# Lab: Syntax, Functions and Statements

Problems for in-class lab for the ["JavaScript Advanced" course @ SoftUni](https://softuni.bg/courses/js-advanced). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/2749/Syntax-Functions-and-Statements-Lab>

## Echo Function

Write a JS function that takes **one string parameter** and **prints** on two lines the **length** of the parameter and then the **unchanged parameter** itself.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 'Hello, JavaScript!' | 18  Hello, JavaScript! |
| 'strings are easy' | 16  strings are easy |

### Hints

* Write a function that receives a single **parameter**.
* Use the console.log function to print text on the console. Each call prints a new line automatically.
* The string’s length **property** is used to determine how many characters are in a given string

## String Length

Write a JS function that takes **three** **string arguments** as an input.

Calculate the **sum** of the **length** of the **strings** and the **average length** of the strings **rounded** **down** to the nearest integer.

The **input** comes as **three string arguments** passed to your function.

The **output** should be printed on the console on two lines.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 'chocolate', 'ice cream', 'cake' | 22  7 |
| 'pasta', '5', '22.3' | 10  3 |

### Hints

* Write a function that receives three string arguments.
* Declare two variables named sumLength and averageLength that will keep the mathematical results.
* Calculate the length of the strings using the length **property**.



* Calculate the sum of the three lengths. 
* Calculate the **average length** of the strings **rounded** **down** to the nearest integer. Use the Math.floor() function.



* Print the results on the console.



## Largest Number

Write a function that takes **three number arguments** as an input and find the **largest** of them. Print the following text on the console: '**The largest number is {number}.**'.

The **input** comes as **three number arguments** passed to your function.

The **output** should be printed to the console.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5, -3, 16 | The largest number is 16. |
| -3, -5, -22.5 | The largest number is -3. |

### Hints

* Write a function that receives three number arguments.
* Declare a variable named result that will keep the result.



* Make several checks to find out the largest of the three numbers. Start with num1.



* Do the same for the others.



* Print the result on the console.



## Circle Area

Write a function that takes **a single argument** as an input.

**Check the type** of the input argument. If it is a **number**, assume it is the radius of a circle and **calculate the circle area**. Print the **area** **rounded** to **two decimal places**.

If the argument type is **NOT** a number, print the following text on the console:   
'We can not calculate the circle area, because we receive a {type of argument}.'

The **input** comes as a **single argument** passed to your function.

The **output** should be printed on the console.

### Example

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5 | 78.54 |
| 'name' | We can not calculate the circle area, because we receive a string. |

### Hints

* Write a function that receives a single argument.
* Declare a variable named result that will keep your result.



* Check the type of the input argument with the typeof operator.



* If the type is equal to 'number', calculate the circle area and print it on the console rounded to two decimal places. To do this, use the method toFixed().  
  The Math.pow() function returns the base to the exponent power, that is, base exponent. You can find more information about the area [here](https://en.wikipedia.org/wiki/Circle):



* If the type is **NOT** a 'number', print the following text on the console:



## Math Operations

Write a JS function that takes **two** **numbers** and **a string** as an input.

The string may be one of the following: '**+**', '**-**', '**\***', '**/**', '**%**', '**\*\***'.

Print on the console the result of the mathematical **operation** between **both numbers** and the **operator** you receive as a string.

The **input** comes as **two numbers** and **a string argument** passed to your function.

The **output** should be printed on the console.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| 5, 6, '+' | 11 |
| 3, 5.5, '\*' | 16.5 |

### Hints

* Write a function which receives **three** arguments:



* Declare a variable named result that will keep your mathematical result.
* Write down switch command that will take the string from your input and depending on it, perform the mathematical logic between the two numbers.



* Print the result on the console.



## Sum of Numbers N…M

Write a JS function that takes two numbers **n and m** as an input and **prints the sum** of all numbers from **n** to **m**.

The **input** comes as **two string elements** that need to be **parsed** as numbers.

The **output** should return the **sum**.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| '1', '5' | 15 |
| '-8', 20' | 174 |

### Hints

* Write a function that receives two string arguments and parse them as numbers. Use Number(string) function to parse the input.



* Declare a variable named result that will keep the mathematical results.
* Write a for loop from num1 to num2 and for every turn of the cycle, until it’s completed, add the current value.



* Finally, return the result.



## Day of Week

Write a function that prints a number between 1 and 7 when a **day of the week** is passed to it as a string and an **error message** if the string is **not recognized**.

The **input** comes as a single string argument.

The **output** should be returned as a result.

### Examples

|  |  |
| --- | --- |
| **Input** | **Output** |
| Monday | 1 |
| Friday | 5 |
| Invalid | error |

## Square of Stars

Write a function that **prints a rectangle** made of **stars** with variablesize, depending on an input parameter. If there is **no parameter** specified, the rectangle should **always** be of **size 5**. Look at the examples to get an idea.

The **input** comes as a single **number** argument.

The **output** is a series of lines printed on the console, forming a rectangle of variable size.

### Examples

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |  | **Input** | **Output** |
| 1 | \* | 2 | \* \*  \* \* | 5 | \* \* \* \* \*  \* \* \* \* \*  \* \* \* \* \*  \* \* \* \* \*  \* \* \* \* \* |  | \* \* \* \* \*  \* \* \* \* \*  \* \* \* \* \*  \* \* \* \* \*  \* \* \* \* \* |

## Aggregate Elements

Write a program that performs different operations on an array of elements. Implement the following operations:

* **Sum(ai)** - calculates the sum all elements from the input array
* **Sum(1/ai)** - calculates the sum of the inverse values (1/ai) of all elements from the array
* **Concat(ai)** - concatenates the string representations of all elements from the array

The **input** comes as an array of number elements.

The **output** should be printed on the console on a new line for each of the operations.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| [1, 2, 3] | 6  1.8333  123 | [2, 4, 8, 16] | 30  0.9375  24816 |

**10. Fruit**

Write a function that calculates how much money you need to buy fruit. You will receive a **string** for the type of fruit you want to buy, **a number** for weight in grams and another **number** for the price per kilogram.

Print the following text on the console:

**'I need ${money} to buy {weight} kilograms {fruit}.'**

Print the weight and the money **rounded** to two decimal places.

The **input** comes as **three arguments** passed to your function.

The **output** should be printed on the console.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 'orange', 2500, 1.80 | I need $4.50 to buy 2.50 kilograms orange. |

|  |  |
| --- | --- |
| **Input** | **Output** |
| 'apple', 1563, 2.35 | I need $3.67 to buy 1.56 kilograms apple. |

## Greatest Common Divisor - GCD

Write a function that takes **two** **positive** **numbers** as input and compute the greatest common divisor.

The **input** comes **as two positive integer numbers**.

The **output** should be printed on the console.

**Examples**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 15, 5 | 5 |  | 2154, 458 | 2 |

## Same Numbers

Write a function that takes **an integer** **number** as an input and check if all the digits in a given number are the same or not.

Print on the console **true** if all numbers are same and **false** if not. On the next line print the **sum of all digits.**

The **input** comes as an integer number.

The **output** should be printed on the console.

**Examples**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 2222222 | true  14 |  | 1234 | false  10 |

## Time to Walk

Write a function that **calculates** how long it takes a student to get to university.   
The function takes **three numbers**:

* The **first** is the number of **steps** the student takes from their home to the university
* Тhe **second** number is the length of the student's footprint in **meters**
* Тhe **third** number is the student speed in **km/h**

Every 500 meters the student rests and takes a **1 minute break**.

Calculate how long the student walks from home to university and print on the console the result in the following format: **'**hours:minutes:seconds**'**.

The **input** comes as **three numbers**.

The **output** should be printed on the console.

**Examples**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 4000, 0.60, 5 | 00:32:48 |  | 2564, 0.70, 5.5 | 00:22:35 |

## Road Radar

Write a function that determines whether a driver is within the speed limit. You will receive the speed and the area. Each area has a different limit:

* On the **motorway** the limit is **130 km/h**
* On the **interstate** the limit is **90 km/h**
* In the **city** the limit is **50 km/h**
* Within a **residential** area the limit is **20 km/h**

If the driver is **within the limits**, there should be printed speed and the speed limit.

**`Driving {speed} km/h in a {speed limit} zone`**

If the driver is **over the limit**, however, your function should print the severity of the infraction and the difference in speeds.

**`The speed is {difference} km/h faster than the allowed speed of {speed limit} - {status}`**

For speeding up to **20** km/hover the limit, speeding should be printed

For speeding up to **40** km/h over the limit, excessive speeding should be printed

For anything else, reckless driving should be printed

The **input** comes as **2 string parameters**. The first element is the current speed (**number**), the second element is the area.

The **output** should be printed on the console.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 40, 'city' | Driving 40 km/h in a 50 zone |
| 21, 'residential' | The speed is 1 km/h faster than the allowed speed of 20 - speeding |
| 120, 'interstate' | The speed is 30 km/h faster than the allowed speed of 90 - excessive speeding |
| 200, 'motorway' | The speed is 70 km/h faster than the allowed speed of 130 - reckless driving |

## Cooking by Numbers

Write a program that receives 6 parameters which are a **number** and a **list** of five operations. Perform the operations **sequentially** by starting with the **input number** and using the result of every operation as starting point for the next one. Print the result of every operation in order. The operations can be one of the following:

* **chop** - divide the number by two
* **dice** - square root of number
* **spice** - add 1 to number
* **bake** - multiply number by 3
* **fillet** - subtract 20% from number

The **input** comes as **6 string elements**. The first element is the starting point and must be **parsed** to a number. The remaining 5 elements are the names of the operations to be performed.

The **output** should be printed on the console.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| '32', 'chop', 'chop', 'chop', 'chop', 'chop' | 16 8 4 2 1 |

|  |  |
| --- | --- |
| **Input** | **Output** |
| '9', 'dice', 'spice', 'chop', 'bake', 'fillet' | 3  4  2  6  4.8 |

## Validity Checker

Write a program that receives total of 4 parameters in the format **x1, y1, x2, y2.** Check if the distance between each point and the start of the cartesian coordinate system (0, 0) is **valid**. A distance between two points is considered **valid**, if it is an **integer value**.

In case a distance is valid, print"{x1, y1} to {x2, y2} is valid**"**

If the distance is invalid, print **"**{x1, y1} to {x2, y2} is invalid**"**

The order of comparisons should always be first **{x1, y1}** to **{0, 0}**, then **{x2, y2}** to **{0, 0}** and finally **{x1, y1}** to **{x2, y2}**.

The **input** consists of two points given as **4 numbers**.

For each comparison print either "{x1, y1} to {x2, y2} is valid**"** if the distance is valid, or **"**{x1, y1} to {x2, y2} is invalid**"** if it is invalid.

**Examples**

|  |  |
| --- | --- |
| **Input** | **Output** |
| 3, 0, 0, 4 | {3, 0} to {0, 0} is valid  {0, 4} to {0, 0} is valid  {3, 0} to {0, 4} is valid |
| 2, 1, 1, 1 | {2, 1} to {0, 0} is invalid  {1, 1} to {0, 0} is invalid  {2, 1} to {1, 1} is valid |

## \*Words Uppercase

Write a program that **extracts all words** from a passed in string and converts them to **upper case**. The extracted words in upper case must be printed on a single line separated by **", "**.

The **input** comes as a single string argument - the text to extract and convert words from.

The **output** should be a single line containing the converted string.

### Examples

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Input** | **Output** |  | **Input** | **Output** |
| 'Hi, how are you?' | HI, HOW, ARE, YOU | 'hello' | HELLO |