Rock, paper, scissors

For this assignment, you will create the "Rock, Paper, Scissors" game in the browser using Javascript. You need to make changes to the Javascript file, **and answer the questions in the comments**.

PLEASE MAKE SURE YOU READ ALL THE INSTRUCTIONS BEFORE YOU START WORKING ON THE CODE.

How the game works

- The player opens the HTML document in the browser.
- The HTML page displays 3 images:
  - rock
  - paper
  - scissors
- The player clicks one of the images.
- The clicked / selected image now displays with a thick 10 pixels black border around it.
- The "computer" and picks randomly one of the following:
  - rock
  - paper
  - scissors
- The image matching the choice of the computer is displayed in the page (see ID `computer_choice`).
- The result of the round is displayed in the browser:
  - if the player wins, display `You won!` in the `h1` element with ID `result`
  - if the computer wins, display `You lost!` in the `h1` element with ID `result`
  - if the player and the computer selected the same image, display `It's a tie!` in the `h1` element with ID `result`
- The `Play again` button is displayed.
- When the player clicks the button:
  - the player images are reset (no borders around any of them)
  - the computer choice is hidden
  - the `Play again` button is hidden
  - the user can play the game again normally

Hints

- The HTML document is already provided, and complete.
- The CSS file is already provided, and complete.
- **YOU CAN NOT MAKE CHANGES TO THE HTML OR THE CSS FILE. YOU WILL NOT SUBMIT THEM (ONLY THE JAVASCRIPT).**
- You can generate a random number with `Math.random()`. It returns a floating point number between 0 and 1.
  - You can multiply that number and convert the result to an integer (for example, between 1 and 3!).
  - You can use that number and compare it to fixed values to obtain a "random" element:

        - There is (approx.) a 33% chance that the result of `Math.random()` falls between `0` and `0.33`
        - There is (approx.) a 33% chance that the result of `Math.random()` falls between `0.33` and `0.66`
        - There is (approx.) a 33% chance that the result of `Math.random()` falls between `0.66` and `1`
    - Any of these techniques will allow you to "make a random choice" for the computer.
- After the computer made a choice, the player can still click on images, and sort of keep playing the game in a broken way. It is fine if that happens on your page, but you get bonus marks if it does not!
- To put it in another way: the player is not expected to click on an another image before clicking on the `Play again` button first.

Requirements / steps

Write the code that "selects" the player image

In the Javascript file, add the code that highlights an image with a thick border when it is clicked.

> 💡 **Look at the CSS file.**

> 💡 Maybe create a function `handleClick` that does the job?

> 💡 You can add classes to an HTML element by using: `element.classList.add("i-love-css")`

> 💡 **Look at the CSS file.**

Write the `play()` function

This function is the heart of the game. You must complete it and use it in your code later.

The function must:

- make a random choice for the computer between `"rock"`, `"paper"`, and `"scissors"`
- display the relevant image on the page in the computer choice section
    - you can change the display style of the image in Javascript
    - 💡 **Look at the HTML file**
    - The computer images have IDs.
    - 💡 **Look at the HTML file**
    - The image chosen by the computer does not have a border or a class applied to it
- get the player selection from the DOM
    - your selected image has a class
    - but you know there should be only one...
    - that looks like a good use case for `querySelector`! 💡
    - The player images have very descriptive IDs! You should use them.
- compare the player and the computer choices, and **RETURN** one of the following **INTEGERS**:
    - `0` if the game is tied
    - `1` if the player won

- `-1` if the player lost

⚠️ This function does not display the computer choice. It only returns a number and does not make any changes to the DOM.

> 💡 you can write the function in the Javascript file, and call it when you want from the browser Javascript console. Use `console.log` to help with debugging.

Reminder

- `rock` wins over `scissors` (`1`)
- `scissors` wins over `paper` (`1`)
- `paper` wins over `rock` (`1`)
- `rock` loses to `paper` (`-1`)
- `paper` loses to `scissors` (`-1`)
- `scissors` loses to `rock` (`-1`)

Add to the previous code

- Add to the existing code (user clicks => an image is selected with a border) so that the `play()` function is also called whenever an image is clicked.
- Using the return value from that function, display the relevant status message (win/lost/tie).
- Display the play again button.

Add code for the play again button

- Add to the existing code, so that clicking on the play again button:
  - resets all player images
  - hides any "computer" image
  - resets the "result" text to nothing
  - hides the play again button

Answer the questions (see comments in the Javascript file)

Submission

You must submit your Javascript file (and only your Javascript file) to the Learning Hub. When grading, I will reuse the HTML and CSS provided to run your code.

YOU CAN ONLY SUBMIT ONCE. ANY SUBMISSION IS FINAL!

Grading

Item	**Marks**
Click on the image displays a thick border	**1**
`play()` function	**4**
- random computer choice	1
- obtain player choice	1
- compare player and computer choices / return values	1
- code quality and conciseness	1
Click on the image plays the game and displays the result	**2**
Play again button	**2**
- reset game (player and computer images, play again button)	
General code quality and style, includes:	**2**
- code formatting and readability	
- variable names	
- no duplicate code (use a function instead)	
Questions in Javascript comments	**3**
TOTAL	**14**