Exercise on Classes

We'll apply our knowledge of classes to a video game model in which we there are two types of characters.

Exercise 1:

Create a `PlayerCharacter` and a `NonPlayerCharacter` with a common ancestor `Character`. The characters are located in a 10x10 game field. All characters appear at a random location. Create the three classes, and make sure you can query where each character is.

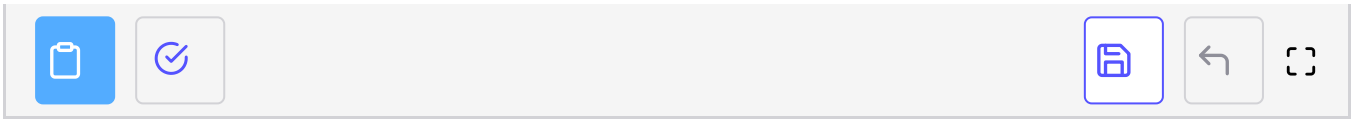
```
class Character {
    constructor( id, name, x, y ) {
       //Write your code here
  }

  get position() {
    //Write your code here
  }
}

//Define Player Character and NonPlayerCharacter classes here

function createPlayer( id, name ) {
  //Write your code here
}

function createNonPlayer( id, name ) {
  //Write your code here
}
```



Explanation:

This exercise has many solutions. We just used one. For the sake of simplicity, we chose not to model the game field. We placed x and y inside the character objects as coordinates.
At this stage, there was no difference between player and non-player characters.
We still created them to match the requirements.

Exercise 2: Judge For Yourself

Each character has a direction (up, down, left, right).

Player characters initially go right, and their direction can be changed using the faceUp, faceDown, faceLeft, faceRight methods. Non-player characters move randomly. A move is automatically taken every 5 seconds in real time.

Right after the synchronized moves, each character console logs its position. The player character can only influence the direction he is facing. When a player meets a non-player character, the non-player character is eliminated from the game, and the player's score is increased by 1.

Explanation:

We modeled the direction of each character with the dx and dy variables, describing the change in coordinates during one step. For instance, if the character faces upwards, dx is 0, and dy is -1. The specification allows non-player characters to occupy the same position.

Influence the movement of the player by executing

```
player.faceUp()
player.faceDown()
player.faceLeft()
player.faceRight()
```

Feel free to play around with the game in the console. If you want to test that updating the score works, you have a 50% chance of catching a wumpus by executing the following sequence:

```
player.faceLeft();
player.x = 0;
player.y = 0;
npcArray[0].x = 0;
```

```
npcArray[0].y = 0;
```