# **Advanced Data Analytics**

#### Workshop I

This is workshop leverages the power of Python to solve mathematical, statistical, and data analysis problems. Your task is to solve the following problems **without** using any libraries and with the fewest lines of code possible. Academic integrity is imperative. Any form of dishonesty will result in a grade of zero and formal reporting to the university. As a university representative, you are expected to uphold the highest standards of ethical conduct.

### Requirements

- → Solutions must be submitted by midnight on the agreed deadline.
- $\rightarrow$  Solutions *must* be submitted in a Jupyter Notebook .ipynb with its outputs.
- $\rightarrow$  Solutions *must* include your name, student ID and be written in English.
- $\rightarrow$  Solutions *must* be clear, concise, and well documented.
- $\rightarrow$  Solutions *must* be submitted first to the submission library provided.

# Exercise 1 (15 points)

This section explores Python concepts with creative and practical tasks.

- 1. Write a function char\_count(s: str) -> dict[str, int] that returns a dictionary mapping each character in the string s to its frequency. (5 points)
- 2. Write a function generate\_primes(n: int) -> list[int] that generates a list of all prime numbers less than or equal to n. (5 points)
- 3. Write a function  $is\_prime(n: int) \rightarrow bool$  that checks whether a given number n is a prime number. Return True if n is prime, False otherwise. (5 points)

## Exercise 2 (15 points)

This section contains problems to understand and tackle the basics of mathematical operations in Python.

- 1. Write a function factorial(n: int) -> int that calculates the factorial of a given number n iteratively. (5 points)
- 2. Write a function fibonacci(n: int) -> list[int] that generates a list containing the first n numbers in the Fibonacci sequence. (5 points)
- 3. Write a function sum\_naturals(n: int) -> int that calculates the sum of the first n natural numbers. (5 points)

### Facultad de Ingeniería

# Exercise 3 (15 points)

This section contains problems to understand and tackle the basics of statistical operations in Python.

- 1. Write a function calculate\_stats(data: list[float]) -> tuple[float, float] that calculates the mean, median, and mode of a given list of numbers. (5 points)
- 2. Write a function variance\_and\_std(data: list[float]) -> tuple[float, float] that calculates the variance and standard deviation of a dataset. (5 points)
- 3. Write a function weighted\_mean(data: list[float], weights: list[float]) -> float that calculates the weighted mean of a dataset data given a corresponding list of weights weights. Ensure both lists have the same length. (5 points)

## Exercise 4 (15 points)

This section focuses on analytical problems to develop problem-solving and logical reasoning in Python.

- 1. Write a function is\_palindrome(s: str) -> bool that checks if a given string s is a palindrome. Return True if it is, False otherwise. (5 points)
- 2. Write a function reverse\_words(sentence: str) -> str that reverses the words in a sentence while preserving their order. (5 points)
- 3. Write a function find\_longest\_word(words: list[str]) -> str that returns the longest word from a list of strings. If there is a tie, return the first one. (5 points)

#### Exercise 5 (15 points)

This section contains intermediate-level problems to reinforce Python concepts.

- 1. Write a function remove\_duplicates(data: list[int]) -> list[int] that removes duplicates from a list while preserving the order of elements. (5 points)
- 2. Write a function group\_by\_key(data: list[tuple[str, int]]) -> dict[str, list[int]] that groups values in *data* by their keys. For example, the input [('a', 1), ('b', 2), ('a', 3)] results in the dictionary {'a': [1, 3], 'b': [2]}. (5 points)
- 3. Write a function flatten\_list(nested\_list: list[list[int]]) -> list[int] that flattens a nested list of integers into a single list. For example, [[1, 2], [3, 4]] becomes [1, 2, 3, 4]. (5 points)