**INFO6019 – Physics 1**

**Midterm Exam – Wednesday, October 30th, 2024**

Instructor: Michael Feeney

## The exam format:

* It is an “open book” exam. You have access to anything you book or internet resource you’d like.
* You may use any resources you feel are necessary to complete the exam, and we’ve discussed what you can and can’t do in class. You are free to use whatever resources have been provided for you (github, etc.) and any code *you* have made for yourself during this term.   
  + <http://www.fanshawec.ca/admissions/registrars-office/policies/cheating-policy>
  + <http://www.fanshawec.ca/sites/default/files/assets/Ombuds/cheating_flowchart.pdf>
* I’m mostly concerned about that *you* did this *yourself* in that you know what you’ve typed in. If I suspect this is something you’ve taken/stolen/generated from somewhere else and you have no clue what it does or how it works, I’ll ask you about it; if *you* wrote it yourself, you should know how it works and how it might be changed/updated/fixed/done-differently – if not, then how did you do it?
* The questions are ***NOT*** of equal weight. The exam has **five (5)** questions and **eight (8)** pages.
* Separate the questions or combine?   
  + I will *not* modify your code, so **don’t** add some “comment/un-comment this block of code” or “move file X to location Y…” nonsense, because:  
    1. You should be able to add something like a keystroke/menu/whatever to change how your program runs and…
    2. I’m not going to change the code, so it’s pointless.
  + If you feel that the questions “build” or “combine” together clearly, then you may submit combined answers to each question. For example: if Q1 had “draw an island”, then Q2 had “add some buildings”, and Q3 had “put some lights on the buildings”, etc. – then these reasonably “build” on one another. A different way of thinking about this: If you submitted an answer with building with lights that sat on an island, think: does this answer all three (Q1-Q3) questions? If yes, then you can combine them.   
      
    What wouldn’t combine might be a day/night scene, say Q4 (in this example) was “change the scene so it’s night-time by adjusting the lights from Q3”. If you added a keyboard shortcut that switched from day to night, then you *could* submit one program, but if you didn’t do this, then you’d have to submit two programs: one for daytime (Q1-Q3) and one for night-time (Q4).
  + If you submit one solution/project, I’m going to assume that it answers all the questions, i.e. you have combined all the questions into one answer.
  + If you submitted separate solutions/projects, then CLEARLY indicate which ones answer what questions. If you do this, I’d suggest copying the entire project (or entire solution) and CLEARLY renaming it something like “Question\_01” or “Questions\_01\_to\_04” or something like that.
  + Do whatever you think would make it the easiest for me to mark.
  + If there’s something I need to know (keyboard controls, pressing a button to start, etc.) then make a README type file AT THE ROOT location, not buried somewhere in the folder or code.
  + You many have other “utility” libraries, like ones to load textures, models, sounds, etc. However, make sure your submission is complete so that I can build your solution.
  + Please delete all the “extra” Visual Studio files before zipping it up (like the “Debug” and “Release” folders, “obj”, as well as the IntelliSense file (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.  
      
    This includes you missing parts of your project/solution. Be *sure* you are submitting everything you need. An good check is to place your submission on another drive (or better, another computer) and see if you have everything you need.
  + Solutions must be using a x64 Release library C++ project using the **default** settings of Visual Studio 2022 (C++ **14**) using the OpenGL 4.x API (with glfw, glad, and glm).
  + If you use the C++ “**auto**” keyword, you will get a mark of zero. No exceptions.   
    If you use the “boost” library (or any derivative), you will also receive a mark of zero.
  + Your solution may not contain any third party “core C++” libraries (like boost). The same goes for any other libraries you have not cleared with me before hand (like assimp). The *only* libraries that should be present are GLFW, glm, and glad. Note: If you are using some sound libraries from another course, that’s likely fine – just warn me. I’m referring to C/C++ libraries.

|  |
| --- |
| You have until **11:59 PM** on **Wednesday, October 30th,** 2024 to submit all your files to the appropriate drop box on Fanshawe Online (FOL).  **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, then you effectively didn’t submit anything for the exam, there’s nothing to mark, so you’d get a mark of zero. |

See Policy A131 (<https://www.fanshawec.ca/sites/default/files/2023-09/A131%20-%20Evaluation%20of%20Student%20Learning.pdf>) for an explanation of “exceptional circumstances”.

*Please don’t be late submitting.*

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

**“Asteroids are a pain in the space station...”**

Big space ships, far from home, are quite vulnerable, especially to asteroids.

If you’ve seen any science fiction movies, you’d think that an “asteroid field” would be PACKED with asteroids, when in reality, even the closest asteroids in our solar system’s “asteroid belt” are something like 1,000,000 km apart! (<https://www.timeanddate.com/astronomy/asteroid.html>)

In fact, the “dense” asteroid belt is so empty, that NASA doesn’t even consider it when sending spacecraft through it! Apparently, the odds of an asteroid hitting a spacecraft, even in the asteroid belt, is *less than one in a billion*: <https://en.wikipedia.org/wiki/Asteroid_belt#Exploration>

But we aren’t going to listen to science! We’re making a video game!!!

We’re going to assume our ship is going through an asteroid belt that is VERY full of asteroids, like the ones you see in movies like Star Wars:

* <https://www.youtube.com/watch?v=C_DnrceDEI8> (Asteroid Field Empire Strikes Back 1080p HD) or
* <https://www.youtube.com/watch?v=PyUdFRrKQwg> (The Asteroids Strike Back) or
* <https://www.youtube.com/watch?v=p0-0Jod4WrQ> (Asteroid Field vs. Star Destroyer)



In this still image from The Empire Strikes Back, you can see how many asteroids there are; everything that isn’t the Millennium Falcon (grey ship in the middle of the image) is an asteroid.

You are going to use the “Babbage Station” space ship model to simulate a large space station heading through one of these asteroid fields.

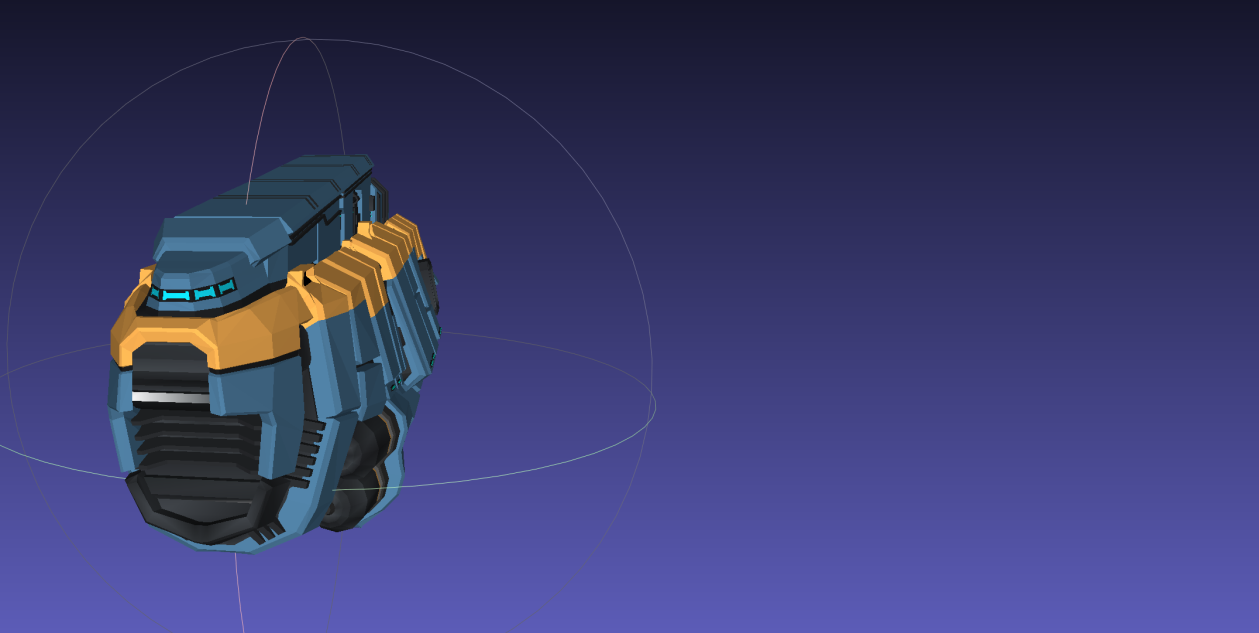
Your ship is the “SM\_Ship\_Massive\_Transport\_01” from Synty Studios (<https://www.syntystudios.com/>), which you’ll find in the models.7z or models.zip files (both archives have the same files).



There are a couple asteroid models as well: Asteroid\_011\_x10 and Asteroid\_015\_x10.

Note that these models are huge, so you’ll likely have to adjust your projection transformation near and far plane to show all of it and to avoid z-fighting.

What you are going to simulate is a field of asteroids heading towards the ***side***of this ship, rather than them hitting the front of the ship (as if the ship was heading headfirst into the asteroid field).  
You can imagine it’s stopped or maybe going through the rings of a planet (like Saturn).

In the image, it looks like the ship is at a slight angle, but what I want is the asteroids to hit *directly at the side, perpendicular to the direction the ship is facing*.

Asteroids flying in this direction

1. (40 marks) Place the ship to the left side of the screen (like the image on page 4) using enough light to see what’s going on. My suggestion is to place one, or more, point light(s) far away from the station, and drop the attenuation, so that it looks like a directional light.   
     
   Every 100 milliseconds (so about 10x a second):  
   * Randomly pick one of 6 “types” which are:  
     + Asteroid\_**011**\_x10 at regular size (not scaled)
     + Asteroid\_**011**\_x10 at ½ (50%) its regular size
     + Asteroid\_**011**\_x10 at 1.5x (150%) its regular size
     + Asteroid\_**015**\_x10 at regular size (not scaled)
     + Asteroid\_**015**\_x10 at ½ (50%) its regular size
     + Asteroid\_**015**\_x10 at 1.5x (150%) its regular size
   * Place this asteroid in a random location far “off-screen” to the side of the ship:



* + - You want to span the asteroids far “off-screen” to the side of the ship. In the image above, you’d spawn your asteroids in the yellow area.
    - This area is a 2D region as “high” and “wide” as than the “length” (front to back) of the ship *at least* as far away from the ship as the yellow arrow above. This isn’t an exact measurement – I just want them the be visible when appearing if you set the camera approximately the angle in the picture in page 4.
    - Again, referring to the camera angle in page 4, the fastest asteroid should fly across the screen in about a second, and the slowest should take about three (3) seconds.   
      Pick a random speed for each asteroid within these ranges.

Note: at this point, the asteroids can pass right through the ship.

1. (60 marks) Simulate the asteroids colliding with the space station.   
     
   As the asteroids fly past the ship, some will intersect (collide) with it.   
     
   Test for the collision of the asteroids at each frame.   
     
   If an asteroid collides with the space station, simulate the collision in this manner:   
   * Draw an “explosion” by placing a small red sphere at the point of collision.
   * Over the next few seconds, change the scale of the sphere so that it grows to about 5x its original size, then disappears.
   * Remove the asteroid from the simulation.
   * Place a small (smaller than the original red sphere, but large enough that it can be seen), dark grey sphere at the place where the impact occurred. It’s supposed to look like a *small* hole or scar mark on the side of the hull (It should be smaller than the windows of the ship).  
       
     It will stay in place, and *not* disappear or move. The idea is that as asteroids collide with the space station, more and more of these dark grey spheres will accumulate, showing the “damage” to the station.
2. (60 marks) Simulate a “LASER” destroying the asteroids before they can hit.   
     
   Choose a key, on the keyboard, to enable this feature. One press should enable it. Pressing the button again disables it. I should *not* have to hold down the key to have the LASER enabled.   
     
   When enabled, have the following happen:  
   * When an asteroid gets “in range” (say ½ the length of the space station), **and** is *going* to hit the station at some point in the future, enable the “Asteroid LASER”.
   * Asteroids that miss the ship are ignored.
   * The LASER is created like this:  
     1. The LASER emits from the middle small hole/port in the side of the ship.   
        It’s the hole/port in the picture below, inside the red circle.
     2. From this location, draw a straight light, made up of a series of very small, bright green spheres, that make a line from this point to the asteroid. The spheres should be close enough that they seem to be a solid, 3D object. In other words, there should be no noticeable space between the spheres.
     3. This line (LASER) should appear at once, stay for a fraction of a second, and then disappear.
     4. Like the “explosion” effect in question 2, draw a red sphere where the asteroid is, make the sphere grow, over 1-2 seconds, then disappear. When the sphere disappears, the asteroid will also disappear the moment it’s hit.



1. BONUS: 25 marks: When the asteroids are shot by the LAZER, they explode into small pieces.   
     
   When the red sphere “explosion” happens, replace the asteroid that you destroyed with a random number of 3-7 smaller asteroid “chunks”. These are 1/5th (20%) the size of the original asteroid.  
     
   These will fly in random directions from the source of the explosion. You can do this by setting the x, y, and z velocities to the same speed of the original asteroid (though this was only travelling in one direction, towards the ship).   
     
   If the smaller asteroids hit the ship, do the same explosion you did in question 2 (red explosion sphere, then a small black permanent mark)
2. BONUS: 25 marks: BUILDING ON QUESTION 4: Change the asteroid generation so that the asteroids are coming from every direction. They still travel in a straight line. This means that:  
   * The asteroids can hit *any* point on the station.
   * There are two LASERs, one on each side of the ship (there’s a matching hole/port on the other side of the ship, too)
   * You have to have the same functionality as questions 3 & 4, in that if the LASER is “off”, the damage must be shown on the station, like in question 2, but with the LASER “on”, the asteroids are being blown apart like in question 4.

That’s it.