### Technical Design Document Template

***Rush Hour***

***Elevator Pitch:***

The game is a 2D top down game which is controlled purely by the ai. This game will involve robbers, police. The game will involve the robbers running around the map to find and collect money bags. The police will be wandering around the map. If they see a robber, they will chase after them. If the robber is caught, they disappear from the map and would not be able to continue stealing from the police. The ai needs to get as many money bags as possible before they get caught.

These agents or characters will have the following:

**Robbers:**

* Follow Path
  + This follow path will be for getting to a certain location in the map. To collect money bags
  + The follow path will occur until the police is within distance of the robber.
  + The flee behaviour will then be implemented when the police are around
* Flee Behaviour
  + This flee behaviour will occur when the robbers are spotted by the police.
  + It will continually flee until the police are out of range of the robber. It will revert to follow path afterwards.

**Police:**

* Wander Behaviour
  + The wander behaviour will move the police around the map at random.
  + This behaviour will continue until the police are within range of the robbers. When this occurs, the seek behaviour will occur.
* Seek Behaviour
  + The seek behaviour will occur when the police are within range of the robber.
  + The police will seek towards the robber until they caught them.
  + When the robber is caught, or they are out of range, the police will be return back to a wandering behaviour.

**1.0 Revision History**

|  |  |
| --- | --- |
| Version | Description |
| 1.0 | Initial Document |
| 2.0 | Final Document |

**Development Environment:**

**Game Engine/ Third Party Libraries**

Raylib

**IDE**

Microsoft Visual studio

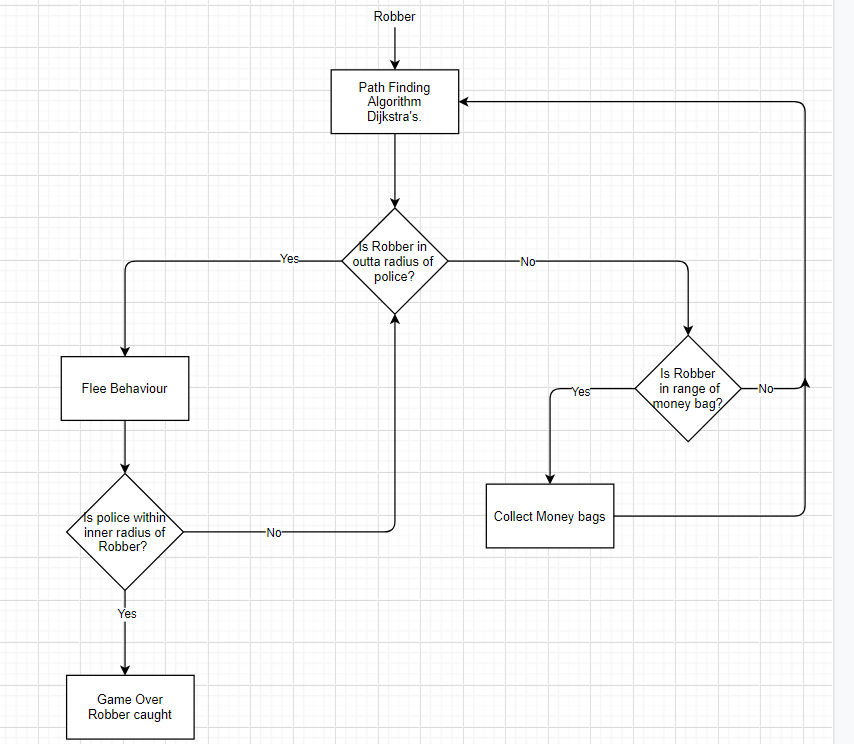
**Source Control**

Git-GitBash

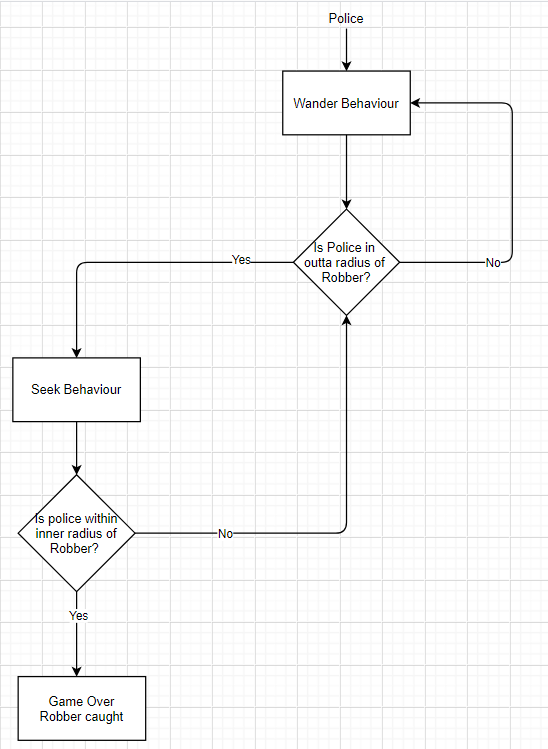
**Other Software**

Tiled

**Diagram of the Robber behaviours within the game.**



**Diagram of the Police behaviours within the game.**



**7.0 Physics**

In terms of interactions, if the robber comes within the radius of the police, the robber will use the flee behaviour to run away from the police whilst the police will use the seek behaviour to run towards the robber. As the robber is faster than the police, the robber will eventually run away from the police. When the robber is out of the radius of the police, the robber will return to their path finding algorithm to find the money bags.

**11.0 Interface - Controls**

The player would be able to press tab to view all the nodes and movements that the robbers and police will be performing whilst the program is running.

**14.0 Asset List**

In relation to the Tiled software, I will be using certain tiles to make the map, the robber and the police.

***Third Party graphical implementations:***

* **Evaluation of the framework’s suitability:**

Within the solution, I had to make the project initially very hacky. This was to sustain its suitability to work to the requirements of the document. However, reflecting upon the solution, the way to make the project less “hacky” would be to convert variables to member variables so that they can be controlled and changed during runtime instead of in console. This can be considered as controlled variables that would affect the motion of the game, such as the players speed and radius.

* **Technical impact of the framework:**

In terms of checking every node within the game for path finding for visited and stack from one position to another, it is one of the slowest methods to find a fast path to the end location. The use of the O(N) is quite slow as the larger the project gets, the longer the time it takes to find the path. For example, if the path is twice as long, the time will be twice as long. The best approach to the problem would be using the O(logn) notation as the larger the path or the project gets, the least amount of time is needed to find the path.

* **Framework licensing, any licensing issues:**

**Map of the Game:**

**Key:**

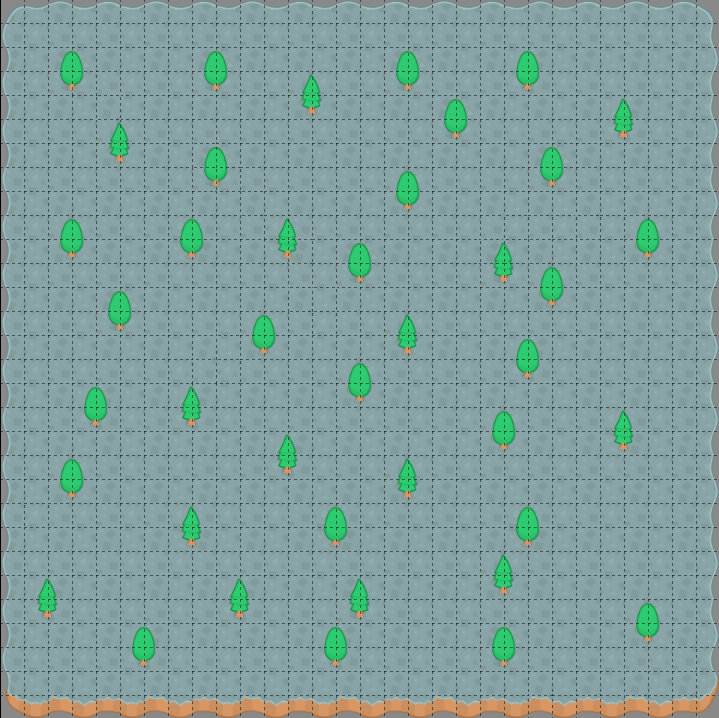


Ground:



Obstacles:

Coins(Points):

Player/Robber:



Police: