Asteroids Report

To run the game, run Game.main()

**Controls**

UP = Thrust

LEFT/RIGHT = Rotate Left/Right

SPACE = Shoot bullet (limited to 2 bullet per second)

SHIFT = Activate Shield (Lasts 10 seconds)

**Objects**

 Player Ship

 Asteroids

 Teleporter- Random position given when touched

 Shield Pickup (Shields pickups last 15 seconds before disappearing)

 Enemy Saucer- Extremely accurate. Avoid at all costs

 Force Field- Adds random velocity to ship

**Scoring**

Destroying asteroid = 500 points

Destroy saucer = Current Level \* 750

**Main Design Choices**

During the development of the game, I decided that game content would be generated dependant on level. I decided to create a class that would generate the GameObject array at the start of the level. It also includes an update function that adds objects such as shields to the array when appropriate.

I decided I would like the game to use a consistent theme throughout. I tried to match the player ship and enemy ships design by using sprites that were similar to one another in terms of design era. The programming of the game has a large emphasis on inheritance as every object on screen extends the GameObject class. This makes them easy to manage therefore functions such as collisionHandling can be carried out effectively using a nested loop. This means in game collisions can be calculated effectively because of no need to hard code detection.

**Playability**

The number of Asteroids per level is level+1. The Saucers health also works in a similar way, dependant on level. i.e. The saucer must be hit once in level 1, twice in level 2, three times in level 3 and so on. Shields pickups would be limited dependant on the level ensuring gameplay is balanced in terms of difficulty. The player starts with 3 lives, so they can get a feel for the game but at the same time not struggle too much or find it too easy.

The player is awarded a new life every 10000 points. This helps them through boss stages in later levels. This is a key feature as future levels become increasingly difficult. The max speed of the player ship is limited although max speed will likely never be reached as collision with an object is inevitable. The initial speed of a player’s bullet during early stages of development was a lot higher than it is now. This is simply because the game was too easy as a player could shoot asteroids before came close to the players ship. Bullet timeout is set to 5 seconds meaning the player does not have an unfair advantage

**Appraisal**

During the development process, I ran into numerous problems. The Vector2D class was particularly difficult to understand at first. However, with a little perseverance, became easy to understand. The Game classes update method became very intricate and complicated. This was because of the number of steps there was to complete in each iteration. Sectioning some parts into individual functions helped to declutter it. Small parts of the game such as shields gave small problems. When an asteroid collides with a player ship when a shield is currently in use, the player ship would immediately become hit. This was because there was no cooldown period. To fix this issue, I created a cooldown period of less than a second to ensure the shield functionality was properly fulfilled.

I am particularly proud of the of the saucer controller. This is because it accurately shoots at the player ship. It does this by taking the player ships position and velocity and perform vector mathematics to essentially predict where the player ship will be. The player ship will be hit unless the player takes appropriate action after the saucer has taken a shot.

I believe my final submission has no problems that need to be addressed. If I were to do this project again, I believe I would change how I structured the program from the outset so that so many changes are not required. I thoroughly enjoyed this assignment as it has given me an insight into game design and programming.