# Assignment #5: Functions

---

**Master in Informatics and Computing Engineering**
**Programming Fundamentals**
**Instance: 2018/2019**

## 0. Introduction

***Goals**: to write programs using functions*

***Pre-requirements (prior knowledge)**: See bibliography of Lecture #7 and Lecture #8*

***Rules**: You may work with colleagues, however, each student must write and submit in Moodle his or her this assignment separately. Be sure to indicate with whom you have worked. We may run tools to detect plagiarism (e.g.duplicate code submitted)*

***Deadline**: 8:00 Monday of the week after (29/10/2018)*

***Collaborators**: list their codes in the file `collaborators.py`*

## 1. Sum of integers

Write a Python function `sum_numbers(n)` that returns the sum of all positive integers up to and including `n`. Save the program in the file `sumNumbers.py`

For example:

- for n=`10`, the function returns the integer `55` (because 1+2+3+. . . +10)

*By the end of your work with this assignment, to submit the activity using Codeboard, you'll be asked to copy the program to the body of the function `sum_numbers` in the file `sumNumbers.py`.*

## 2. Perfect numbers

Write a Python function `is_perfect(n)` to check whether a number `n` is perfect or not. Save the program in the file `perfect.py`

In number theory, a perfect number is a positive integer that is equal to the sum of its proper positive divisors, that is, the sum of its positive divisors excluding the number itself.

For example:

- for n=`6`, the function returns `True`
- for n=`12`, the function returns `False`
- for n=`28`, the function returns `True`

*By the end of your work with this assignment, to submit the activity using Codeboard, you'll be asked to copy the program to the body of the function `is_perfect` in the file `perfect.py`.*

## 3. Assembly digits

Write a Python function `adigits` that receives 3 strings, each one with a single character, representing a decimal digit. Save the program in the file `aDigits.py`.

The function returns the highest integer number that you can assemble with the 3 digits given as parameters.

For example:

- `adigits("4", "2", "5")` returns the integer `542`
- `adigits("9", "1", "9")` returns the integer `991`

*By the end of your work with this assignment, to submit the activity using Codeboard, you'll be asked to copy the program to the body of the function* ***`adigits`*** *in the file* ***`aDigits.py`***

## 4. Mastermind

Write a function `mastermind(g1, g2, g3, c1, c2, c3)` to evaluate a *single line* of a mastermind game. Save the program in the file `mastermind.py`

The function receives 6 single character strings. Each string can be "`b`" for blue, "`w`" for white and "`y`" for yellow. The first 3 arguments are the *user guess*. The last 3 arguments are the *correct key*. Argument 1 is the user guess for the value at argument 4 and so on.

You should give 3 points for each user guess that is completely correct, that is, same color at the same position in the key and 1 point if the user guessed the color right but at the wrong position (that is, the color exists in the key but at another wrong position).

For example:

- `mastermind("b", "w", "y", "b", "w", "y")` returns the integer `9`
- `mastermind("b", "b", "y", "b", "w", "b")` returns the integer `4`
- `mastermind("b", "w", "y", "w", "y", "w")` returns the integer `2`

*By the end of your work with this assignment, to submit the activity using Codeboard, you'll be asked to copy the program to the body of the function* ***`mastermind`*** *in the file* ***`mastermind.py`***

## 5. Get positions

Write a function `get_positions` that receives two arguments: `word_list` (a list with **3** strings) and `sentence` (a string). Save the program in the file `getPositions.py`

The second argument contains 2 words that appear in the 1st argument concatenated with a single space in between.

The function returns a string with the position in the list of the first word and the position of the second word in the list, separated by a single space. Positions start counting at zero.

For example:

- `get_positions(["hello", "world", "lousy"], "lousy world")` returns the string "`2 1`"
- `get_positions(["hello", "lousy", "world"], "lousy world")` returns the string "`1 2`"
- `get_positions(["hello", "brave", "world"], "hello world")` returns the string "`0 2`"
- `get_positions(["hello", "brave", "hello"], "hello hello")` returns the string "`0 2`"

*By the end of your work with this assignment, to submit the activity using Codeboard, you'll be asked to copy the program to the body of the function* ***`get_positions`*** *in the file* ***`getPositions.py`***

## The end.

*FPRO, 2018/19*