# Homework: JavaScript Functions and Objects

This document defines the homework assignments from the [“JavaScript Basics“ Course @ Software University](http://softuni.bg/courses/javascript-basics/). Please submit as homework a single zip / rar / 7z archive holding the solutions (source code) of all below described problems.

## Last Digit of Number

Write a JavaScript function **lastDigitAsText(number)** that returns the **last digit** of given integer as an English word. Write a JS program **lastDigitOfNumber.js** that invokes your function with the sample input data below and prints the output at the console. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| 6 | Six |
| -55 | Five |
| 133 | Three |
| 14567 | Seven |

## N-th Digit of Number

Write a JavaScript function **findNthDigit(arr)** that accepts as a parameter an array of two numbers **num** and **n** and returns the **n-th** digit of given decimal number **num** counted from **right to left**. Return **undefined** when the number does not have n-th digit. Write a JS program **nthDigitOfNumber.js** that invokes your function with the sample input data below and prints the output at the console. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| [1, **6**] | 6 |
| [2, -**5**5] | 5 |
| [6, **9**23456] | 9 |
| [3, 145**1**.78] | 1 |
| [6, 888.88] | The number doesn’t have 6 digits |

## Number with Largest Sum of Digits

Write a JavaScript function **findLargestBySumOfDigits(nums)** that takes as an input a sequence of positive integer numbers and returns the element with the largest sum of its digits. The function should take a **variable number of arguments**. It should return **undefined** when 0 arguments are passed or when some of the arguments is not an integer number. Write a JS program **largestSumOfDigits.js** that invokes your function with the sample input data below and prints its output at the console. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5, 10, 15, 111 | 15 |
| 33, 44, -99, 0, 20 | -99 |
| 'hello' | undefined |
| 5, 3.3 | undefined |

## Bigger Than Neighbors

Write a JavaScript function **biggerThanNeighbors(index, arr)** that accept a **number** and an **integer** **array** as parameters. The function should return a Boolean number: whether the element at the given position in the array is **bigger** than its two neighbors (when such exist). It should return **undefined** when the index doesn't exist. Write a JS program that invokes the function with the sample data below and prints the result at the console following the samples below:

|  |  |
| --- | --- |
| **Input** | **Output** |
| 2, [1, 2, **3**, 3, 5] | not bigger |
| 2, [1, 2, **5**, 3, 4] | bigger |
| 5, [1, 2, 5, 3, 4] | invalid index |
| 0, [1, 2, 5, 3, 4] | only one neighbor |

## Reverse Every Word in a String

Write a JavaScript function **reverseWordsInString(str)** to reverse the characters of every word in the string but leaves the words in the **same order**. Words are considered to be sequences of characters separated by spaces. Write a JavaScript program **reverseWords.js** that prints on the console the output of the examples below:

|  |  |
| --- | --- |
| **Input** | **Output** |
| Hello, how are you. | ,olleH woh era .uoy |
| Life is pretty good, isn’t it? | efiL si ytterp ,doog t'nsi ?ti |

## Count Number of DIVs

Write a JavaScript function **countDivs(html)** to count the number of all DIVs in given HTML fragment passed as string. Write a JS program **countOfDivs.js** that invokes your function and prints the output at the console. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| <!DOCTYPE html>  <html>  <head lang="en">  <meta charset="UTF-8">  <title>index</title>  <script src="/yourScript.js" defer></script>  </head>  <body>  <div id="outerDiv">  <div  class="first">  <div><div>hello</div></div>  </div>  <div>hi<div></div></div>  <div>I am a div</div>  </div>  </body>  </html> | 7 |

## Find Youngest Person

Write a JavaScript function **findYoungestPerson(persons)** that accepts as parameter an **array** of **persons,** finds the youngest person and returns his **full name**. Write a JS program **youngestPerson.js** to execute your function for the below examples and print the result at the console.

|  |  |
| --- | --- |
| **Input** | **Output** |
| var persons = [  { firstname : 'George', lastname: 'Kolev', age: 32},  { firstname : 'Bay', lastname: 'Ivan', age: 81},  { firstname : 'Baba', lastname: 'Ginka', age: 40}]  findYoungestPerson(persons); | The youngest person is George Kolev |

## \* Sum of Two Huge Numbers

Write a JavaScript function **sumTwoHugeNumbers(value)** that accepts as parameter an array of the two numbers for summing. The input numbers are represented as **strings**. The result should be printed on the console. Example:

|  |  |
| --- | --- |
| **Input** | **Output** |
| ['155', '**65**'] | 220 |
| ['123456789', '123456789'] | 246913578 |
| ['887987345974539','4582796427862587'] | 5470783773837126 |
| ['347135713985789531798031509832579382573195807',  '817651358763158761358796358971685973163314321'] | 164787072748948293156827868804265355736510128 |

Hint: you might find in Internet a JavaScript library for working with big integers.

## \* Array Prototype Function

Write a JavaScript function **removeItem(value)** that accept as parameter a number or string. The function should **remove** all elements with the given value from an array. Attach the function to the **Array** type. You may need to read about **prototypes in JavaScript** and how to **attach** methods to object types. You should **return as a result** the modified array. Write a sample program to demonstrate that your function works correctly for the examples below:

|  |  |
| --- | --- |
| **Input** | **Output** |
| var arr = [1, 2, 1, 4, 1, 3, 4, 1, 111, 3, 2, 1, '1'];  arr.removeItem(1); | [2, 4, 3, 4, 111, 3, 2, '1'] |
| var arr = ['hi', 'bye', 'hello' ];  arr.removeItem('bye'); | ['hi', 'hello'] |

## \* Deep Copy of Object

Write a JavaScript function **clone(obj)** that accept as parameter an object of **reference type**. The function should **return** a **deep copy** of the object. Write a second function **compareObjects(obj, objCopy)** that compare the two objects by reference (==) and print on the console the output given below. Read in Internet about "**deep copy**"of an object and how to create it. Examples:

|  |  |
| --- | --- |
| **Input** | **Output** |
| var a = {name: 'Pesho', age: 21}  var b = clone(a); // a deep copy  compareObjects(a, b); | a == b --> false |
| var a = {name: 'Pesho', age: 21} ;  var b = a; // not a deep copy  compareObjects(a, b); | a == b --> true |

## \*\*\* Group Persons

Write a JavaScript function **group(persons)** that groups an array of **persons** by age, first or last name. Create a **Person** **constructor** to add every person in the person array. The **group(persons)** function must return an associative array, with **keys** – the groups (**age**, **firstName** and **lastName**) and **values** – arrays with persons in this group. Print on the console every entry of the returned associative array. Use function **overloading** (i.e. just one function).   
You may need to find how to use constructors. Examples:

|  |
| --- |
| **Input** |
| var persons = {  people.push(new Person("Scott", "Guthrie", 38));  people.push(new Person("Scott", "Johns", 36));  people.push(new Person("Scott", "Hanselman", 39));  people.push(new Person("Jesse", "Liberty", 57));  people.push(new Person("Jon", "Skeet", 38));  };  group(persons, 'firstname'); |
| **Output** |
| Group Scott : [Scott Guthrie(age 38), Scott Johns(age 36), Scott Hanselman(age 39)]  Group Jesse : [Jesse Liberty(age 57)]  Group Jon : [Jon Skeet(age 38)] // key : value |
| **Input** |
| group(persons, 'age'); |
| **Output** |
| Group 36 : [Scott Hanselman(age 36), Jon Skeet(age 36)]  Group 38 : [Scott Guthrie(age 38)]  Group 40 : [Scott Johns(age 40)]  Group 57 : [Jesse Liberty(age 57)] |
| **Input** |
| group(persons, 'lastName'); |
| **Output** |
| Group Guthrie : [Scott Guthrie(age 38)]  Group Johns : [Scott Johns(age 40),Jesse Johns(age 57)]  Group Hanselman : [Scott Hanselman(age 36)]  Group Skeet : [Jon Skeet(age 36)] |