# Exercises: JavaScript Syntax

Problems for exercises and homework for the ["JavaScript For Front-End Course @ SoftUni"](https://softuni.bg/trainings/1795/javascript-for-front-end-october-2017#lesson-7019).

## Multiply a Number by 2

You are given a number **N**. Create a JS function that **multiplies** the **number by 2** and prints the result. The input comes as an **array of strings**.

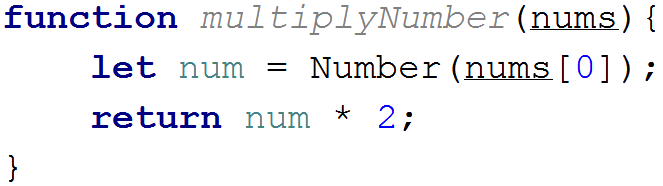
### Examples

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2 | 4 |  | 3 | 6 | 30 | 60 | 13 | 26 |

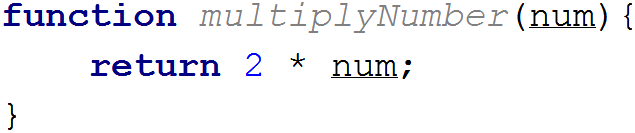
### Hints

* Note that the **input** comes as **array of strings**, so you should take the **first** element and **parse** it to **number**.
* Print the output to the console.

A sample solution might look like this:



Note that a simpler solution could also work, but is not recommended because it relies on automatic type conversion form array of strings to a number:



## Multiply Two Numbers

You are given a number X and a number Y. Create a JS function that multiplies X \* Y and prints the result. The input comes as array of strings.

### Examples

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2  3 | 6 |  | 13  13 | 169 | 1  2 | 2 | 0  50 | 0 |

## Multiply / Divide a Number by a Given Second Number

You are given a number N and a number X. Create a JS function that:

* Multiplies N \* X if X is greater than or equal to N
* Divides N / X if N is greater than X

The input comes as array of strings.

### Examples

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2  3 | 6 |  | 13  13 | 169 | 3  2 | 1.5 | 144  12 | 12 |

## Product of 3 Numbers

You are given a number X, Y and Z. Create a JS function that finds if X \* Y \* Z (the product) is negative or positive. Try to do this **WITHOUT** multiplying the 3 numbers.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 2  3  -1 | Negative |  | 5  4  3 | Positive | -3  -4  5 | Positive |

### Hint

* Count the **negative numbers**. If they are odd number, the result will be negative, otherwise 🡪 **positive**.
* Special case: one of the numbers is **0** 🡪 the **product** is **positive**.

## Print Numbers from 1 to N

You are given a number N. Create a JS function that loops through all the numbers from **1 to N** and prints them. N will always be positive.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 5 | 1  2  3  4  5 |  | 2 | 1  2 |

## Print Numbers from N to 1

You are given a number N. Create a JS function that loops through all the numbers from **N to 1** and prints them. N will always be positive.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 5 | 5  4  3  2  1 |  | 2 | 2  1 |

## Print Lines

You will be, continuously, given input lines, until you receive the command “Stop”. Print each of those lines at the moment you read them, until you reach the ending command. Do **NOT** print the ending command.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| Line 1  Line 2  Line 3  Stop | Line 1  Line 2  Line 3 |  | 3  6  5  4  Stop  10  12 | 3  6  5  4 |

## Print Numbers in Reversed Order

You will be given a few numbers as input. You need to print them in reversed order, each on a new line.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 10  15  20 | 20  15  10 |  | 5  5.5  24  -3 | -3  24  5.5  5 | 20  1  20  1  20 | 20  1  20  1  20 |

## Set Values to Indexes in an Array

You will be given **N** –an integer specifying the **length** of an **array**. Then you will start receiving an **index** and a **value**. For each received line you must **set** the **value** at the given **index** to the **given value**.

When you’ve processed all input data, **print** the array’s elements **each on a new line**.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| **3**  0 - 5  1 - 6  2 - 7 | 5  6  7 |  | **2**  0 - 5  0 - 6  0 - 7 | 7  0 | **5**  0 - 3  3 - -1  4 - 2 | 3  0  0  -1  2 |

## Add / Remove Elements

You will be given a sequence of **commands** (pairs of elements separated by a space): **command** and **argument**. You start by an empty array.

* The command “add {number}” appends the **number** to the array.
* The command “remove {index}”removes the element at the specified **index**. If the index is nonexistent, ignore that input line. When an element is deleted, all elements on the right from it, go one position left.

When you process all input data, **print the array’s elements** each on a separate line.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| add 3  add 5  add 7 | 3  5  7 |  | add 3  add 5  remove 1  add 2 | 3  2 | add 3  add 5  remove 2  remove 0  add 7 | 5  7 |

## Working with Key-Value Pairs

You will be given input lines, each holding **two elements** separated by a space. The first is the **key** and the second is the **value**.

Your task is to store the **value** for each **key**. If a key **already exists**, you need to **replace** the old value with the **new one**. At the last line of input, you will receive a **key**.

Print the **value** corresponding to that **key**. If there is no such, print “None”.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| key value  key eulav  test tset  key | eulav |  | 3 test  3 test1  4 test2  4 test3  4 test5  4 | test5 | 3 bla  3 alb  2 | None |

## Multiple Values for a Key

You will be given input lines, each holding **two elements** separated by a space: a **key** and **value**. You need to **store** the given **values** to the given **keys**. At the last line of the input you will receive a **key**.

Your task is to **print all the values** corresponding to that **key**. If there are no such, just print “**None**”.

### Examples

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| key value  key eulav  test tset  key | value  eulav |  | 3 test  3 test1  4 test2  4 test3  4 test5  4 | test2  test3  test5 | 3 bla  3 alb  2 | None |

## Storing Objects

You will be given input lines, each holding information about a **student**: **name**, **age** and **grade**. The data comes in the following format:

* “{name} -> {age} -> {grade}”

Store the information from the input lines into **JS objects**.

**Print** the objects in their order of appearance, in the format:

|  |
| --- |
| Name: {name}  Age: {age}  Grade: {grade} |

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Pesho -> 13 -> 6.00  Ivan -> 12 -> 5.57  Toni -> 13 -> 4.90 | Name: Pesho  Age: 13  Grade: 6.00  Name: Ivan  Age: 12  Grade: 5.57  Name: Toni  Age: 13  Grade: 4.90 |

## Parse JSON Objects

You will be given input lines (**text**) holding object data in **JSON format**. Use the JSON.parse(str) function to **parse** the data into **JavaScript objects**, and then **print** them as shown in the examples.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| {"name":"Gosho","age":10,"date":"19/06/2005"}  {"name":"Tosho","age":11,"date":"04/04/2005"} | Name: Gosho  Age: 10  Date: 19/06/2005  Name: Tosho  Age: 11  Date: 04/04/2005 |

## Turn Object into JSON String

You will be given input lines holding information about an object in the format “key -> value“. Create a **JS object** and save these keys and values in it.

After you’ve processed all the input data, print the **JSON** version of the object. Use the JSON.stringify(obj) function.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| name -> Angel  surname -> Georgiev  age -> 20  grade -> 6.00  date -> 23/05/1995  town -> Sofia | {"name":"Angel","surname":"Georgiev","age":20,"grade":6,"date":"19/05/1995","town":"Sofia"} |