4 Functions

4.1 Refactoring your code using higher-order functions

4.1.1 Syntax

To loop over an array you can use the forEach higher-order function instead of the for...of loop. In the example below, the array animals is looped through using the forEach function., and each element is stored in the variable “animal”. When the age of the animal is greater than the age of the animal currently stored in the variable result (containing the animal with the highest age untill now), the animal will be stored in the variable result. By doing this, we are sure that “result” will contain the oldest animal.

```
const getOldestAnimal = () => {
  let result = animals[0];

  animals.forEach((animal) => {
    if (animal.age > result.age) {
      result = animal;
    }
  });

  return result;
};
```

4.1.2 Exercise: Use forEach() instead of for...of loops

Use the higher-order forEach function to loop over your arrays.

Evaluation criteria: You should have updated in the file game.js

- ❑ The for..of loop is changed in a forEach in the functions printAllGames, getAverageRating, getHighestRating and printFavouriteGames

4.1.3 Syntax

The filter function is another example of a higher-order function. It filters the elements of an array with the defined filter. In the example below, the animals array is filtered on only those animals of which the name contains the given chars (capitals are not important, therefor we apply the toLowerCase function on the given chars and the name of the i-th animal to which we apply the filtering). When the whole array is done filtering, a new array with only the animals matching the given chars is the result.

```
const printAllAnimals = (chars) => {
  animals
    .filter((animal) => chars ?animal.name.toLowerCase()
       .includes(chars.toLowerCase()) : true)
    .forEach((animal) => addStatus(toString(animal)));
}
```

4.1.4 Exercise: Use the filter function for showing the favourite games.

Use the filter function to filter only the favourite games and showing only these on the screen.

Evaluation criteria: You should have updated in the file game.js

- ❑ The function printFavouriteGames should use the filter function. The forEach doesn't need to check anything anymore: it only prints the name of the games in the filtered array on the screen.
- ❑ You still should use the isFavourite function.

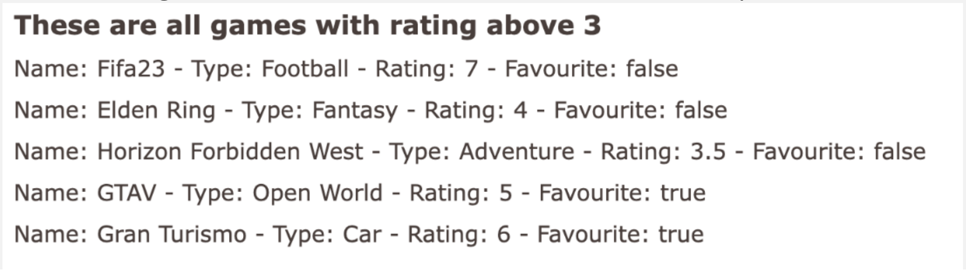
4.2 Adding new functionality

4.2.1 Exercise: Filter games on rating above 3

Implement a function that only shows the games with a rating higher than the given rating. Call this function with a rating of 3.

Evaluation criteria: You should have updated in the file game.js

- ❑ A new function printGamesRatingAbove
- ❑ With parameters games and rating
- ❑ With a combination of the filter function and forEach function
- ❑ With code to print it to the screen (as shown in the screenshot below)
- ❑ This function should be called at the bottom of the file, with the games array as the first parameter, and a rating of 3 as the second parameter. The resulting text when the function is called with these parameters:



```
These are all games with rating above 3
Name: Fifa23 - Type: Football - Rating: 7 - Favourite: false
Name: Elden Ring - Type: Fantasy - Rating: 4 - Favourite: false
Name: Horizon Forbidden West - Type: Adventure - Rating: 3.5 - Favourite: false
Name: GTAV - Type: Open World - Rating: 5 - Favourite: true
Name: Gran Turismo - Type: Car - Rating: 6 - Favourite: true
```

4.2.2 Syntax

A higher-order function accepts another function as parameter. We can use this to apply a custom filter to an array. For instance, in this example we will supply a custom filter to the animals array that we also supply as a parameter:

```
const filterAndPrintAnimals = (animals, customFilter) =>
{
  animals
    .filter(customFilter)
    .forEach((animal) => addStatus(toString(animal)));
}

filterAndAnimals(animals, (animal) =>
animal.name.toLowerCase().includes(“al”));
```

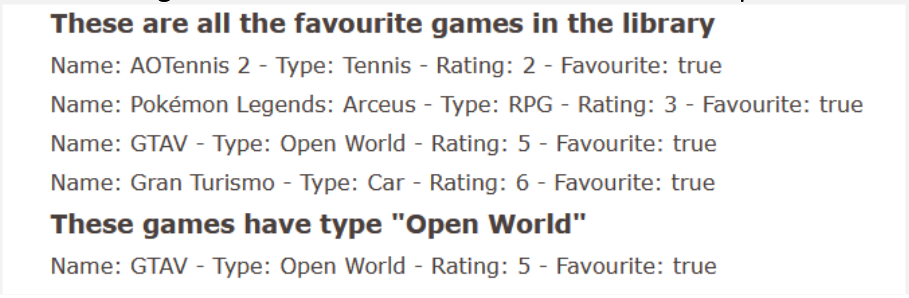
This will print all the animals that have the characters “al” in their name.

4.2.3 Exercise: Create a function that applies a custom filter

Implement a function that accepts a custom filter as a parameter and applies this custom filter to the supplied games. Call this function 2 times, once to print all favourite games, and once to print all games of type “Open World”. Do not use the existing isFavourite function, but write the custom filter inline.

Evaluation criteria: You should have updated in the file game.js

- ❑ A new function filterAndPrintGames
- ❑ With parameters games and customFilter
- ❑ Which filters the supplied games array using the supplied custom filter, and a forEach function that prints the filtered games.
- ❑ With code to print it to the screen (as shown in the screenshot below)
- ❑ This function should be called twice at the bottom of the file
- ❑ Once to filter all favourite
- ❑ Once to filter all games that have the type “Open World”.
- ❑ The resulting text when the function is called with these parameters:



```
These are all the favourite games in the library
Name: AOTennis 2 - Type: Tennis - Rating: 2 - Favourite: true
Name: Pokémon Legends: Arceus - Type: RPG - Rating: 3 - Favourite: true
Name: GTAV - Type: Open World - Rating: 5 - Favourite: true
Name: Gran Turismo - Type: Car - Rating: 6 - Favourite: true
These games have type "Open World"
Name: GTAV - Type: Open World - Rating: 5 - Favourite: true
```

4.2.4 Syntax

Another higher-order function that is built into arrays is the “map” function. This function will apply a transformation to each element, and return a new array with the result.

We can use this to first call the “toString” function on all our elements, and call addStatus on every element of the resulting array:

```
const printAllAnimals = (animals) => {
  animals
    .map(toString)
    .forEach(addStatus);
}
```

4.2.5 Exercise: Update printAllGames to use the map function

Update the printAllGames function to first transform every element with the toString function, and call addStatus on each element of the resulting array.

Evaluation criteria: You should have updated in the file game.js

- ❑ Updated the printAllGames function
- ❑ Use a combination of map function and the forEach function
- ❑ The map function will first call the toString function
- ❑ The forEach function will then call addStatus on every element of the “mapped” array.