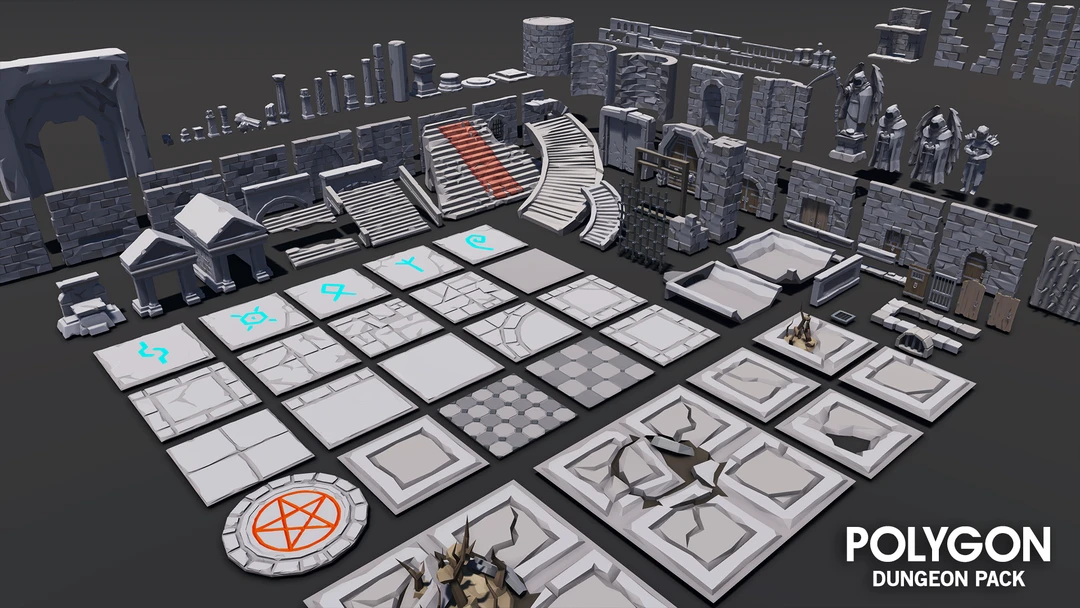
|  |  |
| --- | --- |
| Program: | CPA3 |
| Course: | INFO3111 – C++ Graphics – Summer 2025 |
| Professor: | Michael Feeney |
| Project # 2: | Texturing and lighting enhancements of scene |
| Weight: | 15% of your final mark |
| Due Date: | Friday, August 22nd, 2025 @11:59 PM (“mid-night”)  (Note: This is due *after* the final exam on Monday, August 18th)) |

Description and Purpose

You are to **enhance your project #1 submission** with more advanced lighting, texturing, and transparency.

Note: You will need some additional models to the scene that aren’t in the “dungeon realms”.



**You are going to make a “dungeon” or “labyrinth” using modular models from the Synty Studios “POLYGON - Dungeon Realms” (**[**https://syntystore.com/products/polygon-dungeon-pack**](https://syntystore.com/products/polygon-dungeon-pack)**).**

The Models and textures:

These models have always had textures assigned to them, so now you are to use those textures.

You will also need to find some additional textures for more generic things (like water, etc.).

Dungeon layout, “decorations”, and your original submission:

Since you are enhancing your project #1 submission.   
See the Project #1 document for details about the map and how to interpret it (walls, “X”s, etc.)

* You are using the same log file you generated when you ran the **INFO3111Project1\_S2025\_DungeonMaker.exe** program.   
  You can generate a different file if you’d like.
* Either way: **You need to submit your log file**.
* How “good” does your dungeon have to be (in term of “Project #1”)?  
    
  **It needs to “good enough” to “enhance”.** So it needs correct walls, floors, lights, objects, etc.   
    
  Below is a random sampling of “good enough” submissions; **if your dungeon isn’t comparable to something like this *to start with* then please “fix” it.**

|  |  |
| --- | --- |
| F:\z_FS\3111_3D\2024\Local_SAM\Project_2\Good Project #1 output #3.jpg | F:\z_FS\3111_3D\2024\Local_SAM\Project_2\vlcsnap-2024-06-17-17h35m11s871.jpg |
| F:\z_FS\3111_3D\2024\Local_SAM\Project_2\vlcsnap-2024-06-17-17h34m16s933.jpg | F:\z_FS\3111_3D\2024\Local_SAM\Project_2\vlcsnap-2024-06-17-17h35m49s841.jpg |

Details:

1. (Zero (0) marks**\*\***) Make the dungeon from project #1 including all these key items:  
   * Correct walls: aligned correctly (facing the correct way), touching the ground, etc.
   * Torches on the walls (of correct scale – they are about the size you could carry them).
   * Trees and clusters of plants.
   * Crystals scaled correctly (they are large, not larger than the hallways or too large to walk around).
   * Items correctly scaled (bottles *aren’t* the size of a person, for example. Mugs aren’t either.).

**\*\***While this isn’t worth any marks, if you haven’t met these requirements, the rest of the marks in the project will be scaled accordingly, limiting your maximum mark.

**To help gauge this, here’s how I’m going to mark this:**

* There are five (5) items in the above list. So each is “worth 20%” (so to speak)
* Let’s say your walls are a mess and you have no plants. You’d lose the “marks” for the walls and plants part, so you’d “get 0%” for those two, leaving the remaining three (3) (“60%” left).
* Now let’s say that you’ve got a mug that barely fit in the halls and a chair that’s upside down and floating in the air. The rest of your “items” are OK, though. You’d get ½ for that.
* Your crystals and torches are good, though (as per project #1)
* That would mean you’d start the project limited to:   
     
  0% (walls) + 20% (torches) + 0% (trees) + 20% (crystals) + 10% (items) = 50%

This would mean that you’ve already limited the maximum marks you could get by 50%

I’m not going to debate the precision of this – it’s a “ballpark number”.   
(<https://www.merriam-webster.com/dictionary/ballpark>)

The idea is that if you did OK (75% or higher) on project #1, then you likely *won’t* be penalized.

But if your project #1 is a mess to start with, you can’t “enhance” it much, right?  
  
***Now is the time to “fix” your original submission, if you need to.***

|  |
| --- |
| **Note: I’m only concerned about the meshes and placement of meshes, as the colours, etc., are going to be replaced with textures in this project.** |

1. (150 marks) Update the “torches” in the scene:

Part of the issue with Project #1 torches was that there really wasn’t a “fire” (light source) that you could see. You could see the *effect* of the lights on the walls, etc., but you couldn’t see the actual “fire”.

Update your eight (8) torches in the following manner:

* Using the flame\_xyz\_rgba\_n\_uv.ply (in the flame.7z file) model as the “fire”.
* Place this model in the appropriate location in your chosen torch model.  
    
  NOTE: it may or may not be the correct scale or orientation – be sure to handle this!
* Choose some kind of “fire” texture to texture map the fire model.  
    
  A good example is here: <https://www.freepik.com/premium-photo/fire-texture-background_8304643.htm>   
  **BUT you may NOT use this particular texture.**
* Place a light slightly above the “fire” mesh. You can use a point light or a spot light – your choice (whatever you think looks better)
* The colour should be yellow-orange – like a candle or fireplace flame – NOT bright white.
* Like project #1, make the torches “flicker” by slightly perturbing the linear attenuation each:
  + Choose attenuation values for the “dimmest” the fire will be.
  + Choose another set of attenuation values for the “brightest” the fire will be.
  + It should illuminate the walls, like a torch. (So it doesn’t make the entire hallway super bright)
  + **EACH FRAME:**
    - calculate a random value *between* the “brightest” and “dimmest” attenuation values.
    - Add the “dimmest” attenuation to these random number(s). Pass those attenuation values.
    - In other words the random numbers are picked between the “brightest” and “dimmest” attenuation values, and the change *every frame*, causing a “flickering”
* This is the same effect as project #1, so if you already did that, it’s the same
* If you *didn’t* do it this way (like it’s a constant brightness, or changing slowly over time, or something else), then update it so it “flickers”.
* Make the flame mesh **semi-transparent**.

1. (100 marks): Update the look of your crystals.   
   * Make the crystals semi-transparent.
   * Near one of your crystals, place four (4) more crystals scattered nearby.
   * Ideally, this should be in a larger room, but just make sure they aren’t touching each other.
   * Ensure the crystals have “order independent” transparency   
     (this is the bulk of the mark for this question).
   * If they don’t already have a very high specular (shininess) value, make them *very* shinny.
2. (150 marks): Update the lights over the plants.   
     
   Replace the point lights with **spotlights**, shining down (like a shaft of light):   
   * The angle should be slightly “off” of straight down.   
     (the only time it would shine directly/straight down is if the sun was directly overhead)
   * Place a soft, but visible penumbra around the spot light.
   * [](https://www.wanderinraw.com/post/skylight-cave)Make sure it’s bright enough to be *clearly* seen *as a spotlight.*
   * It should be bright yellow/white like mid-day sunlight.

<https://www.wanderinraw.com/post/skylight-cave>

1. (100 marks) Add a pool of water:

XXXXXXXXXXX.XXXXXXX

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

XXX.X.X....XX.XXXXX

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

X.XX.......XXXX.X.X

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

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

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

X.XX....X.........X

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

X...XXX...XX....X.X

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

XX..X...........XXX

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

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

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

X.X.X......XXXXXX.X

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

XXXXXXX.XXXXXXXXXXX

* Choose an area of the map that’s at least 2x2 “X” spots.   
  (i.e. a room that’s larger than the hallways – which are one “X” wide)
* The square on the map to the right is *more* than large enough   
  (It’s 9x4 “X” squares in size)
* This is where the “pool” of water will be.
* Lower the floor in this area one “wall” deep. Add walls around the edges of the pool - I should *not* see “gaps” between the floor tiles, vertically. Like if you move them “too low” you’ll see gaps.   
    
  (It’s a pretty deep pool)
* Using MeshLab, generate a “Fractal Terrain”:
  + Choose “Filters”, “Create New Mesh Layer”, “Fractal Terrain”
  + The “Ridged multifractal terrain” makes nice water.
  + Use the last three (3) numbers of your student number for the “seed” value (the default is “2”)
* Generate the texture coordinates by:
  + Choosing “Filters”, “Texture”, “Parameterization: Flat Plane”
  + If you didn’t rotate it, it’s aligned to the XY axes, so choose those as the “Projection Plane”
  + Choose “Filters”, “Texture”, “Convert PerWedge UV into PerVertex UV” to get the UV coordinates applied to the Vertices (not the “wedges/triangles/faces”)
* This “Fractal Terrain” will be the surface of the water. Place it at the top of the pool.   
    
  Note that you may have to scale it, rotate it, or whatever, to get it to “fit” into the top of the pool.
* If the surface of your water is particularly “wavy”, make sure the edges are *under* the edges of the pool. I don’t want to see any water “hovering” over the rest of the floor.
* Choose a “water” texture from the internet. Place this on the water (“fractal terrain”) mesh.
* Make the water semi transparent *even if I move the camera under the water.*   
    
  In other words, I should be able to be above the water and look down into the pool, seeing the bottom ***AND***I should be able to move the camera into the pool, looking up out of it, and see things in the dungeon as well (like if I fell into the pool and was underwater).
* Place at least three (3) additional “items” of your choice into the bottom of the pool.   
  (These are the “items” like in project #1: mugs, barrels, dead bodies, skeletons, etc.)

**BONUSES:**

* (+5%) For the torches, update the flame mesh scale each frame to coincide with the “flickering”.   
  This has to be *non-*uniform scaling, only along the *longest* axis of the flame.   
  The idea is that when the fire is the “brightest” it is also the longest flame.
* (+5%) Update the crystals so that they are refractive as well as transparent.
* (+5%) Update the crystals so that they reflect the skybox.
* (+10%) Add a day/night cycle:
  + Pressing a button starts this cycle. Pressing it again stops the cycle.
  + The entire cycle should be 10+ seconds (like going from day to night, then back to day again, should take *at least* 10 seconds).
  + Day to night:
    - Gradually drop the overall ambient (or whatever general lighting you have) down to nothing, so you only see the flames flickering.
    - Slowly dim the spotlights AND adjust their angle so they move (like the sun would).
  + Night to day:
    - Gradually brighten the overall ambient (or equivalent) light.
    - Slowly brighten the spotlight AND adjust their angle.
    - Note: they would be going the other way (because the sun would be rising from the other side of the sky, right?)
* (10+) Add an option to prevent the camera from going through the “walls” so you can mimic moving first-person-style through the dungeon.   
    
  There are two options here:
  + Classic old-school dungeon crawler style where you can only turn 90 degrees at a time and move one “tile” at a time: <https://www.youtube.com/watch?v=GQgHSX2tof4>
  + (*Another* +5%) Modern, where you can go forward and turn at any angle.   
      
    To do either of these, you’ll have to update both the eye position and the target position.
* (+5%) Add a skybox. Note that if you do the day/night cycle bonus, this must change as well.

You will submit:

* **Your entire solution** (PLEASE remove the “extra” files from it, making it smaller), and compress it.
  + Submit the **entire solution(s) and project(s)** *not* just the source code files.
  + Please, for all that is good and sane in the world, delete any files you don’t need as well as any temporary files Visual Studio creates (“.vs”, the debug and release folders, etc.). You won’t lose marks for submitting these, but it’s very annoying and it’s something you should know how to do.
* **If the solution does not build (and run), I will not mark it** (so y*ou 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.
* While I *might* use the Debug build while marking, your submissions **must** build and run in **Release** build.
* Ensure that your *paths are set relative*, not to some folder that’s only on your machine.
* Unless otherwise indicated, all these solutions assume that you are creating/using a **C++ project** using **Visual Studio 2022** using the OpenGL 4.x API (with GLFW, glad, and GLM), and build in **64-bit** **Release** using the **default** setting in Visual Studio (**C++ 14**).
* You are **NOT** permitted to use any type of AI (i.e. something local as you can’t use the internet).   
  If I get a sense that you’ve done this, I’ll will orally/verbally interview face-to-face/zoom, asking you questions about your code, and if I get a sense you didn’t write it (aka don’t understand it/don’t know why you put that code there in that way) then I will assume you’ve cheated, just as if you’d gotten the code from somewhere else – aka, you cheated and will follow the A136: Academic Integrity policy: <https://www.fanshawec.ca/sites/default/files/legacy/oldfanshawe/sites/default/files/assets/policies/pdf/a136.pdf>
* Your solution may ***not*** contain any third-party libraries (like boost), **smart pointers,** or the “**auto**” keyword.
* To be clear: if you have ***any*** of these elements, you will receive a mark of zero (0) for that question.   
  NO exceptions: you should be aware of what’s currently used in industry (99% C++98/2003), and what just happens to be in the newer standards.
* **A video demonstrating your application.** This can be using OBS or zoom or FRAPs or even your cell phone camera but must *clearly* show:  
  + You launching the application from visual studio.
  + CLEARLY showing some identifying information (your name and student number) and the date.
  + What you are doing (buttons being pressed, etc.) as it is happening.

Additional requirements:

* While you may freely “borrow” mine code, your code should be “sufficiently different” from mine in terms of the output on screen. See the "plagiarism" test, later in this document, for more details.
* Further, you *cannot* simply use an existing game engine (or part of a game engine), even if it's a "from source" engine (i.e. you have the entire source) to complete this assignment; it should be either completely new of significantly modified. This includes, but is *not* limited to: Unity, Unreal, Cry, Anarchy, XNA, Cocos, Ogre, the framework from the OpenGL text, etc.
* This also includes chatGPT (or equivalent) AI code generation.  
    
  **Related to plagiarism/cheating and not doing much work:**   
  + If you simply submitted the in-class code, then you invested zero time, so you did no work, so you get a mark of zero (in that case, it is a clear case plagiarism/cheating, and an academic offence would also be submitted).
  + If you took the in-class code and made some trivial changes - like replaced the teapot model with the rabbit model, slightly repositioned them, and maybe changed their colour - you might not have technically "cheated", but you did essentially no work: "How long would it take me - your instructor - to make those changes?" If it's something that would take 10 minutes, you won't get many parks for that.
  + It must be something that a random "typical" person would say "yes, that's significantly different" in order to "be different".

Here’s the thing: I know the code *\*I\** wrote and if I’m suspicious that you had didn’t really write (in other words: understand) the code you submitted, I’ll simply ask you about it face-to-face. Like why you did this or that, how you might do it differently, ask how you might have changed stuff. If I get the sense that just “got this code somewhere” and don’t understand what it’s doing and/or why it’s there, I’ll assume you did (get it from “somewhere” – so you didn’t write it yourself - and file an academic offence.

* If you code does not even compile, I will not mark it. Since it can't run, you would get a mark of zero.
* If you code does not build (i.e. linker error) and run (i.e. no crazy run-time crash that is unexpected), I may investigate this further, but only if there is some simple problem and/or *very* slight and/or *very* obvious (and easy to fix) configuration error or last minute typo. (I might do something minor like try to unzip the libraries files that github stripped out or something along those lines, but I should be able to download it, build it, and run it, without incident.)
* It **MUST** build and run in **RELEASE** mode (64 bits).
* Warning level 3 is fine. In fact, I’ll almost certainly completely ignore any “warnings”. You can’t even build most common libraries on level 4, so don’t even bother.
* **No “boost” library or “auto” keyword. Use of these will give you a mark of zero, no exceptions.**

Project Corrections

If any corrections or changes are necessary they will be posted to the course web site and you will be notified of any changes in class. It is your responsibility to check the site periodically for changes to the project. Additional resources relating to the project may also be posted.

80/8-year old “squinty eye” plagiarism test:

I have very little tolerance for plagiarism, but some students might be unclear about what it is.

Basically, it’s submitting somebody else’s work as your own.

There is sometimes some confusion over this because you could argue nothing is actually “unique” (see: <http://everythingisaremix.info/> for a fascinating overview of this).

The whole point of assignments/tests/projects in this course (or any course, really) is to try to see if you are actually able to ***do*** the coding that’s asked of you. In other words: How competent are you? Handing me someone else’s code and/or making a trivial change isn’t good enough.

Also, it’s illegal:

* <http://www.plagiarism.org/ask-the-experts/faq/>
* <http://definitions.uslegal.com/p/plagiarism/>
* <http://en.wikipedia.org/wiki/Plagiarism>
* <https://www.legalzoom.com/articles/plagiarism-what-is-it-exactly>

In other words, I’m not going to be drawn into a giant debate over how “different” your code is from mine or anyone else’s, if any sensible person (including me) would conclude that the code/application is pretty much the same thing, then it is. It is up to my discretion to decide this.

* While you may freely “borrow” mine (or anyone other) code ***but*** your code should be “sufficiently” different from mine (you might want to replace the word “sufficiently” with “significantly”).
* In other words, you *cannot* simply use an existing game engine (or part of a game engine) to complete this assignment; it should be either completely new of **significantly** modified.
* How will I determine this?
  + If I showed your application and/or your source code to either my pragmatic 80-year-old mother, or a typical 8-year-old, or even some random person walking down the hallway (i.e. a non-expert), and they looked at it, tilted their heads, squinted their eyes, and said “you know, they look the same,” then they ***are*** the same.
  + Another test would: How much time it would take for a "competent programmer" (me, for example) to make the changes you are submitting? The point here is that I don’t “care” if you tell me “But it took me *weeks* to make the changes!” Fine, but if I can make those same changes in 10 minutes, then not a lot of work has been done (certainly **not** sufficient work for someone who is trying to convince me they are proficient).