Python for Engineers 0509-1820, Semester 1 2019-2020

Homework 2

Loops and lists

General instructions

- Read the questions **carefully** and make sure your programs work according to the requirements.
- The homework needs to be done individually!
- Read the submission rules on the course web page. All the questions need to be submitted together in the file ex1_012345678.py attached to the homework, after changing the number 012345678 with your ID number (9 digits, including check digit).
- Final submission date: see course web page.
- Check your code: in order to ensure correctness of your programs and their robustness in the presence of faulty input, for each question run your program with a variety of different inputs, those that are given as examples in the question and additional ones of your choice (check that the output is correct and that the program does not crash).
- Because the checking of the homework is automated, **please make sure to respect the format of the input (including spaces).**
- How to write the solution: in this homework, you need to complete the code in the attached outline file.
- **You are not allowed to change the names of the variables that already appear in the attached outline file. You need to work with the variables already declared in the outline.** The code of each question must work and produce the correct answer for the input that will be given in the variables present in the outline (the variables that feature a question mark and need to be provided with an input, as shown in the example during the recitation). You are also allowed to add additional variables.
- **You may not erase the instructions present in the outline.**

General note: At this stage of the course, you can assume that your input is correct regarding its type—meaning, for example, that if you are told that a variable is a list of integers, you do not need to check this. Apart from this (for example, checking if the list can be empty or not), you cannot assume anything else regarding the input apart from what is said explicitly in the question.

Question 1

Given a list called *A* containing positive integers, and a positive integer called *a*, print the **index** of the first number in the list which is divisible by *a*. If there is no such number, print -1. You can assume that *a* is different from 0, but *A* could also be an empty list.

For example, if *a*=3 and the list is

A = [1,2,3,4,5]

You should print:

```
2
```

And, with the same list, if *a*=8, you should print

```
-1
```

Question 2

Given a list of strings (variables of type *str*) called *B*, find the average length of the strings in the list and print the number of strings whose length is strictly greater than the average length. Do this both with a *while* loop and with a *for* loop. You can assume that the list *B* contains at least one string.

For example, for the string

B = ['hello', 'world', 'course', 'python', 'day']

The average string length is 5.

The output of the program will be:

```
The number of strings longer than the average is: 2
The number of strings longer than the average is: 2
```

The two lines show the output of the *while* loop and the *for* loop, respectively

Question 3

Given a list of numbers called *C*, write a piece of code that prints the **sum** of the **product** of each pair of adjacent numbers in the list.

Special cases:

- For an empty list, print 0
- For a list containing a single number, print this number

For example, for the list:

C = [0, 1, 2, 3, 4]

Print:

20

Because: $0 \cdot 1 + 1 \cdot 2 + 2 \cdot 3 + 3 \cdot 4 = 20$

Question 4

Given a list *D* of positive integers, containing at least two elements, write code that prints a new list based on list *D* with the following additional requirement: the difference (in absolute value) between each pair of adjacent elements is **strictly** greater than all the previous differences. The new list will contain numbers appearing in the original list, and in the same order. In case a number appearing in *D* does not fulfill the above requirement regarding differences, this number will not appear in the new list.
For example, for the list:

D = [1, 2, 4, 6, 5, 9]

You should print:

[1, 2, 4, 9]

Explanation: the difference (in absolute value) between 4 and 2 is 2, and the difference between 6 and 4 is also 2, hence 6 should not appear in the new list. We then compute the difference between 5 and 4, which is also not greater than 2, hence 5 also does not appear in the new list. The difference between 9 and 4 is 5, which does fulfill the requirements, and hence 9 does appear in the new list.

For the list:

D = [1, 2, 4, 8]

You should print:

[1, 2, 4, 8]

Explanation: all the list fulfills the requirements, hence all the list is printed.

Additional example: for the list:

D = [1, 3, 0, 2]

You should print:

[1, 3, 0]

Explanation: all the list fulfills the requirement, except the last number, whose difference with its predecessor (i.e. the difference between 2 and 0) is smaller than the preceding difference (between 0 and 3).

Question 5

Write a program which, given a string *my_string* and a number *k*, prints the first occurrence of a substring of size *k* containing only identical characters. By "substring of length *k*" we mean a contiguous sequence of *k* elements within the string. You can assume that *k* is positive and greater than 0.

Examples:

For k = 1 and my_string = abaadddefggg, you should print:

For length 1, found the substring a!

For k = 3 and my_string = abaadddefggg, you should print:

For length 3, found the substring ddd!

For k = 2 and my_string = abcccccd, you should print:

For length 2, found the substring cc!

In case no substring of size *k* is found, a message will be printed saying no suitable substring was found. For example, for *k* = 9 and my_string = abaadddefggg, you should print:

Didn't find a substring of length 9

Hint: recall how to easily generate a string which is a single character repeated *k* times.