# Exercise: 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/2750/Exercise-Syntax-Functions-and-Statements>

1. **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.** |

1. **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** |

1. **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 the 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** |

1. **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** |

1. **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 a 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, the **status** should be speeding.

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

For anything else, **status** should be reckless driving.

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** |

1. **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 a 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 a number
* **spice** - add 1 to the 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** |

1. **Validity Checker**

Write a program that receives a 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 the 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** |

### Hints

* You may need to use a [Regular Expression](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Regular_Expressions) or alternatively check for all delimiters that can be found in a sentence (ex. ",", " ", "!", "?" and so on).