INHERITANCE

**DEADLINE:** 27/03/2018

## FOLDER STRUCTURE

homework\_04/

├─ index.html

└─ scripts/

└─ script.js

## TASK

1. Create your own **assign** method (see **Code** section for invocation example).
2. Write your custom fighter (see **Code** section for invocation example).  
     
   Create two classes **Champion** and **Monster**.

Both classes have similar attributes such as **name**, **attack** and **hitpoints** (later referred as **total hitpoints**), which they accept on instantiation from a config object.

Both classes should have at least following methods:

* 1. getHitpoints - returns **current hitpoints**.
  2. setHitpoints - sets **current hitpoints**.
  3. getTotalHitpoints - returns **total hitpoints**.
  4. setTotalHitpoints - sets **total hitpoints**.
  5. getAttack - returns **attack** (amount of possible damage).
  6. setAttack - sets **attack** (amount of possible damage).
  7. fight - accepts other fighter(can’t accept itself) and reduces its **current hitpoints** by amount of **attack**. Also, make sure that the target is not dead.
  8. isAlive - returns boolean to indicate that **current** **hitpoints** are more than **0**.

Champion class should have at least following methods:

1. heal - restores **5** **hitpoints** (make sure **current hitpoints** value isn’t higher than **total**)
2. defence - will completely block **next** incoming damage and increase **total hitpoints** by **1**.

Monster class should have at least following methods:

1. enrage - **next two** attacks deal double damage
2. fury - reduces **total** and **current** **hitpoints** by 5 and increases **attack** by **2** (cannot execute this method if there is not enough **hitpoints**).

Champion trains in fight and gets **+1** to **attack** for killing monsters and other champions.

Monsters, on the other hand, feast on dead bodies and restore **25%** of their opponents **total hitpoints** and also add **10**% of their opponents **total** **hitpoints** to their own **total hitpoints** (floored to whole integers).

##### 

**EXTRA TASK** (This is not obligatory).

This homework is intended to practice reusability and inheritance, but if you like the idea of this fighter and want to show some creativity - you can create **fight.js** inside **script** folder, include it in **index.html**. Inside **fight.js** you can create a function that will take 2 fighters as parameters and will show fight process. How would you do it - is up to you. Again, this is not obligatory, but will be great to have.

## RESTRICTIONS

* Do not use ES6 classes
* Do not use any external libraries

## CODE

//Task 1

var defaults = { width: 100, height: 100 };

var options = { width: 150 };

var configs = assign({}, defaults, options); // -> {width: 150, height: 100}

//Task 2

var hunter = new Champion({name: ‘Rexxar’, attack: 10, hitpoints: 60});

var beast = new Monster({name: ‘King Krush’, attack: 8, hitpoints: 80});

hunter.fight(beast);

beast.getHitpoints(); // -> 70

beast.enrage();

hunter.fight(beast);

beast.getHitpoints(); // -> 60

beast.fight(hunter);

hunter.getHitpoints(); // -> 44

…

hunter.fight(beast);

beast.isAlive(); // -> false

hunter.getAttack(); // -> 11

hunter.getHitpoints(); // -> 44

hunter.heal();

hunter.getHitpoints(); // -> 49

## BEFORE SUBMIT

* Check if invocation examples are working with your code;
* Check file structure;
* Check code structure (remove unnecessary code/comments).

## USEFUL LINKS

* <http://eloquentjavascript.net/06_object.html>
* <https://developer.mozilla.org/en-US/docs/Web/JavaScript/Inheritance_and_the_prototype_chain>