Functions

DEADLINE: 14/07/2019

FOLDER STRUCTURE

```
FL11_HW8/*
  └ task/
      └ FL11_HW8.docx
  └ homework/*
    └ js/*
      └ isBigger.js*
      └ isSmaller.js*
      └ getMin.js*
      └ isInteger.js*
      └ pipe.js*
      └ reverseNumber.js*
      └ formatTime.js*
    └ .eslintrc.js*
```

* - required

TASK

Task #1
Write a function - *isBigger*
It should accept two arguments and returns **true** if first one has **greater** value than second one or false otherwise.
Tip: no need for if/else clause nor ternary operator
For example:

```
isBigger(5, -1); // => true
```

Task #2
Write a function - *isSmaller*
It should accept two arguments and returns **true** if first one has **lesser** value than second one or false otherwise.
Tip: consider reusing *isBigger* function
For example:

```
isSmaller(5 -1); //=> false
```

Task #3
Write a function - *getMin*
It should accept **arbitrary** number of integer arguments and returns the one with the smallest value.
Tip: since **arguments** is like array, you can use simple iteration over it
and use arguments[i] to get the argument of a given index
For example:

```
getMin(3, 0, -3); //=> -3
```

Task #4
Write a function - *isInteger*
It should accept one number and returns true if it is integer number or false otherwise
For example:

```
isInteger(5); //=> true
isInteger(5.1); //=> false
```

Task #5
Write a function - *reverseNumber*
It should accept an integer and return it`s reversed version. Numbers should preserve their sign: i.e. a negative number should still be negative when reversed.
For example:

```
reverseNumber(123); //=> 321
reverseNumber(-456); //=> -654
reverseNumber(10000); //=> 1
```

Task #6
Write a function - *pipe*
It should accept a number as a first argument and arbitrary amount of functions after. The number should be passed to each function in sequence. The number passed to the next function is the returned result of previous function.
Tip: you need to use **arguments** to access all passed functions
For example:

```
function addOne(x) {
  return x + 1;
}

pipe(1, addOne); //=> 2
pipe(1, addOne, addOne); //=> 3
```

Task #7
Write a function - *formatTime*
It should accept a number as minutes as argument and return formatted string with calculated amount of days, hours and minutes.
Passed argument is always positive integer number.
Tip: no need to use Date object here
For example:

```
formatTime(120); //=> 0 day(s) 2 hour(s) 0 minute(s).
formatTime(59); //=> 0 day(s) 0 hour(s) 59 minute(s).
formatTime(3601); //=> 1 day(s) 0 hour(s) 1 minute(s).
```

BEFORE SUBMIT

- Verify that all functionality is implemented according to requirements;
- Format your code (remove redundant spaces, lines of code etc.);
- Validate code via eslint;
- Add comments if necessary, delete non-relevant comments;
- In order to use npm you should install nodejs (https://nodejs.org/);
- Install eslint to check your code (npm install -g eslint);
- open a terminal (or cmd);
- go to src folder;
- run eslint;
- Code should be without ‘errors’;

SUBMIT

- The folder should be uploaded to github repository '**FL11**' into **master** branch