# **Object Oriented Programming Assignment 2**

## **Features**

This game will be a 2d space shooter with 3d models. The object of the game is to kill as many enemy ships as possible.

**Fuel**: every time the player moves they will use a bit of their fuel. The player will be able to buy fuel at the in-game shops that will randomly spawn in the game every now and then.

**Mining**: The player will be able to mine rocks for credits (money). Rocky asteroids give less money than metallic asteroids and the bigger the rock the more credits the player will get.

**Death**: the player will have one life. If they get hit by an enemy’s bullet they will die.

**Shop**: Shops will randomly spawn in the game. The player will be able to buy:

* Bullets
* Upgrade bullets
* Fuel

**Movement**: the player ship will be able to move forward only and rotate left and right.

**Shoot**: The player will be able to shoot enemy ships. If a bullet hits and asteroid or shop it will explode and not do any damage, if it hits an enemy it will do an x amount of damage to the enemy and then explode.

**World** **Generation**: The world will procedurally generate around the player. The in-game map will be categorized into tiles, each tile will have a set amount of game objects in it

1-3 asteroids

0-3 enemy ships

* 1. shops (shops have a 1/10 chance of spawning)

e.g.

int randNum

random(1, 10)

if randNum == 1 then spawn

## **Models**

* Powerups
* UI
* Asteroids
* Enemy ships
* Player ship
* Shop
* Bullets
* Explosion

## **Scripts**

### **Player Controller**

This script will control the players movements based on the users input.

**Rotation**: When the ship rotates the 3D player model should gradually tilt in the direction. A max tilt should be specified and when the ship reaches that tilt it will not tilt to the side anymore, even if the ship is still rotating.

**Tilt**: The 3d model should tilt slightly in the direction it is rotating.

**Movement**: If the player presses “w” the ship will slowly accelerate until it reaches its maximum speed. If the “w” is not being pressed the ship should slowly de-accelerate until it reaches a speed of 0.

**Detect Collision**: if the player collides with any other game object. It should not be able to fly through it but instead collide with it.

**Hit by bullet**: If the player is hit by a bullet the player ship should be destroyed and a game over message should be displayed on the screen.

### **Shoot**

When a player left clicks on the mouse the player ship will shoot bullets out of it turrets.

**Shoot**: When the player shoots a bullet the shoot script will get the ships current bullet type and shoot that bullet.

**Bullets**: each bullet type will have its own behaviours ( straight moving, heat seeking, etc )

### **Mining**

When the right mouse button is clicked it will create a mining laser that can mine asteroids if they are in range.

**Mining laser**: draw a mining laser in the direction the ship is facing. If it hits an asteroid. Mine the rock for 3 seconds then destroy the object and remove it from the tile.

**Destroy the mined asteroid**: Once an asteroid has been mined, destroy the object, increment the players credits by the asteroids credit value then remove the asteroid from the tile so it does not respawn.

### **Enemy01 Behaviour**

This enemy type will have a simplistic AI. The ship will not move anywhere in the world but will rotate and shoot out bullets around the ship in 30-degree increments.

**Rotate**: The ship will continuously rotate at a specified rotation speed.

**Shoot**: This enemy type will shoot 12 bullets at 30-degree increments around its ship model.

**Hit by player bulle**t: If the enemy is hit by a player bullet then:

Health = health – bulletDamage;

### **Explode**

All bullets will explode when they collide with a game object. Depending on the game object they collide with will determine if they do damage or not.

### **Tile**

The game world will be divided up into tiles. Each tile size will be the 2D in game size of the camera in unity (I think the default is 5). Each tile will have an ID to identify it. A tiles ID will be its centre position in the tile. E.g. when the game first starts the first tile that will load will be the tile with the ID 00 because the x = 0 and y = 0. The bounds right hand side of the tile will be the ( centreX position + camera size/2) the left hand side bounds will be ( centreX position - ( camera size/2 ) etc.

### **Generate world**