codewars review code

study case

If you can't sleep, just count sheep!!

If you can't sleep, just count sheep!!

Instructions :

Given a non-negative integer, `3` for example, return a string with a murmur: `"1 sheep...2 sheep...3 sheep..."`. Input will always be valid, i.e. no negative integers.

Solution

```
var countSheep = function (num){
let str = ";
for(let i = 1; i <= num ; i++){
str+= ${i} sheep...;
}
return str;
}
```

Sample Tests

```
const chai = require("chai");
const assert = chai.assert;
chai.config.truncateThreshold=0;

describe("Fixed tests", () => {
it("Testing for fixed tests", () => {
assert.strictEqual(countSheep(0), "");
assert.strictEqual(countSheep(1), "1 sheep...");
assert.strictEqual(countSheep(2), "1 sheep...2 sheep...");
assert.strictEqual(countSheep(3), "1 sheep...2 sheep...3 sheep...");
});
});
```

studied:

increment

for

template literal

Grasshopper - Terminal game combat function

Instructions :

Create a combat function that takes the player's current health and the amount of damage recieved, and returns the player's new health. Health can't be less than **0.**

Solution

```
function combat(health, damage) {
return Math.max(health - damage, 0);
}
```

Sample Tests

```
const Test = require('@codewars/test-compat');

describe("The combat() function", function () {
it("should work for some example tests", function () {
Test.assertEquals(combat(100, 5), 95);
Test.assertEquals(combat(92, 8), 84);
Test.assertEquals(combat(20, 30), 0, "Health cannot go below 0");
});
});
```

studied:

module Math

arithmetic

Reversing Words in a String

Instructions :

You need to write a function that reverses the words in a given string. A word can also fit an empty string. If this is not clear enough, here are some examples:

As the input may have trailing spaces, you will also need to ignore unneccesary whitespace.

Example (**Input** --> **Output**)

```
"Hello World" --> "World Hello"
"Hi There." --> "There. Hi"
```

Happy coding!

Solution

```
function combat(health, damage) {
return Math.max(health - damage, 0);
}
```

Sample Tests

```
const Test = require('@codewars/test-compat');

describe("The combat() function", function () {
it("should work for some example tests", function () {
Test.assertEquals(combat(100, 5), 95);
Test.assertEquals(combat(92, 8), 84);
Test.assertEquals(combat(20, 30), 0, "Health cannot go below 0");
});
});
```

studied:

module Math

arithmetic

Super Duper Easy

Instructions :

Make a function that returns the value multiplied by 50 and increased by 6. If the value entered is a string it should return "Error".

Solution

```
function problem(x){
return typeof x === "number" ? x * 50 + 6 : "Error"
}
```

Sample Tests

```
const chai = require("chai");
const assert = chai.assert;
chai.config.truncateThreshold=0;

describe("Basic tests", () => {
it("Testing for fixed tests", () => {
assert.strictEqual(problem("hello"), "Error");
assert.strictEqual(problem(1), 56);
assert.strictEqual(problem(5), 256);
assert.strictEqual(problem(0), 6);
assert.strictEqual(problem(1.2), 66);
assert.strictEqual(problem(3), 156);
assert.strictEqual(problem("RyanIsCool"), "Error");
assert.strictEqual(problem(-3), -144);
assert.strictEqual(problem(""), "Error");
assert.strictEqual(problem(0.03), 7.5);
})
})
```

studied:

typeof

arithmetic

L1: Set Alarm

Instructions :

Write a function named setAlarm which receives two parameters. The first parameter, *employed*, is true whenever you are employed and the second parameter, *vacation* is true whenever you are on vacation.

The function should return true if you are employed and not on vacation (because these are the circumstances under which you need to set an alarm). It should return false otherwise. Examples:

Solution

```
function problem(x){
return typeof x === "number" ? x * 50 + 6 : "Error"
}
```

Sample Tests

```
const chai = require("chai");
const assert = chai.assert;
chai.config.truncateThreshold=0;
```

```
describe("Test Suite",()=>{
it("Fixed tests",()=>{
assert.strictEqual(setAlarm(true, true),  false,"Should be false.");
assert.strictEqual(setAlarm(false,true),  false, "Should be false.");
assert.strictEqual(setAlarm(true, false), true,"Should be true.");
});
});
```

studied:

typeof

arithmetic