Lab 7. JavaScript Fundamentals. Data structures, Modern Operators and Strings.

# Task 1.

We're building a football betting app (soccer for my American friends)!

Suppose we get data from a web service about a certain game ('game' variable onnext page). In this challenge we're gonna work with that data.

# Your tasks:

1. Create one player array for each team (variables 'players1' and

'players2')

1. The first player in any player array is the goalkeeper and the others are field players. For Bayern Munich (team 1) create one variable ('gk') with the goalkeeper's name, and one array ('fieldPlayers') with all the remaining 10field players
2. Create an array 'allPlayers' containing all players of both teams (22 players)
3. During the game, Bayern Munich (team 1) used 3 substitute players. So create anew array ('players1Final') containing all the original team1 players plus *'Thiago'*, *'Coutinho'* and *'Perisic'*
4. Based on the game.odds object, create one variable for each odd (called

'team1', 'draw' and 'team2')

1. Write a function ('printGoals') that receives an arbitrary number of player names (**not** an array) and prints each of them to the console, along with the number of goals that were scored in total (number of player names passed in)
2. The team with the lower odd is more likely to win. Print to the console which team is more likely to win, **without** using an if/else statement or the ternaryoperator.

**Test data for 6.:** First, use players *'Davies'*, *'Muller'*, *'Lewandowski'* and *'Kimmich'*.Then, call the function again with players from game.scored

  let [players1, players2] = game.players;

    let gk = players1[0];

    let fieldPlayers = players1.slice(1,11);

    let allPlayers = players1.concat(players2);

    let players1Final = players1.concat('Thiago', 'Coutinho', 'Perisic')

let {team1, x:draw, team2} = game.odds;

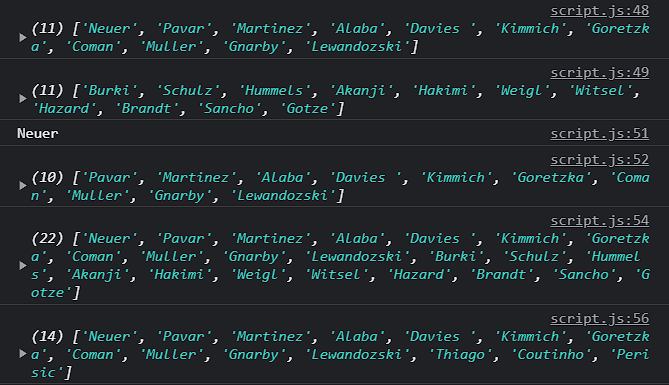
Created 2 arrays for 2 commands.

Created a gk variable for the goalie and used the array index zero.

Created an array of fieldPlayers with combat players using slice().

player1Final added the previously created array + added 3 spare players.

Result



 let scored1 = ['Davies', 'Muller', 'Lewandowski', 'Kimmich'];

    function printGoals(array){

    for(let i = 0; i < array.length; i++){

        let temp = 0;

        for(let j = 0; j < array.length; j++){

            if(array[i] == array[j]){

                temp++;

            }

        }

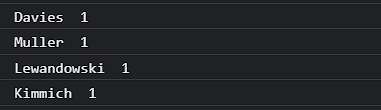
        console.log(`${array[i]}  ${temp}`);

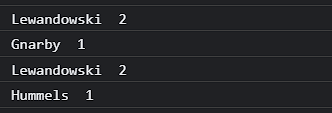
    }

}

printGoals(scored1);

With the help of a double loop, I check how many elements are repeated and display them.





# Task 2.

Let's continue with our football betting app! Keep using the 'game' variable from before.

# Your tasks:

1. Loop over the game.scored array and print each player name to the console, along with the goal number (Example: *"Goal 1: Lewandowski"*)
2. Use a loop to calculate the average odd and log it to the console (We already studied how to calculate averages, you can go check if you don't remember)
3. Print the 3 odds to the console, but in a nice formatted way, exactly like this:

*Odd of victory Bayern Munich: 1.33Odd of draw: 3.25*

*Odd of victory Borrussia Dortmund: 6.5*

Get the team names directly from the game object, don't hardcode them (except for *"draw"*). **Hint:** Note how the odds and the game objects have thesame property names



😉

1. **Bonus:** Create an object called 'scorers' which contains the names of the players who scored as properties, and the number of goals as the value. In this game, it will look like this:

{

}

Gnarby: 1,

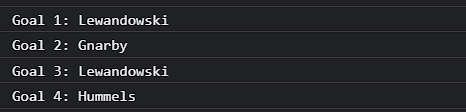
Hummels: 1,

Lewandowski: 2

for(let i = 0; i < game.scored.length; i++){

    console.log(`Goal ${i+1}: ${game.scored[i]}`);

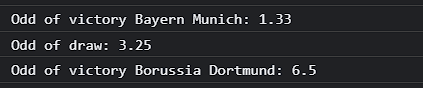
}



console.log(`Odd of victory ${game.team1}: ${game.odds.team1}`);

console.log(`Odd of draw: ${game.odds.x}`);

console.log(`Odd of victory ${game.team2}: ${game.odds.team2}`);



With the help of cycles, I output information + some elements from the object to the console.

# Task 3.

Let's continue with our football betting app! This time, we have a map called 'gameEvents' (see below) with a log of the events that happened during the game. The values are the events themselves, and the keys are the minutes in whicheach event happened (a football game has 90 minutes plus some extra time).

# Your tasks:

1. Create an array 'events' of the different game events that happened (no duplicates)
2. After the game has finished, is was found that the yellow card from minute 64was

unfair. So remove this event from the game events log.

1. Compute and log the following string to the console: *"An event happened, on average, every 9 minutes"* (keep in mind that a game has 90 minutes)
2. Loop over 'gameEvents' and log each element to the console, marking whether it's in the first half or second half (after 45 min) of the game, like this:

*[FIRST HALF] 17* ⚽ *GOAL*



*:*

const gameEvents = new Map(

    [

    [17, '⚽ GOAL'],

    [36, '🔁 Substitution'],

    [47, '⚽ GOAL'],

    [61, '🔁 Substitution'],

    [64,'🔶 Yellow card'],

    [69,'🔴 Red card'],

    [70,'🔁 Substitution'],

    [72, '🔁 Substitution'],

    [76, '⚽ GOAL'],

    [80, '⚽ GOAL'],

    [92, '🔶 Yellow card'],

    ]);

let events = [...new Set(gameEvents.values())];

console.log(events);

gameEvents.delete(64);

console.log(gameEvents);

let avgEvents = 90/gameEvents.size;

console.log(`An event happened, on average, every ${avgEvents} minutes`);

for(let [key, event] of gameEvents){

   console.log(key <= 45 ? '[FIRST HALF]' + event: '[SECOND HALF]' + event);

}

In this task, we are given a Map , using the methods I copied non-duplicate elements of the Map.

Then, the element with the key 64 was removed from the main Map, since this event was unfair.

I displayed a line where I calculated the average time of the event. For this, I divided 90 minutes by the size of the Map.

Using a loop and a ternary operator, I display a line where it is determined in the first half there was an event or in the second half of the match.

Result:



# Task 4.

Write a program that receives a list of variable names written in underscore\_caseand convert them to camelCase.

The input will come from a textarea inserted into the DOM (see code below to insert the elements), and conversion will happen when the button is pressed.

# Test data (pasted to textarea, including spaces):

underscore\_case first\_name

Some\_Variable calculate\_AGE

delayed\_departure

# Should produce this output (5 separate console.log outputs):

underscoreCase

firstName someVariable calculateAge delayedDeparture

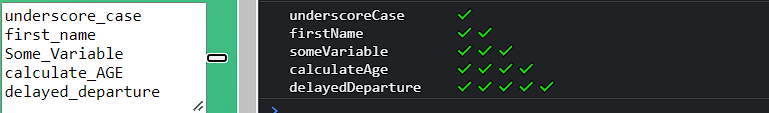


# Hints:

* Remember which character defines a new line in the textarea
* The solution only needs to work for a variable made out of 2 words, like a\_b
* Start without worrying about the  . Tackle that only after you have the variable name conversion working
* This challenge is difficult on purpose, so start watching the solution in case you're stuck. Then pause and continue!
* document.body.append(document.createElement('textarea'));
* document.body.append(document.createElement('button'));
* const text = document.querySelector('textarea');
* const button = document.querySelector('button');
* button.addEventListener('click', function() {
* let words = text.value.toLowerCase().split('\n');
* words = words.map(function (sp) {
* return sp.trim();
* });
* console.log(words);
* const fixedNames = words
* .filter(word => word.includes('\_'))
* .map((word, i) => camelCase(word).padEnd(20, ' ') + '✔️'.repeat(i + 1));
* console.log(fixedNames.join('\n'));
* });
* function camelCase(word){
* halves = word.split('\_');
* halves[1] = halves[1].slice(0, 1).toUpperCase() + halves[1].slice(1);
* return halves.join('');
* }

In this task, we have an arena for entering text and a button that takes the text that has been entered. Further, using the toLowerCase method, I make all the text lower case. We also remove all unnecessary spaces with trim(). We split the word into two words after the character "\_" , remove this character using split('\_') . split the two words into letters then make the first letter of the second word uppercase with the toUpperCase() method. We combine all letters and words into one word join('').

Result:

****