### ***Technical Design Document***

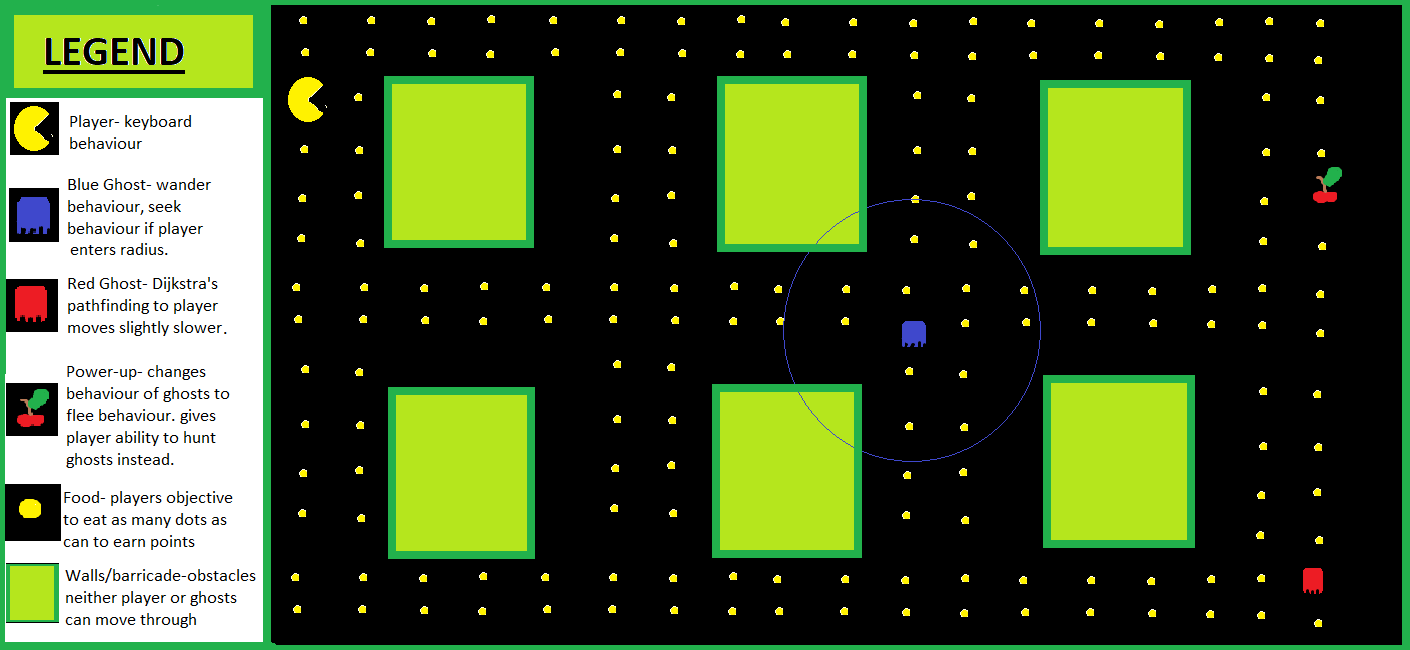
**Game Overview**

**Objectives**

Players objective is to eat as many of the yellow dots as possible increasing score, whilst avoiding ghosts along the way who are trying to hunt the player down in various ways.

**Technical Goals**

* challenging AI that changes behaviour throughout the game based of AI environment.



**Game Objects and Logic**

**-Player**

* Player Is moved using W, A, S, D keys.
* Player controlled Behaviour whose purpose is to eat dots without being eaten by ghosts.
* Force controlled movement.

**- Red Ghost**

* Dijkstra's pathfinding towards Player.
* Flee behaviour when player is in range and powered up.

**-Blue Ghost**

* Wander randomly around maze.
* Flee behaviour when player is in range and powered up.
* Seek behaviour when player in in range and not powered up.

**-Items**

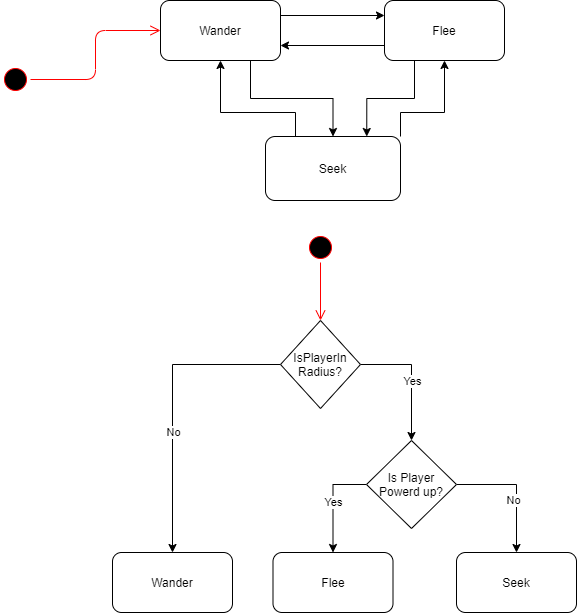
* Fruit to power up player, once powered up changes the behaviour of all nearby ghosts to a flee behaviour. Player has the option to chase the ghosts instead for limited time gaining bonus score.

**-Walls**

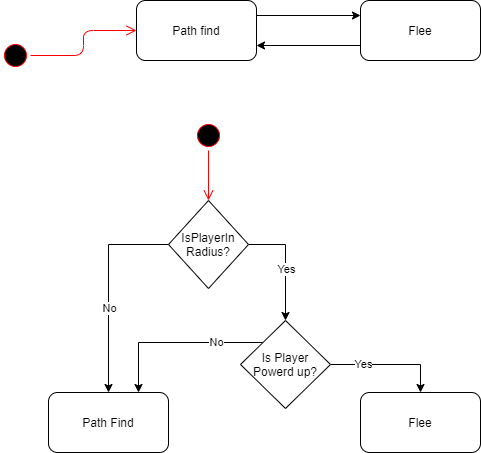
* Boundaries that player and enemy both cannot get pass.

**Artificial Intelligence Flow Charts**

***BLUE GHOST***



***RED GHOST***



**Development Environment/Assets**

**-Asset List**

* Insert assets here

**-Game Engine/Third party libraries**

* Raylib

**-IDE**

* Visual Studio

**-Source Control procedures**

* Git - SourceTree

**-Third Party Libraries**

* Raylib

**-Licences**

|  |  |  |
| --- | --- | --- |
| **Asset** | **Library file location** | **Licence** |
| Raylib | NA | MIT |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Code suitability**

**What hacks did you use to implement the solution, and what approaches could have been used to make it less “hacky”**

**Technical impact**

**- was the searching of visited / nodes on stack optimal, O(N) o(N^2) etc… could it have been improved..**

**-Graphics**

* Top down 2D