**Requirements**

The result of your work should be packed to folder with \*.js files, and inner folder structure should be exactly as follows:

homework\_10/

└ task1.js

└ task2.js

Code has to be tested and well-formatted.

The folder must be loaded into the Github repository 'front-end-lab-8' in the master branch.

**Helpful links**

<http://learn.javascript.ru/functions-closures>

<https://css-tricks.com/the-difference-between-throttling-and-debouncing/>

**Task 1**

Create a function which takes two arguments where first argument is callback function and the second one delay is ms. This function should returns debounced function that limits the rate at which a function can fire.

function callback() { ... }

let debouncedCallback = debounce(callback, 1000);

debouncedCallback().

debouncedCallback().

Subsequent calls to the debounced function return the result of the last func invocation.

Please, take a look at example below

let iterator = 0;

function increaseIteratorBy1() {

iterator++;

printIteratorValue();

}

function printIteratorValue() {

console.log('Iterator value ', iterator);

}

var increaseIterator = debounce(increaseIteratorBy1, 1000);

increaseIterator();

increaseIterator();

increaseIterator();

...

increaseIterator(); // Should print 'Iterator value 2'

Tips: You should play around with timeoutId returned by setTimeout(...) and clear/reset this value to prevent setTimeout(...) to invoke callback function.

**Task 2**

Create a function ‘fighter’ which takes an object with properties of a fighter and returns interface with fighter methods.

The properties of fighter are following:

* name
* attack
* hp (health points)

The methods of fighter are following:

* **Get fighter name**
* **Block.** This function either returns true or false. If returned value is true the incoming attack is blocked.
* **Get fighter stats**. This function returns an object with properties of fighter (eg. name, attack, HP).
* **Get combat history**. Return an object witch contains information about wins and loses of the fighter.
* **Fight.** This function takes an enemy fighter as parameter (which is the defender in the current situation). If incoming damage of fighter1 (not a defender) is greater than health points of defender then fighter2 has lost and fighter1 has won. If fighter2 block damage of fighter1, HP remains the same. If fighter1 deal damage to fighter2, than fighter2 will lose HP equal to damage dealt. This function should return ‘true’ if the defender is damaged, otherwise false (e.g. if attack was blocked).

Test your implemented ‘fighter’ function by using the following code.

var fighter1 = fighter({name: 'John', attack: 100, hp: 100});

var fighter2 = fighter({name: 'Kayn', attack: 50, hp: 300});

var fighter3 = fighter({name: 'Bill', attack: 40, hp: 100});

fighter1.fight(fighter2); // If true, fighter 1 dealt damage to fighter 2, else fighter 2 blocked incoming damage

fighter1.fight(fighter3); // If true, fighter 1 dealt damage to fighter 3, else fighter 3 blocked incoming damage

/\*\*

\* Fighter John

\* - Combat stats: { wins: 1, loses: 0 }

\* - Properties: { name: 'John', attack: 100, hp: 100 }

\*/

showResult(fighter1);

/\*\* Fighter Kayn

\* - Combat stats: { wins: 0, loses: 0 }

\* - Properties: { name: 'Kayn', attack: 50, hp: 200 }

\*/

showResult(fighter2);

/\*\*

\* Fighter Bill

\* - Combat stats: { wins: 0, loses: 1 }

\* - Properties: { name: 'Bill', attack: 40, hp: 0 }

\*/

showResult(fighter3);

Please, note that **showResult** implemented already, you should not re-implement it.

Please, note that comments beside function shows result when fighter 1 dealt damage to other fighters.