**Story Design:**

In addition to the minimum requirement of phrase 1, we are going to add some features to the game. First, we add the main menu, in the main menu, the gamer can choose to start the game, difficult or exit game. Second, during the game running, we add the pause, restart and quit options to the game. These help the game become more flexible to the user.

More, to lead the game to become more interest, after the gamer collects all the rewards, they can go to the next level until they get the catch. By making more challenging of the game, we add the timer function, the gamer needs to collect all the rewards before the time runs out, otherwise, lose.

**Approach to build the game (Detail in UML Diagram)**

We are going to use a different kind of class to implement the game. The classes we are going to use are GameManager, Score, Board, Player, Rewards, Enemies, Trap, Barrier, Door, Input and output.

For the input class, is basically read the gamer input and return to the GameManager to implement the move action. For the output class, is used to show the game information on the screen to the gamer like the Board of the game, the position of player, enemy, trap and the player score.

For the Score class, it’s used to store the player score.

For the GameManager, it is the main class of the game. It implements the details of game start and end of the game.

For the board, it is also the main class of the game. It stores the position of different characters and objects of the game, like the position of the player and enemies, rewards and barrier. When the player wants to move to the new position. The Map class will check whether the position is barrier or rewards to deciding whether the player can move to the new position and the punishment/rewards the player got.

All the following class are subclass of the map.

1. Player:

It stores the main character location and controls its movement.

1. Reward：

It stores where the reward position and the rewardScore.E.g: When gamer got the reward, it will run the getRewardScore.

1. Enemies:

It stores enemies location. It has the move and trackthePlayer function to ensure that the enemies keep tracking the player and have the shortest distance with the player.

1. Door:

There are two door location in each level of the game, tell the user where start and where can go to the next level.

1. Trap:

It stores where the trap location and the damage of that trap to the player. When the player goes to the trap location. It will run randomDamage as the score deduct is random.

1. Barriers

It stores where the barrier location and to check whether the enemies and player can move to that location.