**INFO6044 – Game Engine Frameworks & Patterns**

**Final Exam – Tuesday, December 14th, 2021**

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 **eight (8)** questions and **six (6)** pages.
* This exam is a single application/game, where the various “questions” indicate the requirements that you are expected to implement.   
    
  Your mark will be based on the amount of functionally you’ve included in your submission.
* PLEASE delete any temporary files that Visual Studio generates (to reduce the upload size)
* **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).
* You may ***not*** use the “auto” keyword. No exceptions.
* You many have other “utility” libraries, like ones to load textures, models, sounds, etc.
* **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.
* Your code should run under Windows 10 (i.e. a “win32 application”) with Release and x64 (“64 bit”) build.   
  I will *not* be building a “32 bit” application (note: “win32” means “windows API” and *not* “32 bit”).

**Implementing one of the classics, Space Invaders.**

One of the first majorly commercially successful video games was Space Invaders, released in 1978.

While it took a year for one person (Tomohiro Nishikado) to develop, you’ll only have a few hours.

However, keep in mind that:

* He also developed the hardware (circuit boards, etc.) for it to run.
* He may or may not have had an assembler.
* He certainly *didn’t* have a compiler or something like Visual Studio.
* Whatever he had wouldn’t have been able to compile and run the code in seconds.
* There’s no way he had anything remotely resembling a debugger.
* He didn’t have a way to easily draw things onto a screen.

If you want to see the game in action, here it is: <https://www.youtube.com/watch?v=MU4psw3ccUI&ab_channel=GameArchive-NoCommentaryGameplay>

You can also play it online at various places:

* <https://www.andoverpatio.co.uk/21/space-invaders/>.
* <http://www.freeinvaders.org/>
* <https://www.free80sarcade.com/spaceinvaders.php>

**The gameplay is quite simple:**

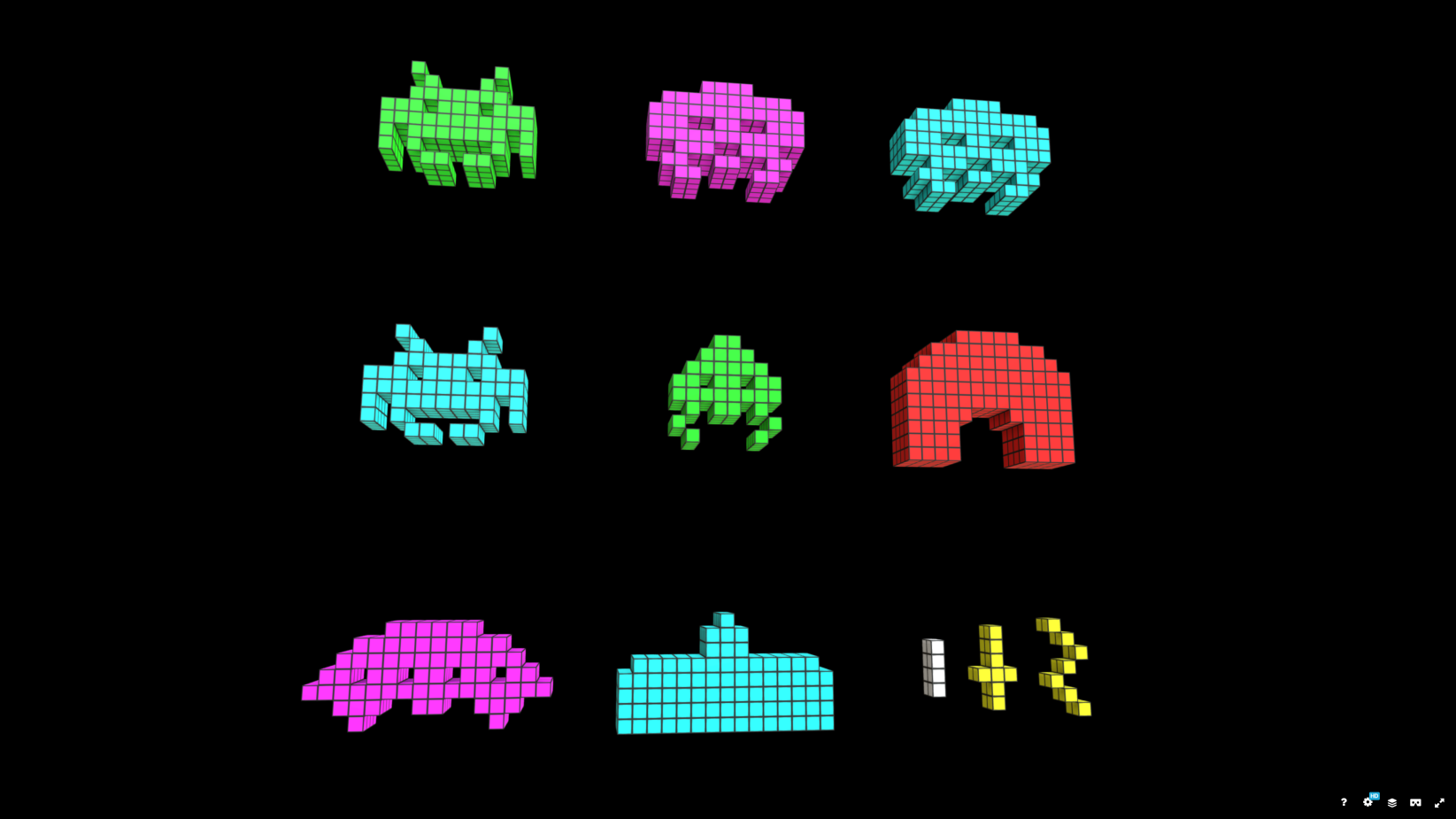
* A grid of aliens are positioned above the player (the “tank” or shooting thing at the bottom).
* They move one increment at a time in one direction, get to the end, then reverse.   
  Each time they reverse direction, they descend a little.
* They occasionally shoot/drop bombs.
* The player cam move their “tank” left or right across the bottom of the screen.
* They can shoot, but only up, and only one projectile at a time.
* If this projectile hits an alien, it explodes.
* The fewer aliens, the faster they move.
* If an alien bomb hits the player, the player explodes.
* If the aliens reach the bottom of the screen, the player loses.
* Every so often, a “mother ship”/UFO moves across the top of the screen.

There is also four “shields” or “bunkers” that the player can hide behind.   
These get gradually destroyed by the alien bombs.

There are also so rudimentary “animations” that the aliens and bullets exhibit:

* The aliens have two “states” which they alternate as they move.
* The bombs make a “squiggle”/worm sort of shape as they fall.
* The aliens are replaced with an “explosion” graphic when hit.

You are going to make a stylized version of Space Invaders, replacing the “pixel art” style of 1978 with 3D cubes/voxels, like this:



This image is from the models from this site: <https://sketchfab.com/3d-models/voxel-space-invaders-pack-c97fde5a91f242ed956b518bb73ea610>

There are also other example models in the “Space Invader #1” through #3 folders.

However, you are not going to use those models. Instead, you are basing your models on the files located in the “Space Invader Block Files” folder and the “Invader\_Single\_Cube.ply” file.

**..X.....X..**

**...X...X...**

**..XXXXXXX..**

**.XX.XXX.XX.**

**XXXXXXXXXXX**

**X.XXXXXXX.X**

**X.X.....X.X**

**...XX.XX...**

These files are quite simple. Here’s the “SpaceInvader\_I1\_block\_pose\_1.txt”:



...which corresponds to this model:

You may generate your models in one three ways:

* 1. Use the information in the “Space Invader Block Files” files to draw an grouping of small cubes (the “Invader\_Single\_Cube.ply” file),   
       
     ...or...
  2. Use the “INFO6044\_SpaceInvader\_Block\_to\_Model\_Gizmo” program that I’ll provide that will take some very simply configuration files and generate ply models.   
       
     ...or...
  3. Somehow generate the models yourself - but they have to *look like* they are made of 3D blocks.

Here’s the features/functionality and how many marks they are worth (i.e. the “questions”):

1. (100 marks) Display a static start screen for the game. This should:
   * Look like 25 seconds into the <https://www.youtube.com/watch?v=MU4psw3ccUI&ab_channel=GameArchive-NoCommentaryGameplay> video.
   * You do *not* need to show any of the text (there is bonus for this later in the exam)
2. (200 marks) Move the aliens from side to side. This should:
   * Happen in a “stepped” manner, like in the original game. That is they should “jump” one “pixel” to the left/right. Of course, we’re using cubes, not pixels. In the original game, this originally happens every second or so.
   * When the aliens get to one side, they should go “down” one “pixel”, and reverse direction.
   * At each “step”, they should change “animation”, in that they should change to the other “pose” that they have.
   * They movement of the aliens should “ripple” as they move, starting from the bottom row to the top. Notice that they all don’t move at once, but in sort of “waves”.
   * If they reach the “bottom” (at the top of the “shields”/”bunkers” are, they should stop).  
     (This would mean the player “lost”)
3. (50 marks) Move your “tank from left to right and shoot:
   * Assign two (2) keys to your tank at the bottom. One moves left, the other right.
   * When no key is pressed, the tank stops moving.
   * Assign another key (space bar, maybe?) to “shoot” a “missile” up at the invaders.
   * You don’t have to worry about the invaders being hit – the missile can pass right through.
   * Only allow one shot at a time.
4. (50 marks) Add the UFO/mother ship at the top:
   * Every once in a while, have a UFO ship appear at the top, move along to the other side and disappear.
5. (200 marks) Implement the player hitting the aliens:
   * If the player’s missile hits an alien, make it “explode”.
   * You can do this by replacing the alien “model” with one of the “explosion” models.
   * (**BONUS 100**: Have each “pixel”/3D block of the alien fly apart, sort of like a particle effect. This can be the same explosion every time or random)
   * As there are fewer aliens, their movement should speed up.
6. (200 marks) Implement the aliens dropping bombs on the player:
   * Random aliens can drop “bombs” on the player.
   * This can be any alien or just the “lowest” alien at that row.
   * Note the bomb has a very basic “animation” where it sort of “wiggles” its way down the screen. You can implement this by changing the model each “pixel” down the screen it goes.
   * If it hits the player, the player should “explode”.
   * You can implement this by replacing the player’s model with a *combination* of the player’s model *and* the explosion model, or with a particle effect like in question 5 (**BONUS 50**).
7. (**BONUS: 100 marks**) Implement the shield/bunkers wearing down:
   * In the original game, an “explosion” of pixels eats away at the shields.
   * The way the collision detection worked was to see if there were any “green” pixels there and if there were, the “bomb” stopped (fun fact: it wasn’t actually “green” – it was a black and white screen, and there was a transparent green plastic on the CRT display. Pretty sexy high technology stuff. You have to make it green, though.
   * You can do this by drawing a number of 3D cubes instead of the “shield” models, removing them as “bombs” hit them.
8. (**BONUS: 100 marks**) Implement scoring and player count using 3D cube models:
   * Implement the scoring display at the top, using the same “3D cube as pixel” style.
   * The “SpaceInvader\_Numbers.txt” contains a grid for that.

**That’s it.**