EXAM REVIEW

EXERCICE 1

WHAT YOUR PROGRAM SHALL DO
We want to check if the word is containing the latter that we want We enter on console: 1. A word 2. A letter We print: - Valid if the word contains that letter - Otherwise print Not Valid

Function name	toContains
Parameters	A word (string) A letter (string)
Return value	**Boolean**
Examples	toContains(“Rady”,“Y”) -> **True**

WARNING: The letter can be upper and lower case.

EXAMPLES	
CONSOLE	EXPLANATION
> Ronan > A >Valid	The word contains the letter **A** So, we return **True**
> RADY > d >Valid	The word contains the letter **d** So, we return **True**
> HIM > A >Not Valid	The word does not contain the letter **A** So, we return **False**
>Meng heang >X > Not Valid	The word does not contain the letter **X** So, we return **False**

EXERCICE 2

WHAT YOUR PROGRAM SHALL DO

We want to reverse a string.
For instance, the reversed string of "HELLO WORLD" is "DLROW OLLEH"

To perform this program, you MUST code and use this function:

Function name	toReverse
Parameters	A word (string)
Return value	**A new word** (string) which is the reversed of the given string.
Examples	toReverse("RADY") -> YDAR

Example:

INPUT
ABC123

OUTPUT
321CBA

INPUT
PNC

OUTPUT
CNP

EXERCICE 3

WHAT YOUR PROGRAM SHALL DO

First you will to implement the following function:

Function name	multiplyArray
Parameters	array
Return value	The multiplication of the number in array.
Examples	multiplyArray([2,3,4,5]) -> 120 Explanation: we access to each value in array and multiply it together. 2 * 3 * 4 * 5 = **120**

INPUT:

- Enter an array of integers to the console

***array is not given, you have to code it using eval(input())

OUTPUT:

- Print the number of multiplications of number in array

EXAMPLES	
CONSOLE	EXPLANATION
> [4,1,3] >12	The answer is 12 because: 4 * 1 * 3 = 12 So, we return 12
> [1, 2,3,5] >30	The answer is 30 because: 1 * 2 * 3 * 5 = 30 So, we return 30
> [5,4,3] >60	The answer is 60 because: 5 * 4 * 3 = 60 So, we return 60
> [] >0	Nothing to multiply. So, we return 0

EXERCICE 4

WHAT YOUR PROGRAM SHALL DO

First you need to implement the following function:

Function name	countChar
Parameters	A list of string (array) A letter
Return value	The count number of letter that we found in a given string
Examples	countChar (["RONAN","Rady"],"n") → 2 Explanation: we found 2 letters n in the word: **RONAN** = 2 **RADY** = 0
Warning	1, If we cannot find the letter in the given word, **we need to return 0** 2, The character that we found can be uppercase or lowercase.

We want to count the number of letters that exists in the list of words.

For instance:

We want to find how many characters "Z" exists in ["BBB", "CCZ", "XXZ"]

The answer is 2, because we can find 2 characters "Z" in this list of words.

EXAMPLES	
CONSOLE	EXPLANATION
countChar (["hello","hi","bye bye"],"H") >2	we need to find the letter that we give in list array. ["hello","hi","bye bye"] So, we return 2
countChar (["hello, world","pnc"],"Y") >0	we need to find the letter that we give in list array. ["hello, world","pnc"] We didn't find the letter Y in list array. So, we return 0

EXERCICE 5

WHAT YOUR PROGRAM SHALL DO
First you need to implement the following function:

Function name	Average
Parameters	A array of number
Return value	The average number of a range number in array
Examples	Average ([2,5]) → **3.5** Explanation: we start from **2** and we end at **5:** 2 + 3 + 4 + 5 = **14** 14/4 = **3.5**

Warning: if we have only one number in array, we need to start that value with 0.

EXAMPLES	
CONSOLE	EXPLANATION
Average ([8,16]) >12	we start from **8** and we end at **16:** 8 + 9 + 10 + 11 + 12 + 13 + 14 + 15 + 16 = **108** 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 = **9** 108 / **9** = **12** So, we return **12**
Average ([4,10]) >7	we start from **4** and we end at **10:** 4 + 5 + 6 + 7 + 8 + 9 + 10 = **49** 1 + 1 + 1 + 1 + 1 + 1 + 1 = **7** **49** / **7** = **7** So, we return **7**
Average ([4]) >2	we start from **0** and we end at **4:** 0 + 1 + 2 + 3 + 4 = **10** 1 + 1 + 1 + 1 + 1 = **5** **10** / **5** = **2** So, we return **2**