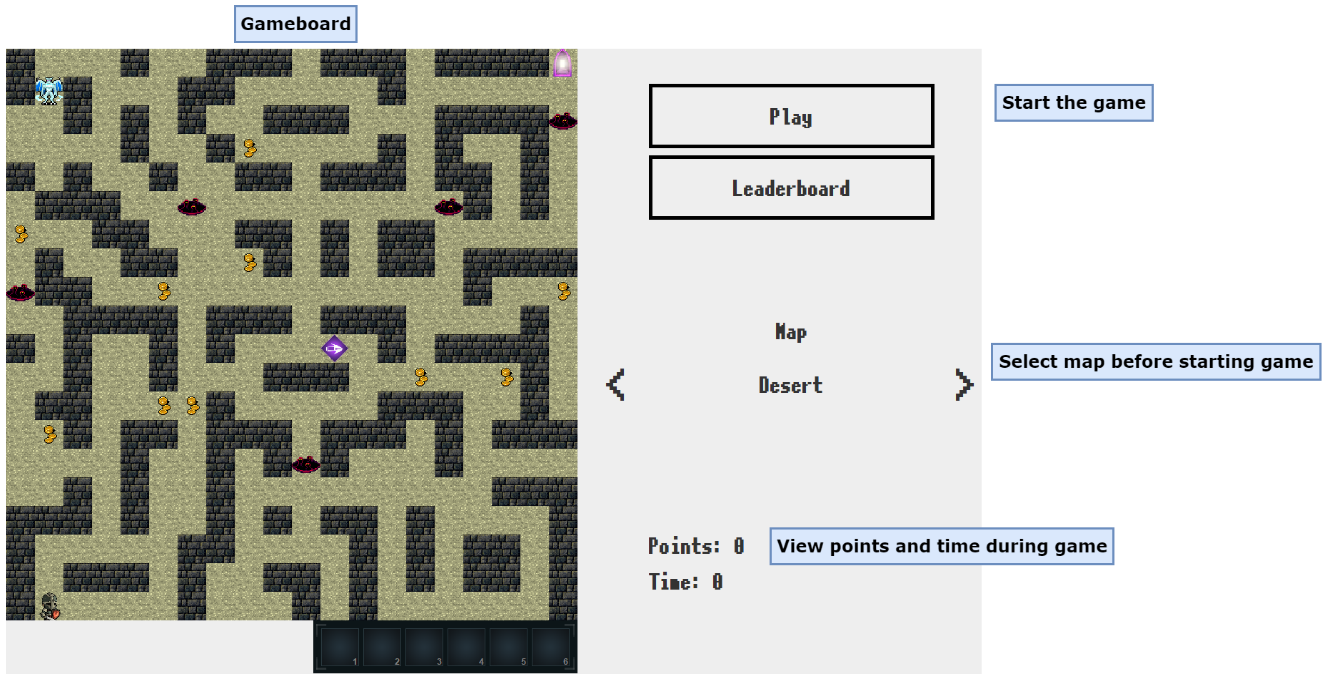
The final product of our group is a single player game called Knight Escape which follows a similar playstyle as pacman. The game has additional features such as having a leaderboard, the ability to choose different maps and the ability to use power ups after collecting them. The final product contains several adjustments compared to the original design. Our group decided that the option to change the skin and difficulty of the game is not needed since it does not add much changes to the game. Additionally, we decided that the option to have a tutorial of the game on the menu is not necessary since the game is quite simple and we want the players to learn the game as they play. As a result, the game does not have a separation between the game screen and the menu since the menu would only contain the option to choose maps. Furthermore, we made changes to the leaderboard system such that highscores would be automatically saved to the leaderboard rather than require adding manually. In our original design, the bonus reward can spawn an infinite amount of time, randomly throughout the game. However, in the final product, we decided that the bonus reward will only spawn once because it gives us more control over the range of scores that the players can get. Our group also decided to make the character able to move slightly faster than the enemy in the final product because it makes the game less frustrating to play. In addition, the final product contains power ups which are not part of the original design.

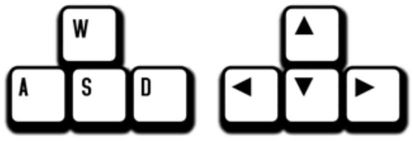
Since our group only has two members, we were not able to implement some of the game mechanics in order to focus more on the core features. However, despite the unfortunate circumstances, we adapted and learnt how to work better under pressure. As a result, we were able to deliver most of our works on time. Through this project, we had the opportunity to learn and use tools such as JUnit or Maven which we had no experience using before. Furthermore, the project gave us a better understanding of the software development process and the importance of planning in software designs. It also allowed us to practice lessons that we learnt in class such as the design pattern and testing. Furthermore, the project helped us learn more about problem solving and how to approach different problems that we faced throughout the implementation process. We also had the chance to apply knowledge from other classes such as using the breadth first search algorithm.

Tutorial

1. Setting up the game
   Once the game is launched (build and run instructions in the readme) the following interface will be shown:



2. Game mechanics



Controls

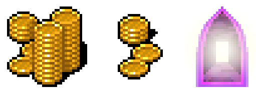
- Movement: WASD or Arrow Keys
- Use item in inventory slot: 1-6 number row

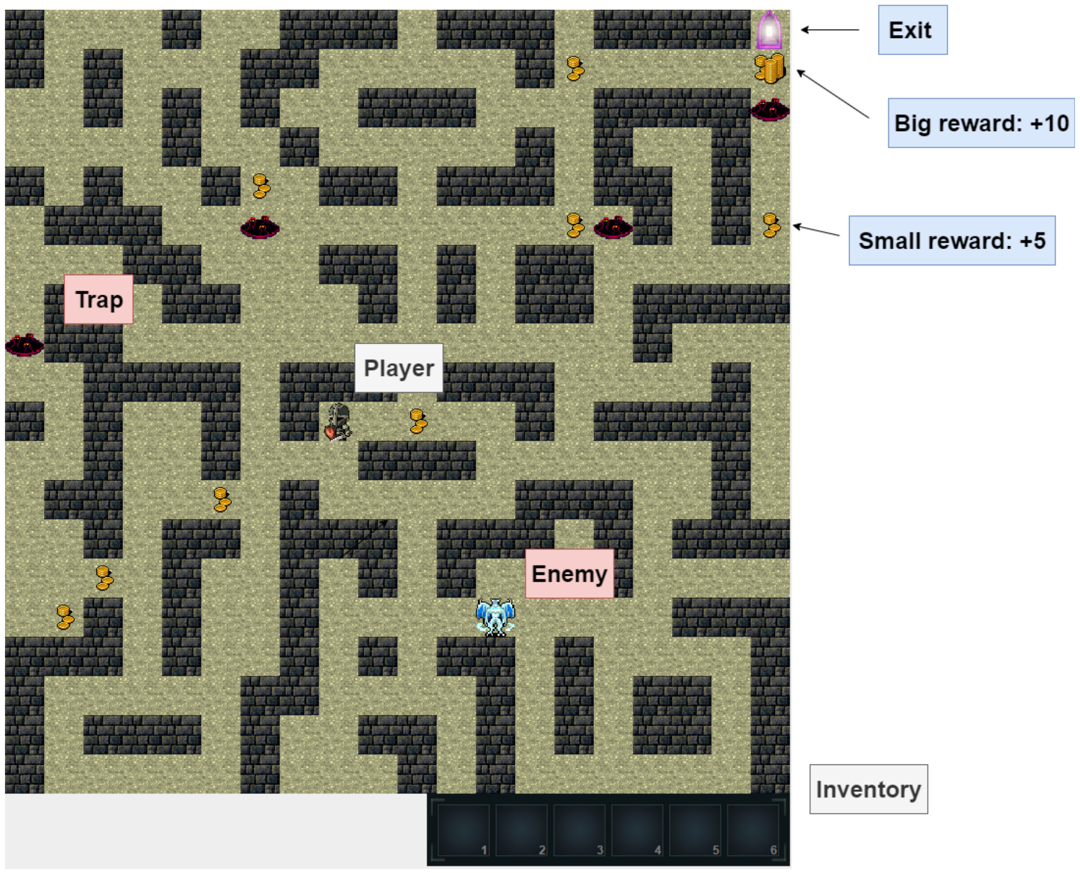
Winning the game

- Get total 50 points by walking over coins, small: +5 points, large: +10 points
- Then walk to exit (diagram below)

Losing the game

- Waking on a trap tile reduces points by 5, less than 0 points will result in loss
- Avoid the enemy, touching the enemy will result in a loss





3. Additional strategy

Avoiding the enemy

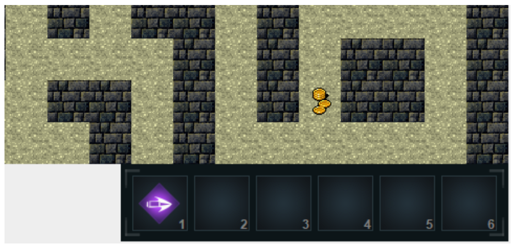
- The enemy uses an algorithm that finds the shortest path to the player, utilizing walls and loops around the map can allow the player to dodge the enemy and have an opening to move in a certain direction

Movement

- Many areas of the maps are blocked off with one exit, avoid those areas or areas with traps

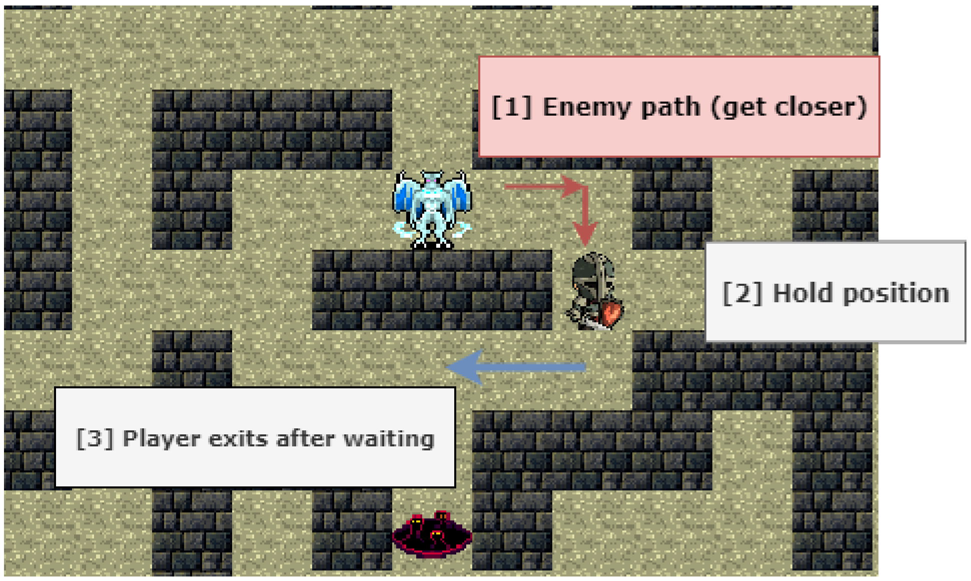
Inventory

- Use power ups at anytime after picking them up to further help you dodge the enemy or obtain points
- The dash powerup allows the player to jump 3 blocks either in the left or right direction (depending on where the player is looking)
- Example of an item picked up and available for use after pressing the (1) key:

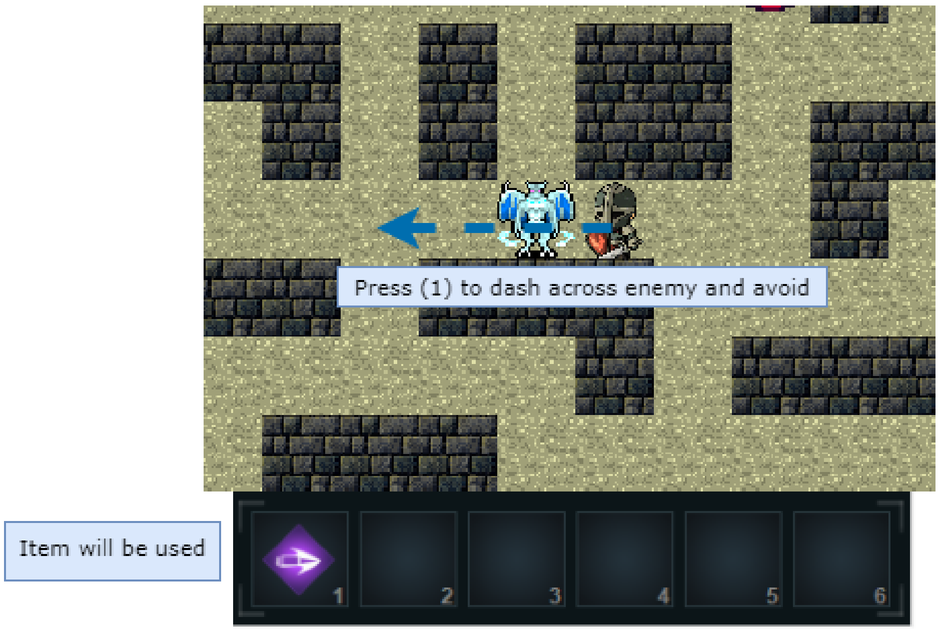


4. <u>Scenarios</u>

(1)



(2)


Press (1) to dash across enemy and avoid
Item will be used
1
2
3
4
5
6