INFO6020 – Graphics 2 - Mid-term Exam – Winter 2019

Instructor: Michael Feeney

The exam format:

- You may use any resources you feel are necessary to complete the exam, but you are to answer the questions **on your own**. I will be looking for plagiarism (i.e. copying) very carefully. There is *no possible way* that the specific code to answer these questions, or the output to the screen, would be very similar to the look of another student's code. Remember, this is a test and there are very clear policies about cheating on tests.

    - http://www.fanshawec.ca/admissions/registrars-office/policies/cheating-policy
    - http://www.fanshawec.ca/sites/default/files/assets/Ombuds/cheating_flowchart.pdf

- The questions are ***NOT*** of equal weight. There are five (5) pages with five (5) questions

- The answers may be one or a combination of the following:

    - Short answer (in your own words)
    - Snippets of code
    - Complete running solutions

- CLEARLY indicate which answer goes to which question. My suggestion is that you place each answer in its own folder, named "Question_01", "Question_02" and so on (or something equally clear). Another option is to create a Visual Studio solution and add a number of projects – one per question – to it. If I can't make heads or tails of what question is what, I probably won't even mark it.

- Place any written answers into a Word, RTF, or text file. Again, *clearly* indicate which question you are answering.

- If you are combining answers (which is likely), please indicate this with a "readme" file or some note (*not* buried in the source code somewhere).

- For applications: if it doesn't build and run, *it's like you didn't answer it*. I'll correct trivial, obvious problems (like you clearly missed a semicolon, etc.), but you need to be sure that it compiles and/or runs.

- You have until **11:59 PM** on **Monday, February 25th** to submit all your files to the appropriate drop box on Fanshawe Online.

  NOTE: Although this may "look and feel" like a project, it isn't, it's an **exam**, so there is **no concept of "late marks**"; if you don't submit your files by 11:59 PM, you don't get any marks at all. *Don't Be Late submitting.*

  (Also be **SURE** that you are actually submitting the correct files)

- You can reach me through e-mail (mfeeney@fanshawec.ca) or by calling the school.

- There is also a **PlyFiles.7z** file you will need. It's available on FOL with the mid-term.

Questions:

You are to create a variation of Pac Man, but with the Daleks from Dr. Who.

Using the keyboard, you are able to navigate a Daleks through a large maze, but by viewing what they see from a centralized TV screen.

There is a maze generator code that will generate a large, random maze, that will be drawn on screen, and visible all the time.

Floating above that maze is a giant TV (yes, it's sort of surreal and strange, but think about if you went to a sporting event, like a basketball or hockey game, where there's basically a giant TV placed above the court/ice). This TV will show the viewpoint of the Dalek inside the maze.

The TV model is separated into the "TV body" and the "TV screen" portion.

To simplify the movement, the Dalek can only go left, right, up, and down. So you don't have to worry about the Dalek (and camera) turning.

You also don't have to worry about going through the walls.

1. (20 marks) Using the cMazeMaker, generate a large maze (at least 50x50 blocks).

   I WOULD SUGGEST A MUCH SMALLER MAZE FOR YOUR INITIAL TESTING, THOUGH!

   The code is taken from this site: https://codereview.stackexchange.com/questions/135443/c-maze-generator, and you can use it as is, or you can use the cMazeGenerator, or you can generate the maze as a text file and load it – I really don't care, but there MUST be a way for me to use a different maze (like one I generate or load).

   Note that the code generates a maze that is placed into a vector of vectors of vectors...

```
std::vector< std::vector< std::vector< bool > > > maze;
```

   The 1st and 2nd vectors basically make a 2D array, so maze[x][y] gives you each "cell" of the maze.

   The "cells" can be filled (walls) or unfilled (hallways/corridors), which is indicated with the 0th element of the 3rd vector. As you can see from the original code,

   `if (maze[a][b][0]==true)`, then it's a wall, otherwise it's a hallway/corridor.

   Draw this maze on the screen, using cubes for the walls. You can literally go through the "maze" structure, and drawing a cube where there's a wall.

   Place the maze on screen so that the camera is just above the maze, looking slightly down. You should be able to see all (or essentially all) of the maze. Something like this:

Set up the lighting so that you can see all of the maze, clearly.

I want it evenly lit, by as many lights as you'd like. The cubes should be texture mapped with something like stones, rocks, etc. I'm not concerned about how nice the texturing looks, just like the maze looks like it's made of rocks.

Place a "ground" under the maze, too. This can be a single large cube, a flat quad, or something else.

2. (20 marks) Place a Dalek inside the maze and show that it can move around (left, right, up, and down) with keyboard control. Use the WSAD or arrow keys to do this.

   Note that there are several resolutions of the Dalek model, so use the one that will give you OK performance.

   The Dalek should be about the same height as the blocks, and should be clearly visible, even though it's relatively small (in relation to the entire scene).

   Make the Dalek some very bright and/or vivid colour to help it stand out.

3. (10 marks) Place the giant TV model above the maze. You should be able to see both the maze and the TV. It might be that part of the maze (the part in the distance) might be blocked by some of the TV, but the idea is that you can see the maze, the Dalek moving in the maze, and the TV at the same time.

4. (30 marks) Update the render so that it takes 2 passes.
   - The first is where the camera is placed where the Dalek is, rendering to an off-screen texture.
   - The second pass is the maze, the Dalek, and the TV body, and the TV screen. Note that the camera here is *not* where the Dalek is, but where it was up to question 3. The TV screen model should have the off-screen texture (from the perspective of the Dalek) on it.

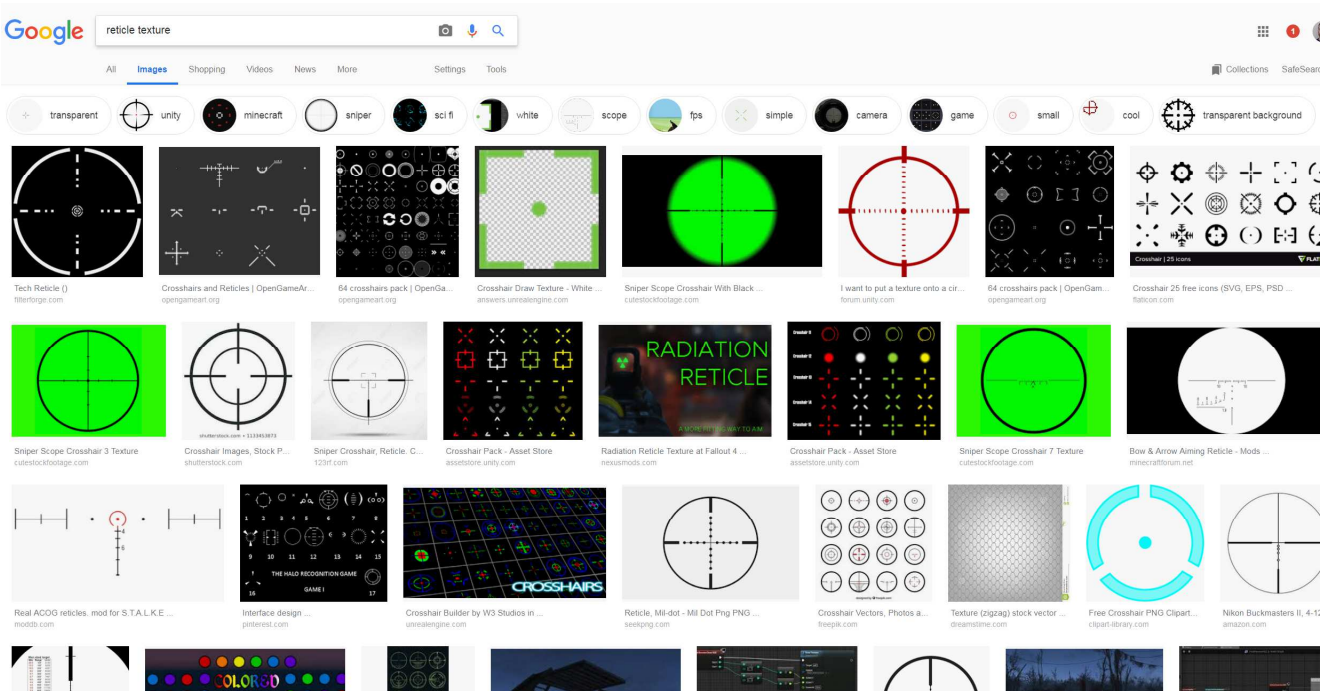
   So when you move the Dalek, now, we should see the Dalek moving *and* the TV screen showing what the Dalek would see.

   Note: In the 1st pass, I would suggest *not* drawing the Dalek at all. This will prevent issues where the Dalek model is blocking the camera. You can draw the Dalek, if you'd like, but note that you'll have to place the camera slightly *ahead* of where the Dalek actually is. This seems a little more complicated.

   Note: you can't have the Dalek model block this 1st pass camera, or you will lose marks.

5. (20 marks) Superimpose a “reticle” texture onto the screen above the maze. This is to get the impression that you are looking through the camera built into the Dalek.

   Choose any one you’d like. If you google “reticle texture”, you’ll get a number of possible ones to choose from:



(That’s it for the exam).