# **INFO-3111 “C++ Graphics” Final Exam Make-up, 2023 (Friday, June 27th, 2023)**

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 **seven** **(7)** questions and **ten (10)** pages. The questions involve submitting a working Visual Studio solution.

|  |
| --- |
| * The questions build on each other, to make a complete scene. However, you may decide that a different camera angle will better show the scene, so:   + You are to submit a single solution (not multiple solutions/projects)   + Use keyboard controls to best show the appropriate question, so pressing “1” will place the camera to best show question 1, “2” to show question 2, etc.   + NOTE: you do not *need* to do this, but it’s very likely that it’s to your advantage.   **Do not** comment out your code, or expect me to alter it in any way at all. It should be ready to run as is. |

* 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.
* When ready to submit, PLEASE for the love of all that is good on this Earth, delete all the “extra” Visual Studio files before zipping it up (part of knowing what you’re doing is knowing – or figuring out – what files are needed and what aren’t.), like the “Debug” and “Release” folders with the “obj” and log files, as well as the hidden “.vs” (intellisense) folder – this will save a tremendous amount of space and shorten your upload time..  
  + **But**, give me the **ENTIRE SOLUTION**, not just the source files.
* **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.  
    
  If you’re missing some common stuff that git doesn’t like, like the lib files, it’s no big deal (I can unzip those), but if you’ve adjusted to the project/solution – like placing the includes/libs in a different folder – *and* it doesn’t build because you’ve forgotten to submit these, that’s pretty annoying…
* **No “auto” or the boost library** (or any other 3rd party library we haven’t used).   
  If you use either, I will not mark your submission and you will receive a mark of zero.  
  No exceptions.   
  This is a trivial restriction – it takes almost no effort to *not* use these items, and there is no justification for “needing” to use either in INFO-3111 (or any other C++ development for that matter, but I digress).

**PLEASE make sure you give me your entire solution, so that I can simply download, un-compress, build and run it.**

Also PLEASE remove the things you don’t need, for example:

* I don’t need *the entire github repository* including the lectures, etc.
* I don’t need the “.vs” folder (which can be gigabytes in size)
* I don’t need *every single model you’ve ever seen*, or downloaded, or whatever.  
  Do I need the “Galactica” model? The “fractal terrain”? The “hockey player”?   
  Seriously? Delete these massive things, please.

## The Models and textures:

## The vast majority of the models are the ones in the “Dungeon\_models” folder that’s been on github for a few weeks and/or the “PLY (POLYGON – Dungeons Map - Selected models).7z” from the bonus (CPXX) checkpoint. In other words, if you’ve pulled from git in the last week or so (if you didn’t, how did you get this file??)

## The textures are under the “textures” folder inside the “Dungeon\_models” folder. Almost all of them use the “Dungeons\_Texture\_01.png” file (Synth Studios has variations of this to give different colours, etc.)

## There is an additional “Dungeon\_models (additional)” compressed file that has some additional models you’ll need. It’s only the models that are missing from git, so if you unzip this folder and overwrite the “Dungeon\_models”, it will only add the missing model files.

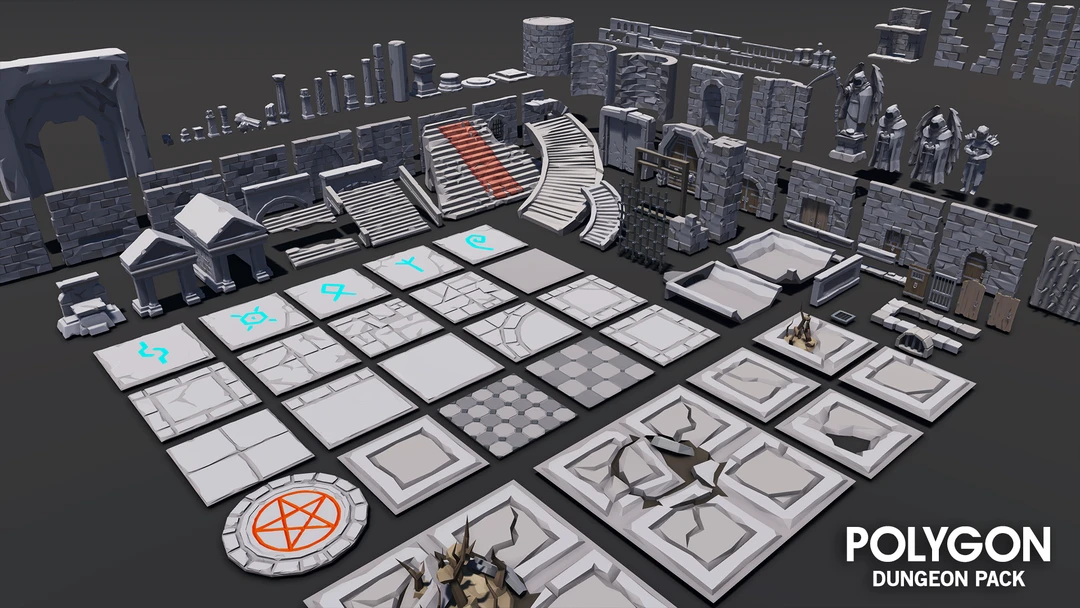
## The 7z and zip files have the same content.

## The Questions:

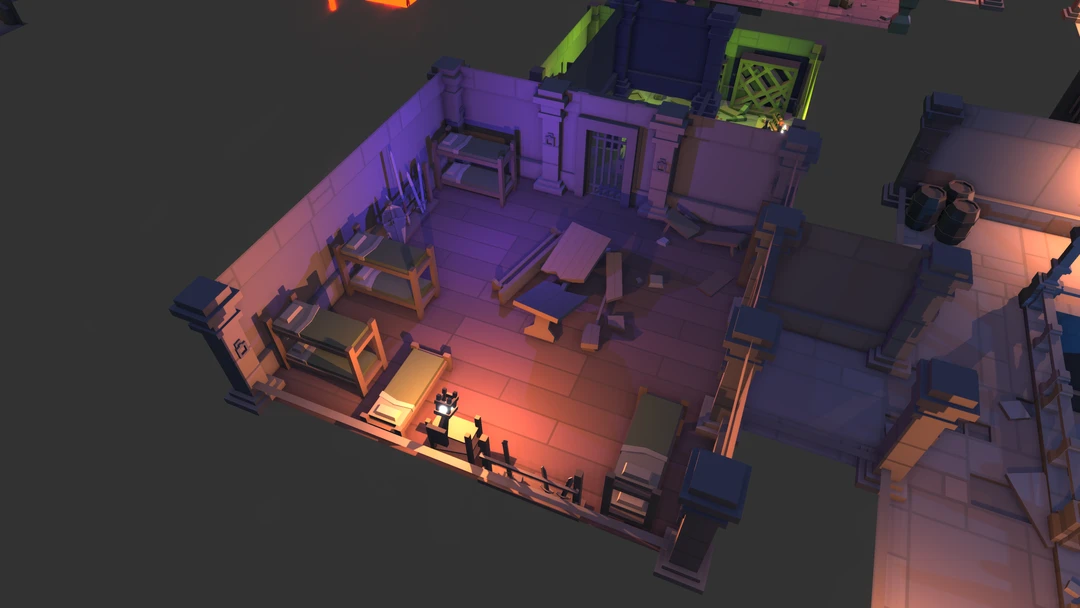
You are to create a small dungeon made up a several large open spaces and smaller corridors.

You will use the assets taken from the Synty Studios “POLYGON - Dungeon Realms” (<https://syntystore.com/products/polygon-dungeon-pack>).

Many of the “environment” assets can be combined like LEGO bricks into any shape you’d like. They are all proportional sizes. In the picture below, you can see there’s a bunch of different “floor” models as well as many “wall” models.



When combined, you can make rooms like the one below. Also note that the small corridor is also made of floor and wall models:



To get the layout of your maze, you will need to run:

***NEW!!!***

* the **cMazeMaker\_INFO3111.exe** file  
    
  and
* the **WhatModelsShouldIUseINFO3111SummerFinal2023.exe** programs.

|  |
| --- |
| **Note that you need to submit the output of BOTH programs (the .txt and .log files)**  **If you do NOT include these files, your exam will NOT be marked (and you will get a mark of zero).** If you screw up the files, then regenerate them – the values should be the same and is based on your student number (the *file name* won’t be, though – that’s based on the time). |

**Some notes about the models:**

* They are aligned in one corner of the model. This allows you to “snap” them together more easily (MeshLab “Render”, then “Show Axis” will show this.)
* The textures that correspond to the model are shown in the ply file, for example, the SM\_Env\_Floor\_Grate\_01.ply file has the following:  
    
   comment TextureFile Dungeons\_2\_Texture\_01\_A.png   
    
  This means that you can use the Dungeons\_2\_Texture\_01\_A.png file as the appropriate texture.

**You are to make the MAZE from the cMazeMaker\_INFO3111.exe file, and IGNORE the “dungeons” from the WhatModelsShouldIUseINFO3111SummerFinal2023.exe file.**

**You *will* need the list of items that is at the bottom of the WhatModelsShouldIUseINFO3111SummerFinal2023.exe file.**

**As an example, assume that your log output on the right 🡪**

Your SN is: 1234567

Your maze is:

XXX.XXXXXXXXXXXXX

X...X.....X.....X

X.XXX.XXXXX.XXX.X

X.X...X.......X.X

X.X.XXX.XXXXXXX.X

X.X...X...X...X.X

X.X.X.X.XXX.X.X.X

X.X.X.X.X...X.X.X

X.X.X.X.X.XXX.X.X

X.X.X...X...X...X

X.XXXXXXXXX.XXX.X

X...X.......X.X.X

XXX.X.XXXXXXX.X.X

X.X.X.....X...X.X

X.X.XXXXX.X.X.X.X

..........X.X...X

XXXXXXXXXXXXXXXXX

Note: you will use *your* ***own***maze **\*NOT\*** this one.

1. (50 marks) Create a floor and outside wall.   
   * The “X” are walls or pillars and the “.” are open areas   
     (i.e. they are just floor)
   * Note that the wall and floor models are not the same thickness. In other words, the “X” characters around the outside are much thinner.
   * Note that there’s one or two entrances (a break in the “X”s around the outside). In the example, you can see this at the bottom and top.
   * Add enough evenly lit lighting to see everything.   
     This should be “bright enough” but not “too bright” – it’s an underground dudgeon, right?
   * Apply the appropriate textures to match the models.
   * Do ***NOT*** but a “roof” on the maze.
   * Lower the floor tiles along a path of at least fifteen (15) locations with at least two (2) bends. This area should be *noticeably* sunken (lower) than the rest of the floor.   
     This will be a small wandering river type water feature, and will be “filled with water” later in the exam. If you feel that you need “sides” to enhance how sunken it is, then use more “wall” tiles.

|  |
| --- |
| Note that the “walls” can also be made like the pillars, like four (4) walls wherever there’s an “X” will give you walls. i.e. you don’t *have* to make the walls thinner than the pillars.  However, if you want something fancier, see the “bonus” questions later in the exam. |

**Note: You will *not* be able to manually place these in the time you have for the exam.**

What you need to do is either read the output of the file or hard code this text file into a 2D array (or vector of vectors) or whatever, then read through that array, placing the floor tiles.

Remember that a string is a 1D array, so something like this is a 2D array where you can access like any other 2D array:

std::string myMaze[20] = {

"XXX.XXXXXXXXXXXXX",

"X...X.....X.....X",

"X.XXX.XXXXX.XXX.X",

"X.X...X.......X.X",

"X.X.XXX.XXXXXXX.X",

"X.X...X...X...X.X",

"X.X.X.X.XXX.X.X.X",

"X.X.X.X.X...X.X.X",

"X.X.X.X.X.XXX.X.X",

"X.X.X...X...X...X",

"X.XXXXXXXXX.XXX.X",

"X...X.......X.X.X",

"XXX.X.XXXXXXX.X.X",

"X.X.X.....X...X.X",

"X.X.XXXXX.X.X.X.X",

"..........X.X...X",

"XXXXXXXXXXXXXXXXX"

};

for ( std::string row: myMaze)

{

std::cout << row << std::endl;

}

for (unsigned int row = 0; row != 19; row++)

{

for (unsigned int col = 0; col != 19; col++)

{

char cell = myMaze[row][col]; // cell is the character

}

}

Each “X” is a pillar (4 models + floor tile) and each “.” is just a floor tile (random from 6 of them).

You can do the same sort of thing with the outside walls.

The inner walls can be done in the same way (“X” is a pillar).

1. (100 marks) Fill in the rest of the walls inside the maze.

* The “X” inside can represent walls or pillars/columns.
* If there’s a few “X”s in a row, then it’s a wall.
* If there’s an “X” by itself (i.e. not touching another “X”), then it’s a pillar or column.
  + Note that if an “X” touches another “X” *at a diagonal* then it’s *not* connected.  
    i.e. it’s not part of the wall or it’s an adjacent pillar.   
    There aren’t any “diagonal” walls.
* Walls can be placed right on the floor (or slightly penetrating if that looks better).
* Pillars/Columns are made of four (4) wall models in a square, the size of a floor model, making a square shape, with the “backs” of the wall models facing towards the inside.
* Apply the appropriate textures to match the models.

1. (0 marks, yes that’s “zero”): Solve the maze.

* Just like when you were a little kid, solve the very complicated maze.
* You will need this solution for later questions.

1. (150 marks) Place several lit “torches” in the scene ALONG THE SOLUTION PATH:

* Choose one type of “torch” (light) model in the “Torches” folder.
* Place ten (10) of these torch models **spread out evenly** along the solution path, attached to the walls, near the top. The idea is that they are torches with fire or candles, attached near the top of the wall.
* Place small, bright red-orange lights where the fire/candle of the torches would be.   
  These should light up the area nearby (like a small candle or flame would).
* Make the torches “flicker” by slightly perturbing the linear attenuation each.  
     
  You do this by first setting up the “regular” (or non-flickering) attenuation you’d like.   
    
  Then, for each frame, adding a small random number to this “base” attenuation of the light and passing *that* (i.e. total) to the shader. Don’t update the original “regular” value, though. The idea is that you’re *passing* this slightly adjusted value each frame, not saving it.   
    
  Make sure this small random number is negative as well, so when you are “adding” this small value, the torch can get dimmer as well (adding this small negative number makes it slightly dimmer, while adding the small positive number makes it slightly brigher).

1. (50 marks): Place the items from the WhatModelsShouldIUseINFO3111SummerFinal2023 in the maze ALONG THE SOLUTION PATH.   
   * If there are duplicates, place two (or more) of them.
2. (50 marks): Choose a small utensil item of your choice (ie. A glass or goblet or vial or bottle or something like that). You can ***NOT*** use any “crystal” models.   
   * Place at least five (5) of these objects *somewhat* close together along the “solution path” of the maze. These should look like they have randomly been scattered or dropped, like they were in a bag that had a hole in it, and as a traveller walked through this maze, they fell out one by one.
   * Make them semi-transparent (so you can see through them).
   * The transparency should be “order independent”, in that I can move the camera around and see all the crystals through all the other crystals.
   * Make them very “shiny” (i.e. have a very high specular component)
   * **BONUS (100 marks):** Make them reflect the skybox as well.
3. (150 marks): Place some skeletons in the corridors OFF the solution path.   
   * Place three (3) skeleton or “dead body” models in various paths off the solution path.
   * These are travellers that got lost in the maze.
   * Place spot lights above these poor souls, lit from above with a tight angle and a tight penumbra. The light should be dim and yellow in colour.
   * Place some plants around these dead bodies.

**BONUSES:**

1. (100 mark **bonus**): Mimic the time of day changing.   
   * Gradually move the spot lights from one side of the mode to another.
   * When they get to the one side, gradually dim them (mimicking evening).
   * After this, make the overall lighting dimmer but *don’t* reduce the torch lights.   
     This will enhance the light from the torches.
   * After a little while, place the spot lights on the *other* side other model, gradually increase the overall lighting, and start moving the spot lights again.
2. (150 mark **bonus**): Make the wall thin, not using a “bunch of columns”.   
   * Only use a four (4) sided column for “X” values that are separated from everything.
   * Every other “X” – that’s a wall – use an appropriate two sided wall model (i.e. the wall models facing back to back). Make sure that the corners line up OK.
   * If there’s a “T” or “X” intersection, you can intersect the models if you’d like.
   * In other words, the walls in the map should only be “2 wall tiles”
3. (100 mark **bonus**): Change from light to day over time.   
   * Pick two skybox textures, one for “night” and one for “day”.
   * Gradually, over 20 seconds or so, blend that skybox from day to night and back again.
   * At the same time, gradually increase the overall lighting of the scene during the day, then gradually dim it until only the torches are visible.
   * This means the “spot lights” on the dead bodies and plants would have to get brighter and dimmer, too (they are shafts of light from sunlight).

**That’s it.**