**Danil Ulmasev**
**CART 253**
**Concept and technical pitch.**

After spending some time thinking about the final project and elaborating on various video game concepts, the conclusion was made to proceed with an idea that was already previously made by other video game programmers. The reason for that is to avoid any sort of possibility to reinvent the wheel and come up with extremely complicated concepts in these early stages of video game programming learning. To take something that was already made and make a personal version of that is a good way to examine the processes and of course learn the deeper about the programming language. Of course, the idea of not picking something overcomplicated also applies to the idea of trying to copy already existing game; it would be silly to try to code World of Warcraft at that level of experience unless someone is absolutely natural in it. Thus, the aim of this project is to try to rebuild one of the classic arcade games which have relatively basic controls, fun but simple graphics, score detection and user interaction.

In 1978 a Japanese video game developer Tomohiro Nishikado released an arcade shooter game Space Invaders. The purpose of the game was to defeat as many waves of aliens as possible and shoot them with the laser cannon from the player's spaceship to earn the killing points. The game was inspired a lot by popular stories such as The War of Worlds, Star wars and fellow arcade game cousin Breakout. Additionally, Tomohiro Nishikado had to develop some custom parts and controls in order for him to finalize the game. When first released, Space Invaders was very successful.

For this project, the idea is to imitate the gameplay of the original game and hopefully add few personal modifications. With the basic knowledge of video game programming and all the concepts that were covered in class, this game should be achievable. The challenges that may and will occur during the coding process are such as collision detection and collision response, some sort of game engine, time, and score tracking.

## Gameplay overview.

This game will be based on the original interface where enemy ships come from the top of the screen as waves and the user's spaceship is at the bottom of the screen with its laser cannon pointing at the enemies. The game will start with the player's spaceship in the empty space full of some sort of stars in the background waiting for the incoming forces to enter the screen from above. Enemies should come by waves in lines of five or six units (undecided) and the number of lines will determined according to the time passed in the game. This means that the longer a user is playing the game the more enemies will come throughout the game.

Each enemy will have a classic concept of hit points or as it is often referred to as HP. It will require hitting the enemy ship with few bullets in order to destroy it. The Same concept applies to the user’s spaceship, any sort of collision will result in loss of one hit point whether it was the enemy's bullet that hit the player or the enemy’s ship itself. A number of hit points are undecided at this moment that player and the enemies will have.

The controls of the ship will be on the keyboard where the player will have the possibility to move the ship up, down, left, and right. Having these controls also makes it possible to move the ship diagonally. The laser cannon will be operated by the “space key” on the keyboard and shall release one bullet every time the key is released to avoid the continuous arrays of bullets shooting at enemies otherwise this would make this game too easy.

The game should potentially track the player’s score according to the enemies that were killed and how far in the game the user was able to go to.

**Technical parts.**

The game will consist of several objects that will define all aesthetics and interactions. To begin with, this game will need a simple engine that will control the environment of the game such as object’s position on the display screen, movements and whether the object is dead or alive. Following that, the game will need units to play with. For that, there will be several classes that will make this happen.

One class will define user’s spaceship which will have its coordinates on the screen, some sort of movement, hit points and parameters for the keyboard. The next class will define an enemy ship, which will also have its coordinates, some sort of movement, hitpoints and place of birth. This brings this description to the next class which will define the place of birth of the enemy's ship. This will be an invisible creator of enemy ships which will be located above the screen where users cannot see. It will still have x and y coordinates, however, they will be off the screen the purpose of that class is to set the pinpoint location for the enemy ships to appear to. Next important class will define bullets. Bullets are fairly simple, their purpose is to come out of any ship and fly in the direction that they are supposed to fly. This class could be potentially shared by player’s spaceship and the enemy. Bullet’s place of birth will be at the x and y coordinates of any ship. There will be few other classes which will serve as game graphics and decoration. The background will need a starfield and maybe some occasional asteroids that will pass through to make the game look more exciting.

An interesting way to make the aesthetics look nicer is to draw or build objects with the use of several objects. The plan for that is to create a good looking spaceship that has a nose, tail and wings and the same idea would apply to all ships (research needed).

So far, the game will be timed according to the number of frames counted since the beginning of the play. However, more in-depth research will be needed to better understand how to manipulate the time and frames in the game such as this.

And last but not least, the overall score will be tracked according to a number of enemies that were killed during the game. One point to the score will be added each time an enemy ship dies; once the user's ship dies the final score will be provided to the player and shall reset to zero when the game restarts.

**Graphics and sound.**

Graphics in the game will be fairly simple, 2D with simple colours and probably without imported images. The depth of the image, however, could be manipulated with the brightness and transparency of the colour and maybe the size of the object. The game design will definitely avoid basic geometry shapes as a whole unit or character and instead use those basic shapes to create complex figures.

The sound is definitely necessary, it always takes any sort of game few levels above and makes the experience much more enjoyable. All sounds will be custom built with external synthesisers and sound files will be saved as samples. The good thing about this game would be to create sounds that are identical or similar to the original game to maintain the mood of an antique game.

This should sum up the general overview and ideas of the final project and potential steps in creating the game.