# INFO6019 – Physics 1

# Final Exam – Wednesday, December 12th, 2018

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
- It is an "open book" exam. You have access to anything you book or internet resource you'd like
- The questions are ***NOT*** of equal weight. The exam has **three (3)** questions and **eight (8)** pages.
- Your solution can be either graphical or console based (or graphical + console based if that's helpful).
- **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.
- Do ***NOT*** do some clever "*oh, you just have to comment/uncomment this block of code*" nonsense. However, if the questions ***CLEARLY AND OBVIOUSLY*** build on each other, you may combine them (like if one question places objects, then the next one moves objects around with the keys) – even so, **MAKE IT 100% CLEAR** to me what questions the solution is attempting to answer.
- Place any written ("essay" or short answer) 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 **Wednesday, December 12th** 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 the time the drop box closes, you don't get any marks at all.

  *Please don't be late submitting.*

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

- Your solution may **not** contain any third party “core C++” libraries (like boost) or C++11 "`auto`" feature. If it has either of these things, the question(s) will not be marked (because it won't build). You many have other “utility” libraries, like ones to load textures, models, sounds, etc.

- When ready to submit, please delete all the “extra” Visual Studio files before zipping it up (remember this is C++, so all I really need is the .h and .cpp files, right?), like the “Debug” and “Release” folders with the “obj” files, as well as the intellisense file (in VS2017, that’s the “.vs” folder).

- **If the solution does not build (and run), I will not mark it** (so you will receive zero on questions that can't be built and/or won't run). When I say "run", I'm not speaking about some, random, unforeseen bug, but rather something that you should have obviously dealt with, like memory exceptions, etc.

- Unless otherwise indicated, all these solutions assume that you are creating/using a C++ project using Visual Studio 2008 through 2017 using the OpenGL 4.x API (with glfw, glad, and glm).

# “It’s a ***sky*** pirate life (and death) for me!” *(again!)*

You are to complete another part of the “Sky Pirate” came, namely the flight planning portion.

In this exam, you will need a number of models:

- Islands (9 of them) – these are in the “Islands (seed 200 to 208).7z” file
- Spheres
- Wood House model (optional)
- Sky Pirate ship model (optional)

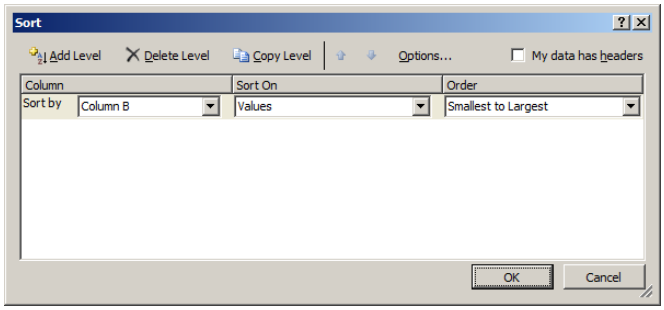
Your solution needs to be a 3D OpenGL solution. Note that I am NOT concerned with execution speed, so don’t worry about that.

1. (10 marks) While you don't have to use the "island generator" to make the nine (9) islands, you will have to create your own unique "Archipelago" of islands.

   The islands are to be arranged in a grid of 3x3 meshes, viewed like this from above (i.e. each cell is one of the mesh island models, arranged in a 3x3 grid, where "x, y" is the centre mesh (likely at the origin):

| | | |
|---|---|---|
| x-1, y-1 | x, y-1 | x+1, y-1 |
| x-1, y | x, y | x+1, y |
| x-1, y+1 | x, y+1 | x+1, y+1 |

   - Use the Pick My Islands helper.xlsx Excel file to assist in the orientation and placement of these islands:
     - The file lists mesh file names on the left ("Seed_of_XXX_n_uv (island).ply")
     - The column to the right has a random number from 0 to 1.0 (orange coloured)
     - The next column (to the right) has labels that match the grid above
     - The next column (to the right) has another set of random values, but these are angles
     - Every time you open or edit the file, these random numbers change
     - To use the file:
       - Select the column called "Random #", copy, then immediately paste over this column, using "Paste Special", and "Values". So you are *overwriting* the "=RAND()" formula in the cells with the values they last had.
       - Do the same for the "Rotation:" column, overwriting the formula with values.
       - Now the two columns won't change
       - Select *both* column A and B (names of the mesh *and* the "Random #" column), and choose "Data", then "Sort", then:
         - Uncheck "My data has headers" (if it's checked)
         - Select "Column B" in the "Sort By" column →
         - This will sort the mesh names in some random order.




       - Open each mesh model and use MeshLab to rotate the mesh around the Y axis by whatever value is in the "Rotation:" column.
     - Match the file name (1st column) with the location in the grid (3rd column) to get your specific layout.
     - ***NOTE: You need to submit your final, edited Excel file, with your submission***

This is an example of what you might end up with:

| | Random # | | Rotation: |
|---|---|---|---|
| Seed_of_204_xyz_n_uv (island).ply | 0.0771241 | x-1, y-1 | 90 |
| Seed_of_203_xyz_n_uv (island).ply | 0.3023859 | x-1, y | 270 |
| Seed_of_201_xyz_n_uv (island).ply | 0.3797692 | x-1, y+1 | 270 |
| Seed_of_208_xyz_n_uv (island).ply | 0.6552644 | x, y-1 | 0 |
| Seed_of_200_xyz_n_uv (island).ply | 0.790561 | x, y | 90 |
| Seed_of_207_xyz_n_uv (island).ply | 0.8967802 | x, y+1 | 90 |
| Seed_of_202_xyz_n_uv (island).ply | 0.9195602 | x+1, y-1 | 90 |
| Seed_of_206_xyz_n_uv (island).ply | 0.9200246 | x+1, y | 90 |
| Seed_of_205_xyz_n_uv (island).ply | 0.9524014 | x+1, y+1 | 180 |

The island layout:

| | | |
|---|---|---|
| x-1, y-1 | x, y-1 | x+1, y-1 |
| x-1, y | x, y | x+1, y |
| x-1, y+1 | x, y+1 | x+1, y+1 |

This would mean that your archipelago would be laid out like this (as viewed from above):

| | | |
|---|---|---|
| Seed_of_204_xyz_n_uv (island).ply (rotated 90 degrees) | Seed_of_208_xyz_n_uv (island).ply (not rotated, i.e. rotated 0 degrees) | Seed_of_202_xyz_n_uv (island).ply (rotated 90 degrees) |
| Seed_of_203_xyz_n_uv (island).ply (rotated 270 degrees) | Seed_of_200_xyz_n_uv (island).ply (rotated 90 degrees) | Seed_of_206_xyz_n_uv (island).ply (rotated 90 degrees) |
| Seed_of_201_xyz_n_uv (island).ply (rotated 270 degrees) | Seed_of_207_xyz_n_uv (island).ply (rotated 90 degrees) | Seed_of_205_xyz_n_uv (island).ply (rotated 180 degrees) |

If you'd like, you can combine these 9 models into one, large mesh, if that will assist you.

Or you can load them all as 9 separate models.

The Sky Pirate ships can navigate from island to island, in the air, obviously, but the like to stay at a relatively consistent height for the entire trip.

Choose a height for your pirate ship to fly. It should be about ¼ the height of the centre island mountain (approximately). I'm not looking for exact values here, just that the ship "looks like" it's flying at ¼ of the height.

And this will also depend on the scale of your particular solution.

Your solution is to "plot" the possible path of the pirate ship, by placing a series of sphere meshes, at regular intervals.

You will control two points, the starting and ending point of the journey, using the keyboard. These can either be two different coloured spheres (perhaps a "green" sphere for the starting point, and a "red" sphere for the ending point, or the actual "ship" and "house" models, the ship for the start – where the ship is, now – and the house for the eventual destination).

There are two general things that have to be taken into account:

- If the pirate ship can fly in a straight line, then it will do that.
- If there's an island "in the way", then they might choose one of two options:
    i. Go up and over the island (not ideal, since they'd have to climb/fall, and if they don't they might smash into the island), or
    ii. Go "around" the island, even if that takes longer. But, they are pirates, so they aren't likely going to do that... instead, they are just going to go "up and over" whatever is in their way! ☺

2. (50 marks) Add the ability to move the two “start” & “end” spheres (or the ship & house model)

   Note: The start and end position can be anywhere on the map, even over the water. If you think it’s silly to place the house model over the water, then just pretend it’s a “house boat” or something.

   - Use either the WASD or “arrow” keys to place the starting position mesh (sphere/ship)
   - Use the same keys, but with the “shift” held down, for the ending position (sphere/house)
   - Assuming you are using the Y as the “height”, and the island models are axis aligned to the X & Z axes, then one set of controls move the sphere along the X axis, and the other along the Z. This should be a fairly small movement per keystroke; so I should be able to position the mesh pretty much wherever I’d like, in the archipelago, to a fair precision (use your discretion about this).

   **Note: These next two item represent the bulk of the marks for this question:**

   - These should be *just* above the surface they are over. This means that if you move the mesh over an island, it should stay at the same height *above the surface of the island mesh.*

     *(You would accomplish this by testing the height of the mesh from the triangles immediately beneath it, keeping the mesh at a constant height above).*

   - The movement of the markers should be immediate; I shouldn’t have to wait several seconds until the meshes move, upon pressing the keys. In fact, I should be able to press and hold the keys and see the markers move along interactively.

     *(in other words, you should be using some kind of broad-phase system to limit the number of tests you are making between the meshes and the underlying terrain)*

   - The movement of the sphere/ship/house does *not* have to be “interpolated”. In other words, if the height of the mesh changes (as it goes over an island), then the sphere/ship/house can immediately move to the appropriate position+height; it does *not* have to “smoothly” move from one location to another.

3. (50 marks) Building on question 2, add the entire, straight line, path from the starting to the ending points. So as the start and end points are chosen, a series of regularly spaced spheres (of another colour, something easy to see, like bright green – use your discretion on this) is placed between the start and end points.

   - The number of spheres should be sufficient to show that this is path. I'm guessing 50-100 spheres, if we are showing the path from one end of the archipelago to the end. But this is, again, up to you. It should be obvious that this is a path, though (not just a few spheres).

   - You place the spheres in the same way that you placed the sphere/ship/house, but this time at the "flying height" of the pirate ship (1/4 the height of the tallest mountain in the centre of the archipelago). So the spheres should be *this height above the terrain/water at all times.*

   - The spheres should be drawn quickly, so that moving the starting/ending locations can happen *interactively* (just like in question 2). Ideally, there should be no performance difference between question 2 and question 3, just that there's now a "path of spheres" between the start and end points.

   - The path would go up and over any mountains, in a straight line, from start to end.

   - (Bonus: 10 marks): The number of spheres is relative to the overall length of the path. What I'm looking for is that the it's a "dotted line" sort of situation, where the *number* of spheres change as the path gets longer/shorter, but the spacing between the spheres stays the same (instead of a fixed number of spheres along the entire path, and these spheres get closer/farther apart, depending on the overall distance), so something like this:

     i. Short path: 

     ii. Longer path: