Southampton Solent University

Computer Games Programming (DAC526)

DAC526 Assessment No.1 Design Project

Keron Sepp

18.10.2018

Table of Contents

[Table of Figures 1](#_Toc527585692)

[1. Identifying tasks 2](#_Toc527585693)

[2. Design tasks 0](#_Toc527585694)

[2.1. The main challenges in creating this project. 0](#_Toc527585695)

[3.11. UML Class diagrams 1](#_Toc527585696)

[3.12. Pseudocode 3](#_Toc527585697)

[3. Justification 4](#_Toc527585698)

# Table of Figures

[Figure 1: Flowchart of the software design. 0](file:///C:\Users\Keron\Desktop\171018\171018\Keron_Sepp_DAC526_AE1_Design_Project.docx#_Toc527585699)

[Figure 2:MindMap design 0](file:///C:\Users\Keron\Desktop\171018\171018\Keron_Sepp_DAC526_AE1_Design_Project.docx#_Toc527585700)

[Figure 3:Inheritance from Creature class 1](file:///C:\Users\Keron\Desktop\171018\171018\Keron_Sepp_DAC526_AE1_Design_Project.docx#_Toc527585701)

[Figure 4:TileClass with inheritance 1](file:///C:\Users\Keron\Desktop\171018\171018\Keron_Sepp_DAC526_AE1_Design_Project.docx#_Toc527585702)

[Figure 5: Classes to handle the game system 2](#_Toc527585703)

[Figure 6: Pseudocode part 1 3](#_Toc527585704)

[Figure 7:Pseudocode part 2 3](#_Toc527585705)

# Identifying tasks

The game is going to be 2D side-scroller made in C++ using Object-oriented programming approach. The player can move, jump and has to avoid obstacles and non-static enemies to reach the end of the level. Player may interact with score items and various power items in order to progress.

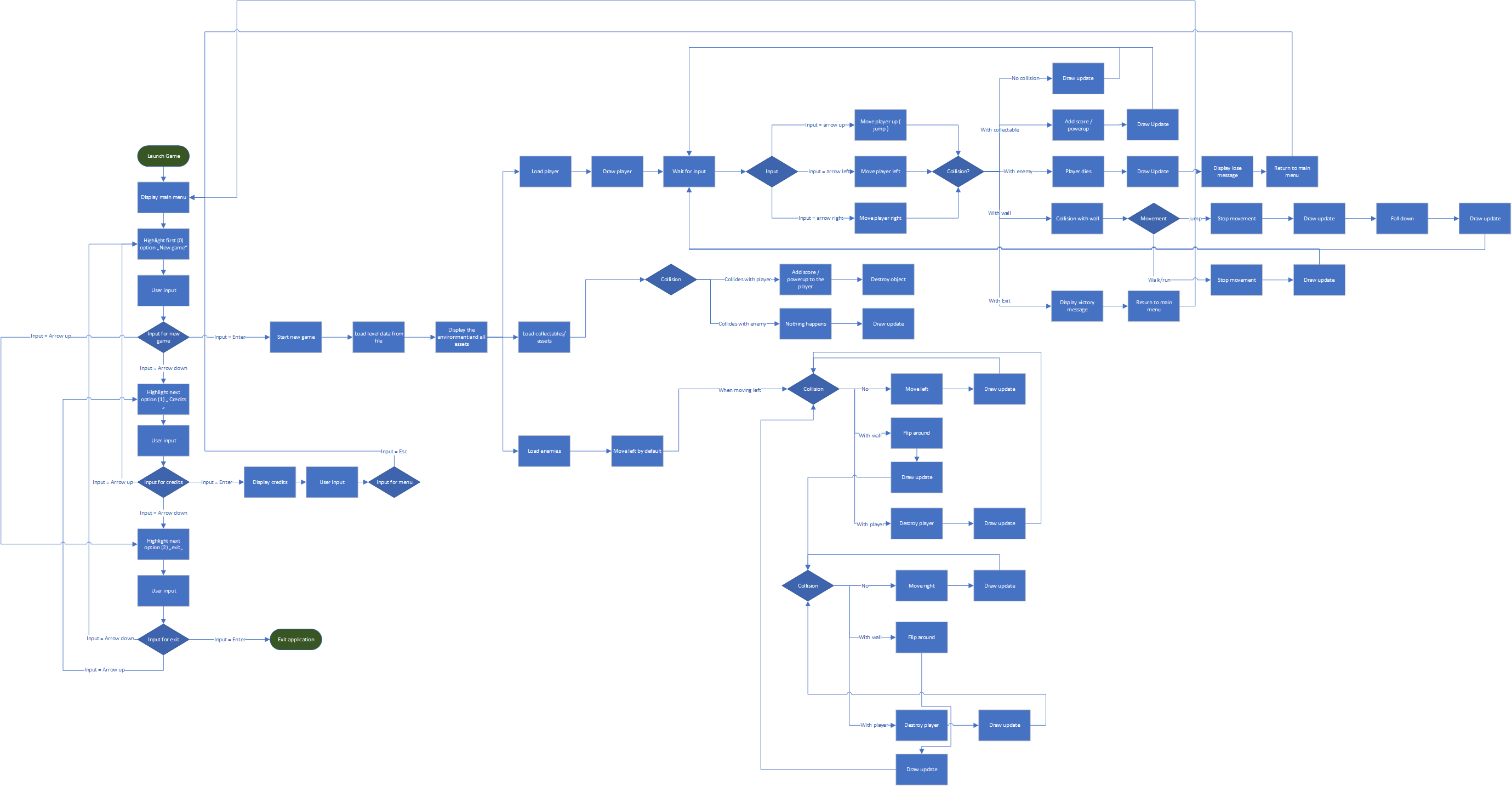


Figure : Flowchart of the software design.

# Design tasks

## The main challenges in creating this project.

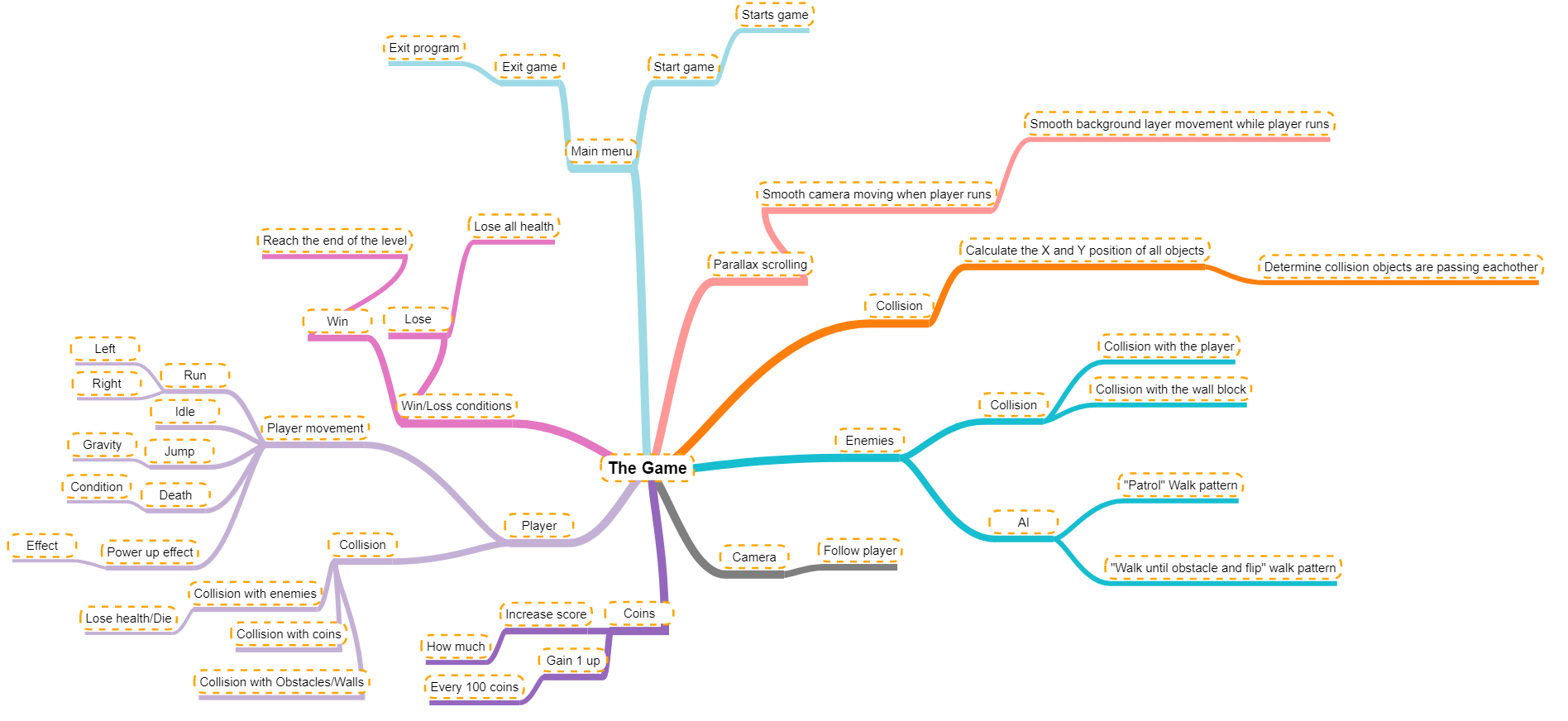
In order to have playable core features of this game 2D world and a level must be created. The second requirement is a level start and end function. To have some interaction in the game the player and camera must be implemented. The player needs set of functions in order to move and collision to interact with the level, and other features. That covers the core mechanics of the game that are needed. Additional features are monsters with AI functions and other features such as score coins, powerups, walls etc.

Figure 2:MindMap design

### 3.11. UML Class diagrams

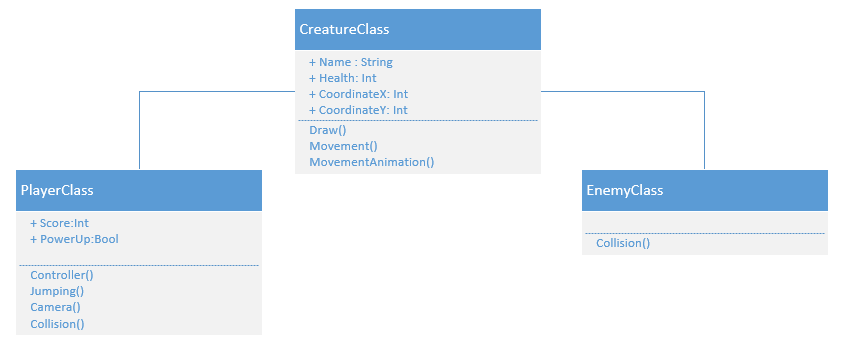
The player and the enemy share many functions together which makes it reasonable to inherit them.

Figure :Inheritance from Creature class

The TileClass is used to block player or enemy movement. ObstacleTile inherits from TileClass as it has the same features, but also hurts the player, or enemy.

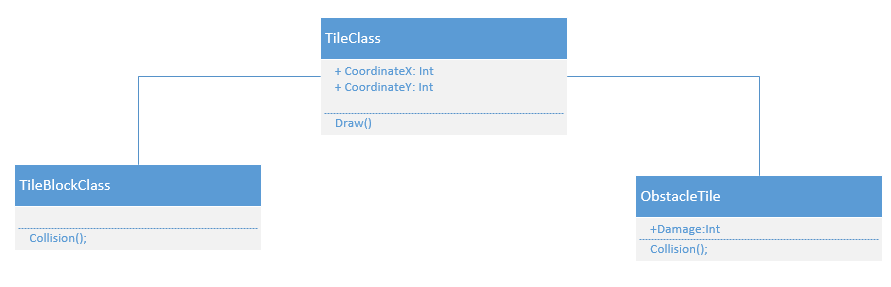


Figure :TileClass with inheritance

GameUIHandler class will handle displaying everything in the game. It is connected with GameHandlerClass as the class deals with loading data, all the data, and also for saving data. Game Systems is linked with previous classes as it has function that must be placed according to the game data.

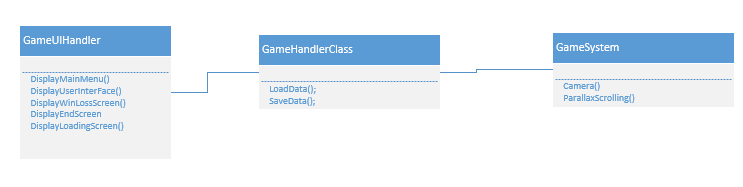


Figure 5:Classes to handle the game system

### 3.12. Pseudocode

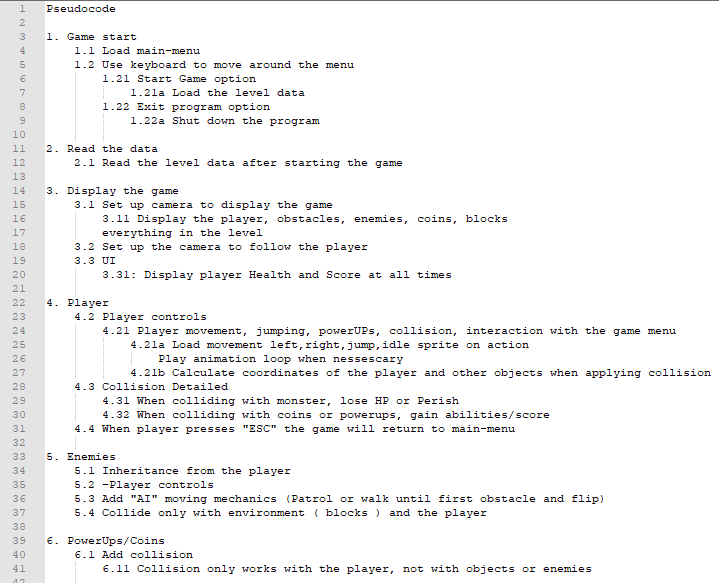


Figure 6:Pseudocode part 1

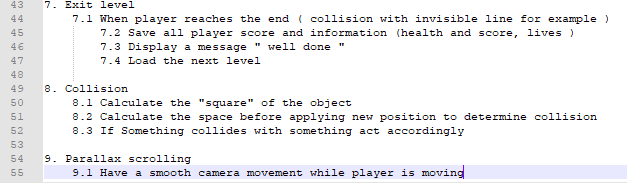


Figure 7:Pseudocode part 2

# Justification

The goal of this design is to create a really basic game mechanics. The reason behind that is to have working template 2D platformer that could easily be built into more advanced 2D game. I think this method is simple as it only requires the fundamental parts of the game to be created like collision, camera, parallax scrolling and simple player, A.I and asset mechanics to make the game functional. The design aims for efficiency and that is the reason why more advanced features are not designed like advanced A.I with pathfinding features, sound or changeable options in the menu.