You have a captain's log due before 2024-04-21 (in 1 day)! Log it now! (/captain_logs/5596018/edit)

0x12. JavaScript - Warm up

JavaScript

Weight: 1

Project over - took place from Jan 8, 2024 6:00 AM to Jan 9, 2024 6:00 AM

An auto review will be launched at the deadline

In a nutshell…

- **Auto QA review:** 113.0/113 mandatory & 29.0/29 optional
- **Altogether: 200.0%**
    - Mandatory: 100.0%
    - Optional: 100.0%
    - Calculation: 100.0% + (100.0% * 100.0%) == **200.0%**

Background Context

JavaScript is used for many things. Here, you will use JavaScript for 2 reasons:

- Scripting (same as we did with Python)
- Web front-end

For the moment, and for learning all basic concepts of this language, we will do some scripting. After, we will make our AirBnB project dynamic by using Javascript and JQuery.

Resources

Read or watch:

- Writing JavaScript Code (/rltoken/3HLjEesLsmyWfRUWnxgUGg)
- Variables (/rltoken/zgOWmcpVLZFEmFlmuwayyg)
- Data Types (/rltoken/VPd6JWaLrwOBzjAeXNAEqg)
- Operators (/rltoken/3HLjEesLsmyWfRUWnxgUGg)
- Operator Precedence (/rltoken/PHtcJJk30gBNmlFQ9R4RVg)
- Controlling Program Flow (/rltoken/tsreKcNh_KmTmLPHsfvJRw)
- Functions (/rltoken/e3EfHIxICdIncGBwwIDbXQ)
- Objects and Arrays (/rltoken/jg7IbvJpV2oLIKgqOAQH1g)
- Intrinsic Objects (/rltoken/jg7IbvJpV2oLIKgqOAQH1g)
- Module patterns (/rltoken/g-MgvO09Ur02RhM63gVyXw)
- var, let and const (/rltoken/gJi61GeJTRX0g-M0Rx-0Iw)
- JavaScript Tutorial (/rltoken/Y8hkOcy5jO22lQGyF6_NiA)
- Modern JS (/rltoken/NZawtiBjWUpiojnrtVywNw)

Learning Objectives

(/)

At the end of this project, you are expected to be able to explain to anyone (/rltoken/UFSXQvb7c_45LRd6SdzFTg), **without the help of Google**:

General

- Why JavaScript programming is amazing
- How to run a JavaScript script
- How to create variables and constants
- What are differences between `var`, `const` and `let`
- What are all the data types available in JavaScript
- How to use the `if`, `if ... else` statements
- How to use comments
- How to affect values to variables
- How to use `while` and `for` loops
- How to use `break` and `continue` statements
- What is a function and how do you use functions
- What does a function that does not use any `return` statement return
- Scope of variables
- What are the arithmetic operators and how to use them
- How to manipulate dictionary
- How to import a file

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

General

- Allowed editors: `vi`, `vim`, `emacs`
- All your files will be interpreted on Ubuntu 20.04 LTS using `node` (version 14.x)
- All your files should end with a new line
- The first line of all your files should be exactly `#!/usr/bin/node`
- A `README.md` file, at the root of the folder of the project, is mandatory
- Your code should be `semistandard` compliant (version 16.x.x). Rules of Standard (/rltoken/1T1yg1vOAChRN20Yyz8crw) + semicolons on top (/rltoken/35q5Pc6A6KWPyd3kGeRQFg). Also as reference: AirBnB style (/rltoken/ilo9MmB3u0utJZjZat-W3Q)
- All your files must be executable
- The length of your files will be tested using `wc`

More Info

Install Node 14

```
$ curl -sL https://deb.nodesource.com/setup_14.x | sudo -E bash -
$ sudo apt-get install -y nodejs
```

Install semi-standard

Documentation (/rltoken/35q5Pc6A6KWPyd3kGeRQFg)

```
$ sudo npm install semistandard --global
```

Quiz questions

Great! You've completed the quiz successfully! Keep going! (Show quiz)

Tasks

0. First constant, first print

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints "JavaScript is amazing":

- You must create a constant variable called `myVar` with the value "JavaScript is amazing"
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ ./0-javascript_is_amazing.js
JavaScript is amazing
guillaume@ubuntu:~/0x12$
guillaume@ubuntu:~/0x12$ semistandard ./0-javascript_is_amazing.js
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`

- File: `0-javascript_is_amazing.js`

(/)

Done! Check your code Get a sandbox QA Review

1. 3 languages

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints 3 lines:

- The first line: “C is fun”
- The second line: “Python is cool”
- The third line: “JavaScript is amazing”
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ ./1-multi_languages.js
C is fun
Python is cool
JavaScript is amazing
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `1-multi_languages.js`

Done! Check your code Get a sandbox QA Review

2. Arguments

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints a message depending of the number of arguments passed:

- If no arguments are passed to the script, print “No argument”
- If only one argument is passed to the script, print “Argument found”
- Otherwise, print “Arguments found”
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

Reference: process.argv (/rltoken/5kTYi3rxb6KM1_pivm-tXg)

```
guillaume@ubuntu:~/0x12$ ./2-arguments.js
No argument
guillaume@ubuntu:~/0x12$ ./2-arguments.js Best
Argument found
guillaume@ubuntu:~/0x12$ ./2-arguments.js Best School
Arguments found
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `2-arguments.js`

☑ Done! Check your code >_ Get a sandbox QA Review

3. Value of my argument

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints the first argument passed to it:

- If no arguments are passed to the script, print “No argument”
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`
- You are not allowed to use `length`

```
guillaume@ubuntu:~/0x12$ ./3-value_argument.js
No argument
guillaume@ubuntu:~/0x12$ ./3-value_argument.js School
School
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `3-value_argument.js`

☑ Done! Check your code >_ Get a sandbox QA Review

4. Create a sentence

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints two arguments passed to it, in the following format: “ is ”

- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ ./4-concat.js c cool
c is cool
guillaume@ubuntu:~/0x12$ ./4-concat.js c
c is undefined
guillaume@ubuntu:~/0x12$ ./4-concat.js
undefined is undefined
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `4-concat.js`

Done! Check your code Get a sandbox QA Review

5. An Integer

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints `My number: <first argument converted in integer>` if the first argument can be converted to an integer:

- If the argument can’t be converted to an integer, print “Not a number”
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`
- You are not allowed to use `try/catch`

```
guillaume@ubuntu:~/0x12$ ./5-to_integer.js
Not a number
guillaume@ubuntu:~/0x12$ ./5-to_integer.js 89
My number: 89
guillaume@ubuntu:~/0x12$ ./5-to_integer.js "89"
My number: 89
guillaume@ubuntu:~/0x12$ ./5-to_integer.js 89.89
My number: 89
guillaume@ubuntu:~/0x12$ ./5-to_integer.js School
Not a number
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `5-to_integer.js`

Done! Check your code Get a sandbox QA Review

6. Loop to languages

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints 3 lines: (like `1-multi_languages.js`) but by using an array of string and a loop

- The first line: “C is fun”
- The second line: “Python is cool”
- The third line: “JavaScript is amazing”
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`
- You are not allowed to use any `if/else` statement
- You can use only one `console.log`
- You must use a loop (`while` , `for` , etc.)

```
guillaume@ubuntu:~/0x12$ ./6-multi_languages_loop.js
C is fun
Python is cool
JavaScript is amazing
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`

- File: `6-multi_languages_loop.js`

(/)

Done! Check your code Get a sandbox QA Review

7. I love C

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints `x` times “C is fun”

- Where `x` is the first argument of the script
- If the first argument can’t be converted to an integer, print “Missing number of occurrences”
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`
- You can use only two `console.log`
- You must use a loop (`while` , `for` , etc.)

```
guillaume@ubuntu:~/0x12$ ./7-multi_c.js 2
C is fun
C is fun
guillaume@ubuntu:~/0x12$ ./7-multi_c.js 5
C is fun
C is fun
C is fun
C is fun
C is fun
guillaume@ubuntu:~/0x12$ ./7-multi_c.js
Missing number of occurrences
guillaume@ubuntu:~/0x12$ ./7-multi_c.js -3
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `7-multi_c.js`

Done! Check your code Get a sandbox QA Review

8. Square

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints a square
(/)

- The first argument is the size of the square
- If the first argument can't be converted to an integer, print "Missing size"
- You must use the character `X` to print the square
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`
- You must use a loop (`while` , `for` , etc.)

```
guillaume@ubuntu:~/0x12$ ./8-square.js
Missing size
guillaume@ubuntu:~/0x12$ ./8-square.js School
Missing size
guillaume@ubuntu:~/0x12$ ./8-square.js 2
XX
XX
guillaume@ubuntu:~/0x12$ ./8-square.js 6
XXXXXX
XXXXXX
XXXXXX
XXXXXX
XXXXXX
XXXXXX
guillaume@ubuntu:~/0x12$ ./8-square.js -3
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `8-square.js`

Done! Check your code Get a sandbox QA Review

9. Add

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that prints the addition of 2 integers

- The first argument is the first integer
- The second argument is the second integer
- You have to define a function with this prototype: `function add(a, b)`
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ ./9-add.js
NaN
guillaume@ubuntu:~/0x12$ ./9-add.js 1
NaN
guillaume@ubuntu:~/0x12$ ./9-add.js 1 7
8
guillaume@ubuntu:~/0x12$ ./9-add.js 13 89
102
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `9-add.js`

Done! Check your code Get a sandbox QA Review

10. Factorial

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that computes and prints a factorial

- The first argument is integer (argument can be cast as integer) used for computing the factorial
- Factorial of `NaN` is `1`
- You must do it recursively
- You must use a function
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ ./10-factorial.js
1
guillaume@ubuntu:~/0x12$ ./10-factorial.js 3
6
guillaume@ubuntu:~/0x12$ ./10-factorial.js 89
1.6507955160908452e+136
guillaume@ubuntu:~/0x12$ ./10-factorial.js 333
Infinity
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`

- File: `10-factorial.js`

(/)

☑ Done! Check your code >_ Get a sandbox QA Review

11. Second biggest!

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a script that searches the second biggest integer in the list of arguments.

- You can assume all arguments can be converted to integer
- If no argument passed, print `0`
- If the number of arguments is 1, print `0`
- You must use `console.log(...)` to print all output
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ ./11-second_biggest.js
0
guillaume@ubuntu:~/0x12$ ./11-second_biggest.js 1
0
guillaume@ubuntu:~/0x12$ ./11-second_biggest.js 4 2 5 3 0 -3
4
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `11-second_biggest.js`

☑ Done! Check your code >_ Get a sandbox QA Review

12. Object

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Update this script to replace the value `12` with `89`:

- You are not allowed to use `var`

(/)

```
guillaume@ubuntu:~/0x12$ cat 12-object.js
#!/usr/bin/node
const myObject = {
  type: 'object',
  value: 12
};
console.log(myObject);
/*
YOUR CODE HERE
*/
console.log(myObject);

guillaume@ubuntu:~/0x12$ ./12-object.js
{ type: 'object', value: 12 }
{ type: 'object', value: 89 }
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `12-object.js`

Done! Check your code Get a sandbox QA Review

13. Add file

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that returns the addition of 2 integers.

- The function must be visible from outside
- The name of the function must be `add`
- You are not allowed to use `var`

Tip (/rltoken/1k6VPdLchwtGubOfPyGL1Q)

```
guillaume@ubuntu:~/0x12$ cat 13-main.js
#!/usr/bin/node
const add = require('./13-add').add;
console.log(add(3, 5));
guillaume@ubuntu:~/0x12$ ./13-main.js
8
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `13-add.js`

Done! Check your code Get a sandbox QA Review

14. Const or not const

#advanced

Score: 100.0% (*Checks completed: 100.0%*)

Write a file that modifies the value of `myVar` to `333`

```
guillaume@ubuntu:~/0x12$ cat 100-main.js
#!/usr/bin/node
myVar = 89;
require('./100-let_me_const')
console.log(myVar);
guillaume@ubuntu:~/0x12$ ./100-main.js
333
guillaume@ubuntu:~/0x12$
```

Do you get it? Tweet! Post! Talk about it!

Hint: Scope

This exercise doesn't pass `semistandard` so don't worry about it.

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `100-let_me_const.js`

(✓) Done! Check your code >_ Get a sandbox QA Review

15. Call me Moby

#advanced

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that executes `x` times a function.

- The function must be visible from outside
- Prototype: `function (x, theFunction)`
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ cat 101-main.js
#!/usr/bin/node
const callMeMoby = require('./101-call_me_moby').callMeMoby;
callMeMoby(3, function () {
  console.log('C is fun');
});
guillaume@ubuntu:~/0x12$ ./101-main.js
C is fun
C is fun
C is fun
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `101-call_me_moby.js`

☐ Done? Check your code >_ Get a sandbox QA Review

16. Add me maybe

#advanced

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that increments and calls a function.

- The function must be visible from outside
- Prototype: `function (number, theFunction)`
- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ cat 102-main.js
#!/usr/bin/node
const addMeMaybe = require('./102-add_me_maybe').addMeMaybe;
addMeMaybe(4, function (nb) {
  console.log('New value: ' + nb);
});
guillaume@ubuntu:~/0x12$ ./102-main.js
New value: 5
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `102-add_me_maybe.js`

Done! Check your code Get a sandbox QA Review

17. Increment object

#advanced

Score: 100.0% (*Checks completed: 100.0%*)

Update this script by adding a new function `incr` that increments the integer `value` .

- You are not allowed to use `var`

```
guillaume@ubuntu:~/0x12$ cat 103-object_fct.js
#!/usr/bin/node
const myObject = {
  type: 'object',
  value: 12
};
console.log(myObject);
/*
YOUR CODE HERE
*/
myObject.incr();
console.log(myObject);
myObject.incr();
console.log(myObject);
myObject.incr();
console.log(myObject);

guillaume@ubuntu:~/0x12$ ./103-object_fct.js
{ type: 'object', value: 12 }
{ type: 'object', value: 13, incr: [Function] }
{ type: 'object', value: 14, incr: [Function] }
{ type: 'object', value: 15, incr: [Function] }
guillaume@ubuntu:~/0x12$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x12-javascript-warm_up`
- File: `103-object_fct.js`

Done! Check your code Get a sandbox QA Review