials (diffuse and specular).
- Transparent and semi-transparent objects (alpha-buffer).
- Texture mapping for at least some of the objects in the scene. Either create a simple texture or load one (use parts of assignment 2 template for this). Note: OpenGL only supports textures of size $2^k \times 2^k$.
- Interaction using mouse operations. Glut provides a set of commands to support this.

Think of how best to incorporate those into your project.

Advanced Features

You should also implement one of the advanced features described below as part of your project (if you want it to be graded out of 90 - see above).

- Shadows - implement those with simplified lighting conditions (one light source). Hint: use multiple copies of Z-buffer to generate the scene with/without lighting. The openGL commands used to access Z-buffer include glDrawPixels, glCopyPixels, glReadPixels, glDrawBuffer, glReadBuffer. Read the relevant entries in openGL manual to understand how to use them.
- Smooth motion paths using splines.
- Collision detection/avoidance - for a tutorial read

  `http://www.ugrad.cs.ubc.ca/~cs314/Vjan2005/schedule.html#week10.`
- Picking - your code would need to recognize which object the mouse location corresponds to (when mouse or keyboard callback is evoked). For a tutorial see http://www.lighthouse3d.com/opengl/picking/ and http://www.ugrad.cs.ubc.ca/ cs314/Vsep2003/slides/week4.wed.col.pdf

Think of how best to incorporate those into your project.

Document the standard and advanced features you have implemented.

BONUS The best three projects will get extra bonus points (10 for first, 8 for second, and 5 for third) and they will be entered into the 314 hall of fame.

Hand-in Instructions

You do not have to hand in ANY printed code. Create a README file that includes your name, your login ID, and any information you would like to pass to the marker. When working in pairs, only one of the partners needs to submit the code. Make sure however that **both** names are listed in the README file.

Create a folder called 'assn3' under your cs314 directory and put all the source files, your makefile, and your README file there. Also include any images that are used as texture maps. Do not use further sub-directories.

The assignment should be handed in with the exact command:

```
handin cs314 assn3
```

Programming Assignment Grading

Similarly to assignment 1, the programming part of this assignment will be graded with face-to-face demos in front of the TAs/Instructor.

Document all the extra features you implemented in the README. If need be, you can also concisely summarize the arguments in your README about incomplete work.

- We will hang a signup sheet for demo slots in the lab (CS 011). Each slot will be 10 minutes.
- You must ensure that your program compiles and runs on the lab machines. If you worked on this assignment elsewhere, it is your responsibility to test it in the lab. The face to face grading time slots are short, you will not have time to do any 'quick fixes'! If your code as handed in does not run during the grading session, you will fail the assignment.

- The code that you demo must match exactly what you submitted electronically: you will show the TA a long listing of the files that you're using, so that he can quickly verify that the file timestamps are before the submission deadline.
- Have a printout of your README file for the TA to read.
- Arrive at CICSR 011 at least 10 minutes before your scheduled session. Log into a machine, and double-check that your code compiles and runs properly. Then delete the executable.
- When the TA comes to your computer you will type the following:

```
ls -l
make
```

- You will then run your assignment. Since you have exactly 10 minutes, think in advance how you want to demo all the cool features of your project. If you fail to mention a feature, it might be overlooked.