R&D SOFTWARE ENGINEER

INGENIA
passion for motion

Overview

The purpose of this test is to better understand your technical knowledge and how you will approach the different described problems. This is not a programming exam and therefore if you don't know how to implement the solution for the problem, just look for alternatives methods of describing how you will solve it (algorithm description, pseudo-code, diagrams, etc).

Exercise 1

We define the *valencia* of a natural number *n* as the absolute value of the subtraction between the sum of the digits that are in odd positions and the sum of the digits that are in even positions (the positions are counted starting in one and from the right to the left). *n* is balanced if its valencia is 0.

For instance 15741 is a balanced number, because the sum of the digits that are in odd positions and the sum of the digits that are in even positions is 9, therefore it has valencia 0. However, 31 is not a balanced number, because its valencia is 2.

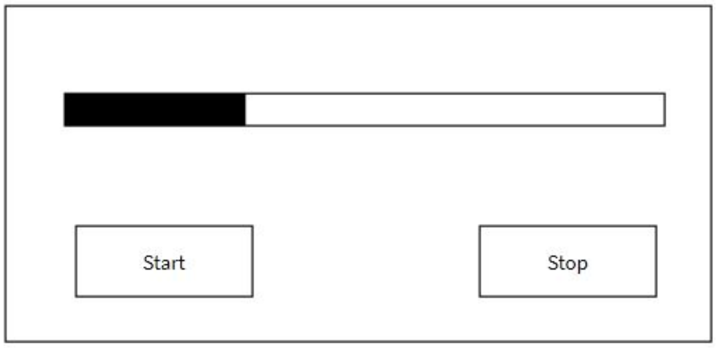
What should you do?

- Your task is to write a program in **Python** that, given a non empty sequence of natural numbers, prints the first balanced number of the sequence. If there is not any balanced number, print the greatest valencia of the numbers in the sequence.
- As other teammates will likely have to work on this code later, create a Git repository for it (e.g. GitHub). Use different commits for each feature implementation.

Exercise 2

We would like to develop a simple GUI that shows a progress bar together with two buttons (Start & Stop). When pressing Start button the application should launch a secondary process that will take 5 seconds and should update the progress bar. This process could be aborted in any moment by pressing the Stop button.

Following there is mockup of the graphical implementation:



What should you do?

- Your task is to write a program in **PyQT** that implements the described application.
- As other teammates will likely have to work on this code later, create a Git repository for it (e.g. GitHub). Use different commits for each feature implementation.

Exercise 3

In this problem you must implement several functions on lists in Python.

1. Write a function *myLength(L)* that, given a list, returns its length.
2. Write a function *myMaximum(L)* that, given a non-empty list, returns its maximum.
3. Write a function *average(L)* that, given a non-empty list of numbers, returns its average.
4. Write a function *buildPalindrome(L)* that, given a list, returns the palindrome that starts with the reverse of the list.
5. Write a function *remove(L1, L2)* that, given a list *L*1 and a list *L*2, returns the list *L*1 after removing the occurrences of the elements in *L*2.
6. Write a function *flatten(L)* that recursively flattens a list whose elements may also be lists of different levels. Hint: use recursion and the *isinstance(x, list)* built-in function.
7. Write a function *oddsNevens(L)* that, given a list of integers, returns two lists, one with all the odd numbers and one with all the even numbers, in the same relative order than the original.

8. Write a function *primeDivisors(n)* that returns the list of all prime divisors of a non-zero positive integer.
9. Write a function *is_increasing*(*L*) that returns a list of booleans. A number is increasing if every digit is less than or equal to the digit which is on its right (if any).