# **INFO6045 – Animation - Mid-term Exam – Winter term 2020**

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 seven (7) 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, March 3rd** 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)



These animations are available on FOL and many of them are in the github repository.

You are to recreate the classic game “Impossible Mission”, from way back when Mr. Feeney was just a kid! You know, back when dinosaurs roamed the land, and the Earth was still being formed from cosmic dust!

Here’s a video of the basic gameplay: <https://www.youtube.com/watch?v=ivHFP3dJAkM>

It’s essentially a “low budget” version of the classic NES Super Mario, but with worse graphics, and easier enemies.

Here’s the key “gameplay”:

* The player (the human) can move and jump.   
  But in our, 2020 version, the player can also attack! By punching!  
    
  You will use the RPG fbx model for both the character and the animations.
* The “enemy” robots *can’t* jump, but can just move back and forth. You will use the Android Mascot robot for these characters. Note that there are several models, an FBX one, and several PLY exports (one with the entire model, and one with the parts separated). None of these has any “animations”, though.
* The “levels” are just flat platforms, which you will make with cubes. You can get a general sense of this from the video – it’s not anything super complicated, just a few “floors” that might have enough room to jump between.

1. (30 marks) Recreate this level from “Impossible Mission”, using the cube 3D model to make up the floor parts, the RPC character for the player, and the Android Mascot Robot for the enemy robots. You don’t have to include the other parts of the level (the desk, computer, vending machines??, etc.). The models are completely still (static) at this point.



1. (80 marks) Enhance question 1 so that the player character can move and animate. It should specifically be able to:   
   * Move left and right, using the S & A or left & right arrow keys on the keyboard.
   * If the shift key is up, then the “walking” animation is used.
   * If the shift key is down, then the “running” animation is used, and the character moves faster.
   * The model should more relative the distance in the animation, to avoid as much “foot slip” as possible.
   * When the character is at the end of a platform, then should “fall” down to the next level, and display the “falling” animation.
   * If no keys are pressed, the character should enter the “idle” animation, facing in whatever direction it was last placed (left or right).
2. (40 marks) Add the ability to jump up to the next platform.   
     
   When pressing the up arrow, the W key, or the space bar, the character should run the “jump” animation, and the character should “jump”. This jump should be high enough to reach the next platform. Allow the character to simply “jump” to the next platform, if it jumps up; in other words, the “ceiling” should not stop it from jumping up from a lower platform – in the image, if you press “jump”, the character should jump, run the “jump animation”, and end up on the platform immediately above.
3. (40 marks) Add a “roll” animation, using another key, like the “S” or down arrow. This takes place if the character has *just* been walking or running, then the button is pressed. This should run the appropriate “roll” animation.   
     
   If all the keys are released when the “roll” animation is completed, the character should enter the “idle” animation.
4. (20 marks) Move the “enemy” Android robots around. These don’t have animations, but should move left and right, on the platform, turning around when they get to the end, then moving the other way.
5. (40 marks) Add basic collision and attack:  
   * When a robot hits the player, the player “dies”. If this happens, stop the movement of the robots, and have the player execute the “dying” (or equivalent) animation.
   * If the “enter” key is pressed, the player executes an “attack” animation in the direction it is facing. If they player is “close enough” to a robot, then the robot should flash red (normally the android robot is green). If the robot is “attacked” 3 times, it should “explode”. This can be done in one of two ways:  
     1. The scale of the entire robot model should increase over a second (to about 1.2x size), then decrease over a couple seconds, until it disappears (scale = 0.0f)
     2. (Bonus 20 marks): The robot “flies apart”, over a few seconds, using the individual Android Robot model parts.

Note: While a robot is dying/exploding, all the animations in the game have stopped (the character animation doesn’t progress, and the other robots don’t move).

1. BONUS (30 marks) When all the robots are destroyed, and the character leaves the level, another, different level is loaded, and the gameplay continues.

That’s it.