## GRA4157 mid-term

#### 3 October 2025

## Exercise 1 — Lists and Dictionaries (4 points)

Consider the list of integers:

```
nums = [4, -2, 15, 0, 8, 23, -7, 9, 5, -3]
```

- (a) Write a Python function count\_positive(xs) that takes such a list as input and returns the number of positive integers in the list. Do not use list comprehensions or the built-in sum with conditions; use a loop.
- (b) Without running code, determine the outputs of the following calls:

```
print(nums[2:5])
print(nums[-4] + nums[1])
print(nums[::3])
print(nums[::-2])
```

(c) Consider the dictionary

```
grades = {"Alice": 87, "Bob": 92, "Carla": 75, "David": 92, "Eva": 68}
```

Write a function top\_students(d) that takes such a dictionary and returns a list of all names (keys) that have the maximum grade.

(d) You are given two dictionaries of population numbers (in millions):

```
pop_2020 = {"Norway": 5.4, "Sweden": 10.3, "Denmark": 5.8, "Iceland": 0.36}
pop_2025 = {"Norway": 5.6, "Sweden": 10.6, "Denmark": 6.0, "Finland": 5.6}
```

Write a function growth(pop20, pop25) that returns a new dictionary where each key is a country present in both dictionaries, and the value is the difference pop\_2025 - pop\_2020.

## Exercise 2 — Reading and Writing Files (3 points)

Consider the input file data.txt:

```
Header information
Some text that should be ignored
Matrix starts below:
3 4 5
6 7 8
9 10 11
```

(a) Read the file using pure Python. For each row of numbers, put all values into a list of integers. Collect these lists into a larger list called matrix. The final result should look like [[3,4,5],[6,7,8],[9,10,11]].

- (b) Write a Python function sum\_rows(mat) that takes such a matrix (list of lists) as input and returns a new list where each element is the sum of the corresponding row. For the example above, the output should be [12, 21, 30].
- (c) Write a program that writes the row sums from part (b) to a file called sums.csv. The file should have the header row, sum, followed by one row per line, e.g.:

row,sum 1,12 2,21 3,30

### Exercise 3 — Vectorized Computations and Pandas (5 points)

You have been hired by HydroAnalytics AS to analyze sensor data from a hydropower plant. The dataset is stored in a Pandas DataFrame named data with the following columns:

	timestamp	flow_rate	head	power_output
0	00:00:00	120.0	50.0	5300.0
1	00:00:10	122.0	49.8	5280.0
2	00:00:20	125.0	50.2	5400.0
3	00:00:30	130.0	50.1	5500.0
•••				

Here: - flow\_rate is measured in  $m^3/s$ , - head is the water head in meters, - power\_output is the actual generated power in kW.

(a) According to physics, the theoretical power output is given by

$$P_{theoretical} = \rho \cdot g \cdot Q \cdot H \cdot \eta,$$

where  $\rho = 1000 \text{ kg/m}^3$ ,  $g = 9.81 \text{ m/s}^2$ , Q is flow\_rate, H is head, and  $\eta = 0.9$  is a constant efficiency factor. Using NumPy vectorized computations, add a new column p\_theoretical with the theoretical power output.

(b) (2 pts) Compute a new column efficiency defined as

$$\mathrm{efficiency} = \frac{\mathtt{power\_output}}{\mathtt{p\_theoretical}}$$

for each timestamp. Then, using Pandas, identify all time intervals (start and end timestamps) where the efficiency was consistently below 0.85 for at least 30 seconds.

(c) (2 pts) The engineers claim that some rows in the dataset are redundant: if three consecutive rows lie perfectly on the same straight line in the power\_output vs. time curve, then the middle row can be removed without changing the curve. Write a function that removes all such redundant rows using vectorized computations (no explicit Python for-loops). The resulting DataFrame should be as small as possible while preserving the exact piecewise linear shape of the original curve.

# Exercise 4 — Web Scraping and Pandas (5 points)

You will now work with web scraping and Pandas to analyze population data of U.S. states and territories.

(a) Use the requests library with an appropriate header (for example User-Agent: PythonRequests) to fetch the HTML source from the following page: https://en.wikipedia.org/wiki/List\_of\_U.S.\_states\_and\_territories\_by\_population

- (b) (2 pts) Use BeautifulSoup to locate the first element in the page. Recall that an HTML table is typically organized with:
  - a header row () containing column names inside cells,
  - subsequent rows containing data values inside cells.

Use the image provided in Figure 1 (file List\_of\_US\_States.png) to interpret the table structure. Extract the headers from the first row of the table, and then loop over the remaining rows. For each row, build a dictionary where the keys are the column names (from the header row) and the values are the corresponding entries. Collect all dictionaries into a list.

- (c) Convert the list of dictionaries into a Pandas DataFrame. Make sure to clean the data so that numerical columns are stored as numeric types (e.g. integers or floats) rather than strings.
- (d) Using the cleaned DataFrame, answer the following:
  - Which entry has the largest value in the population column?
  - What was the US population in 2020 and 2024, according to the data in the table?
  - What is the combined population of all entries that individually account for less than 1% of the U.S. population?

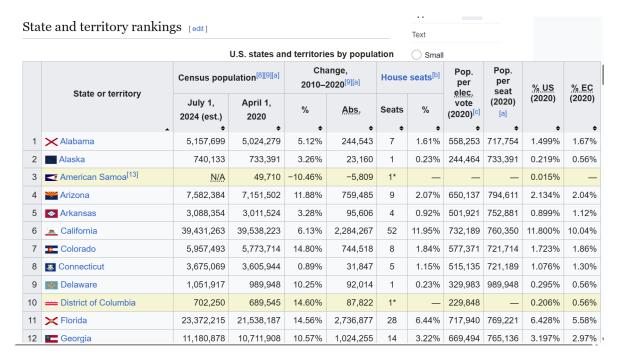


Figure 1: Excerpt of the Wikipedia table with U.S. states and territories by population.

### Good luck!