# Assessment Task 4 – Retro Game Design Document

Asteroids go brr

A screenshot of a video game

Description automatically generated with medium confidence

Contents

[Assessment Task 4 – Retro Game Design Document 1](#_Toc136590683)

[Design Concept 3](#_Toc136590684)

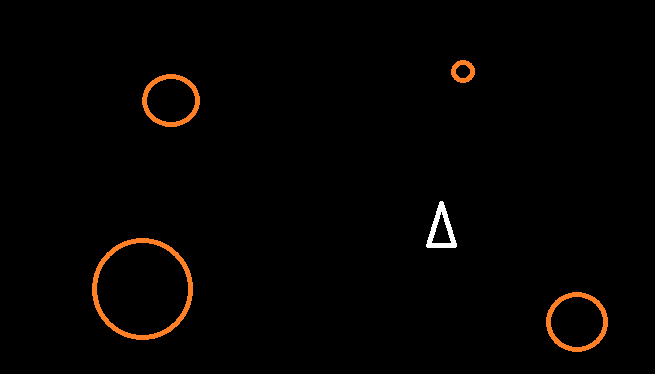
[Data structures and algorithms used. 4](#_Toc136590685)

[Design of Game 5](#_Toc136590686)

[Version Control 6](#_Toc136590687)

## Design Concept

The design of asteroids is simple. We just need a triangle representing the player that shoots at circular objects which break into smaller version of itself or gets destroyed if at its smallest.



In the retro game, the asteroids come in different shapes. This would be possible in Raylib using textures, which I didn’t use because I forgot, but also because I wanted to simplify the collision detections, so we just need to use some basic circle detections.

A picture containing black, text, font, sketch

Description automatically generated 

The ship is displayed using a triangle and shoots small little circles for bullets. The player starts with 3 extra lives, with extra lives being given every 10,000 points gained. The ship needs to move forward and keep that momentum whilst also being able to rotate freely so you can drift around and shoot asteroids. We also should add a brief period of immune time when getting hit in order to make things fairer to the player, otherwise getting hit would melt away all your lives in a snap.

## Data structures and algorithms used.

Raylib by default has a Vector2 struct built in, so I can use that in order to track multiple aspects of the player, the bullets, and the asteroids. Specifically, we’ll be tracking their positions, and their directions using Vector2’s.

A screen shot of a computer program

Description automatically generated with medium confidenceA screen shot of a computer program

Description automatically generated with low confidence

Storing the values within structs and making the variables public, I can access these from my Game.cpp file, and modify them as needed. Since the asteroids split, I can spawn just the large asteroids first, and respawn them later when enough small asteroids are destroyed. This will be done to prevent over-spawning asteroids. Spawning the asteroids is easy, we just need to select a random position and direction before moving them. This is easily achieved via Raylibs GetRandomValue() function

## Design of Game

## Version Control