# **INFO6045 – Animation - 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 six (6) 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 **Tuesday, February 26th** 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](mailto:mfeeney@fanshawec.ca)) or by calling the school.

## Questions:

For this exam, you will use the “RPG Character Mecanim Animation Pack FREE” model from the Unity Asset store: <https://assetstore.unity.com/packages/3d/animations/rpg-character-mecanim-animation-pack-free-65284> (this is the one that we’ve had in the example code in class)



You are to specifically use the following animations:

* RPG-Character\_Unarmed-Attack-L3.FBX
* RPG-Character\_Unarmed-Attack-R3.FBX
* RPG-Character\_Unarmed-GetHit-B1.FBX
* RPG-Character\_Unarmed-GetHit-F1.FBX
* RPG-Character\_Unarmed-GetHit-F2.FBX
* RPG-Character\_Unarmed-GetHit-L1.FBX
* RPG-Character\_Unarmed-GetHit-R1.FBX
* RPG-Character\_Unarmed-Stunned.FBX
* RPG-Character\_Unarmed-Fall.FBX
* RPG-Character\_Unarmed-Death1.FBX

As well as the idle animation, and possibly your choice of other animations.

Continuing the theme of Daleks in mazes, you are to create an application that allows you to sent an animated robot into the maze and attack the Dalek that’s pestering Dr. Who!

1. \*\*\*Note: This is the identical question from the “Graphics 2” mid-term from yesterday, which is why it isn’t worth a whole lot of marks\*\*\*  
     
   (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. (10 marks) Randomly place a Dalek into the maze. It should be in a corridor, not inside a wall. Use whatever resolution model you’d like. It should fit inside the corridor, so be about the height of the cube model.
3. (10 marks) Load the “idle” animation RPG-Character model. Scale the model so that it also fits inside the corridor. Place it at some other point in the maze, fairly far away from the Dalek model.
4. (20 marks) When the character is not moving, run the “idle” animation over and over.
5. (40 marks) Using the WSAD or arrow keys, move the RPG-Character model through the maze. While it is moving, it should be playing the “walk” or “walk-slowly” animation. You need to move the model enough to minimize “foot sliding” (i.e. the feet shouldn’t slide along the ground while the animation is running).   
     
   You don’t need to gradually “turn” your model when it reaches the corners of the maze; you can simply face the model in the direction it’s going in.   
     
   The walk animation should only play while the WSAD or arrow keys are being pressed. If no keys are pressed, then the model should stop moving forward and start playing the “idle” animation.   
     
   Position the camera so that it generally follows the RPG-Character, near it, and floating above the maze. You can use any kind of “follow camera” you’d like. I want to see the character and around 3-4 blocks of the maze nearby. So not the entire maze, but just the region near the RPG-Character. You don’t have to position the camera so that’s it’s “facing in front” of the character – just that it’s showing the character and the region of the maze that’s nearby.
6. (50 marks) When the RPG-Character is “close enough” to the Dalek model, you can attack.  
     
   Using the 1,2,3, and 4 keys, play one of the following attack animations:
   1. 1 causes the "Attack-L3" animation to run
   2. 2 causes the "Attack-R3" animation to run
   3. 3 causes the "Attack-L3" animation to run
   4. 4 causes the "Attack-R3" animation to run

Using the “hand” bone (like the fist, not the fingers) of the skinned mesh, determine if the character is, in fact, close enough to “hit” the Dalek.   
  
To determine this, imagine that the Dalek is actually a cylinder, and pick a radius that’s approximately the size of the model you are drawing on screen. If the hand bone reaches inside this radius, then the Dalek is hit, otherwise it’s not.

To show a hit, do the following:

* Stop the animation at the moment of the “hit”. You can do this by passing the same frame time over and other.
* Change the Dalek’s colour to something ridiculous, like bright red, or bright pink.
* Hold the animation frame of the RPG-Character, and the colour of the Dalek for around 2.0 seconds
* After that, change the colour of the Dalek back to “regular” (whatever it was that you picked), and let rest of the attack animation play out

When the attack animation has finished, continue to play the “idle” animation.

That’s it.