Luiss
Libera Università Internazionale degli Studi Sociali Guido Carli

# Algorithms A.Y. 2022/2023

## Software Project v2 – A Cryptocurrency Explorer and Market Analyzer

Irene Finocchi, Flavio Giorgi, Bardh Prenkaj
finocchi@luiss.it, fgiorgi@luiss.it, bprenkaj@luiss.it

28 February 2023

LUISS

Dipartimento di Impresa e Management

# Group Project Work

## The project requires to:

- read a cryptocurrency dataset
- implement *efficient* algorithmic solutions to different problems

## We release three different parts:

1. February 7 (Today) – due to February 28 (not mandatory)
2. **February 28 – due to March 28 (not mandatory)**
3. March 28 – due May 14 (mandatory)

# Software Project: Input Data

**How to store financial data in a simple way?**

You are given as input a .txt file containing a list of stocks and additional details. Each line has:

crypto_name, day, price, volume

The values represent the price and volume for the crypto_name (e.g., ALGO) in that day.

| Crypto | Day | Price | Volume |
|---|---|---|---|
| **Gala** | 458 | 45 | 5559100 |
| **1inch** | 507 | 288 | 1938100 |
| **Etherium** | 464 | 75 | 3553000 |
| **Bitcoin** | 723 | 65 | 18966800 |
| **Gala** | 397 | 97 | 1314100 |
| **Algorand** | 588 | 1290 | 0 |
| **Algorand** | 581 | 1290 | 0 |
| **Etherium** | 727 | 504 | 0 |
| **Tether** | 643 | 1398 | 0 |

# Software Project: Input Data

You are given four datasets*: dataset_small.txt, dataset_medium.txt, dataset_large.txt, dataset_full.txt*:

| Filename | Size | Rows | Cryptos | Average monitoring days per crypto |
|---|---|---|---|---|
| dataset_small.txt | 901KB | ~24k | 98 | 247 |
| dataset_medium.txt | ~1.8MB | ~48k | 98 | 492 |
| dataset_large.txt | ~2.6MB | ~69k | 98 | 707 |
| dataset_full.txt | ~4.60MB | ~120K | 98 | 1232 |

In the first release we suggest to use only the “dataset_small.txt”.

# Group Project Work : Task 2

In the second project release we require to design and implement :

1) **A Python function that sorts the crypto data**, ordering the cryptos first in alphabetical order and according to the day of monitoring.

2) **A python function to fetch the maximum value**, which takes in input a crypto **C** and a month **m** and returns the maximum price value and the day in which this maximum was reached.

3) **A python function that searches information**, taking in input a value and a crypto **C**, it returns the day in which **C** reached its price closest to the input value.

# Group Project Work : Task 2 function 1

Example:

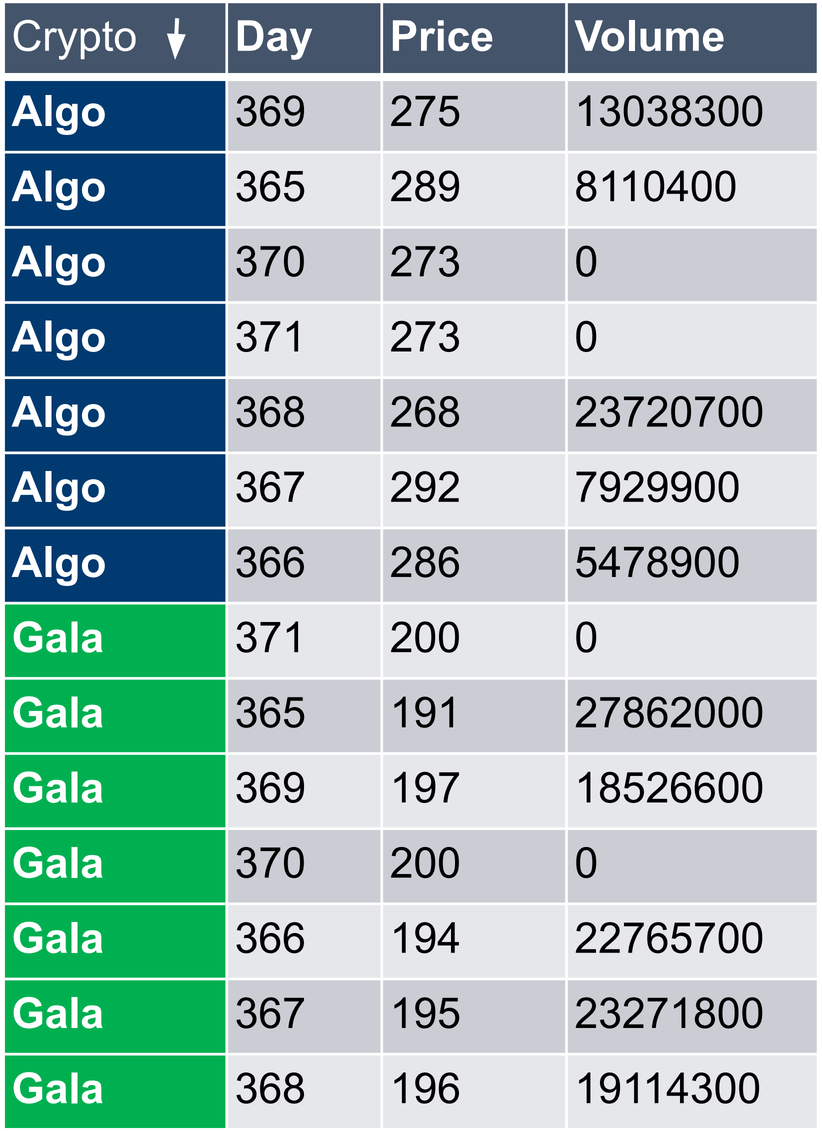
**1)** Sort the data according to the alphabetical order.

**2)** Sort each crypto of the data structured obtained from 1) according to the monitoring day

| Crypto | Day | Price | Volume |
|---|---|---|---|
| Gala | 371 | 200 | 0 |
| Algo | 369 | 275 | 13038300 |
| Algo | 370 | 273 | 0 |
| Algo | 371 | 273 | 0 |
| Gala | 369 | 197 | 18526600 |
| Gala | 370 | 200 | 0 |
| Gala | 365 | 191 | 27862000 |
| Algo | 365 | 289 | 8110400 |
| Algo | 366 | 286 | 5478900 |
| Algo | 367 | 292 | 7929900 |
| Algo | 368 | 268 | 23720700 |
| Gala | 366 | 194 | 22765700 |
| Gala | 367 | 195 | 23271800 |
| Gala | 368 | 196 | 19114300 |

# Group Project Work : Task 2 function 1

Answer to 1):



| Crypto | Day | Price | Volume |
|---|---|---|---|
| Algo | 369 | 275 | 13038300 |
| Algo | 365 | 289 | 8110400 |
| Algo | 370 | 273 | 0 |
| Algo | 371 | 273 | 0 |
| Algo | 368 | 268 | 23720700 |
| Algo | 367 | 292 | 7929900 |
| Algo | 366 | 286 | 5478900 |
| Gala | 371 | 200 | 0 |
| Gala | 365 | 191 | 27862000 |
| Gala | 369 | 197 | 18526600 |
| Gala | 370 | 200 | 0 |
| Gala | 366 | 194 | 22765700 |
| Gala | 367 | 195 | 23271800 |
| Gala | 368 | 196 | 19114300 |

# Group Project Work : Task 2 function 1

Answer to 2):

| Crypto | Day ↓ | Price | Volume |
|---|---|---|---|
| Algo | 365 | 289 | 8110400 |
| Algo | 366 | 286 | 5478900 |
| Algo | 367 | 292 | 7929900 |
| Algo | 368 | 268 | 23720700 |
| Algo | 369 | 275 | 13038300 |
| Algo | 370 | 273 | 0 |
| Algo | 371 | 273 | 0 |
| Gala | 365 | 191 | 27862000 |
| Gala | 366 | 194 | 22765700 |
| Gala | 367 | 195 | 23271800 |
| Gala | 368 | 196 | 19114300 |
| Gala | 369 | 197 | 18526600 |
| Gala | 370 | 200 | 0 |
| Gala | 371 | 200 | 0 |

# Group Project Work : Task 2 function 2

Example:

**1)** Given month m = 13 and crypto C = “Algo”, get the sub-table that corresponds to the crypto in input in the desired month.

**2)** Get the maximum price and the day in which it is reached.

| Crypto | Day | Price | Volume |
|---|---|---|---|
| **Algo** | 365 | 289 | 8110400 |
| **Algo** | 366 | 286 | 5478900 |
| **Algo** | 367 | 292 | 7929900 |
| **Algo** | 368 | 268 | 23720700 |
| **Algo** | 369 | 275 | 13038300 |
| **Algo** | 370 | 273 | 0 |
| **Algo** | 371 | 273 | 0 |
| **Gala** | 365 | 191 | 27862000 |
| **Gala** | 366 | 194 | 22765700 |
| **Gala** | 367 | 195 | 23271800 |
| **Gala** | 368 | 196 | 19114300 |
| **Gala** | 369 | 197 | 18526600 |
| **Gala** | 370 | 200 | 0 |
| **Gala** | 371 | 200 | 0 |

# Group Project Work : Task 2 function 2

Answer to 1):

| Crypto | Day | Price | Volume |
|---|---|---|---|
| Algo | 365 | 289 | 8110400 |
| Algo | 366 | 286 | 5478900 |
| Algo | 367 | 292 | 7929900 |
| Algo | 368 | 268 | 23720700 |
| Algo | 369 | 275 | 13038300 |
| Algo | 370 | 273 | 0 |
| Algo | 371 | 273 | 0 |

**N.B.: Assume that each month has 30 days**

# Group Project Work : Task 2 function 2

Answer to 2):

| Crypto | Day | Price | Volume |
|---|---|---|---|
| **Algo** | 367 | 292 | 7929900 |

# Group Project Work : Task 2 function 3

Example:

**1)** Search for the price value of Gala closest to 199.

| Crypto | Day | Price | Volume |
|---|---|---|---|
| Algo | 365 | 289 | 8110400 |
| Algo | 366 | 286 | 5478900 |
| Algo | 367 | 292 | 7929900 |
| Algo | 368 | 268 | 23720700 |
| Algo | 369 | 275 | 13038300 |
| Algo | 370 | 273 | 0 |
| Algo | 371 | 273 | 0 |
| Gala | 365 | 191 | 27862000 |
| Gala | 366 | 194 | 22765700 |
| Gala | 367 | 195 | 23271800 |
| Gala | 368 | 196 | 19114300 |
| Gala | 369 | 197 | 18526600 |
| Gala | 370 | 200 | 0 |
| Gala | 371 | 200 | 0 |

# Group Project Work : Task 2 function 3

Answer to 1):

**Get the entries with the minimum distance**

| Crypto | **Day** | **Price** | **Volume** | **Distance of Price with 199** |
|---|---|---|---|---|
| **Gala** | 365 | 191 | 27862000 | 8 |
| **Gala** | 366 | 194 | 22765700 | 5 |
| **Gala** | 367 | 195 | 23271800 | 4 |
| **Gala** | 368 | 196 | 19114300 | 3 |
| **Gala** | 369 | 197 | 18526600 | 2 |
| **Gala** | 370 | 200 | 0 | 1 |
| **Gala** | 371 | 200 | 0 | 1 |

# Group Project Work : Task 2 function 3

Answer to 1):

| Crypto | Day | Price | Volume | Distance of Price with 199 |
|---|---|---|---|---|
| **Gala** | 370 | 200 | 0 | 1 |
| **Gala** | 371 | 200 | 0 | 1 |

**Break ties and return the first day in which the price closest to 199 was reached**

# Group Project Work : Task 2 function 3

Answer to 1):

| Crypto | Day | Price | Volume | Distance of Price with 199 |
|---|---|---|---|---|
| **Gala** | 371 | 200 | 0 | 1 |

# Group Project Work : Implementation

We are providing you a skeleton of the code. You should modify the code we provide, adding the missing parts.

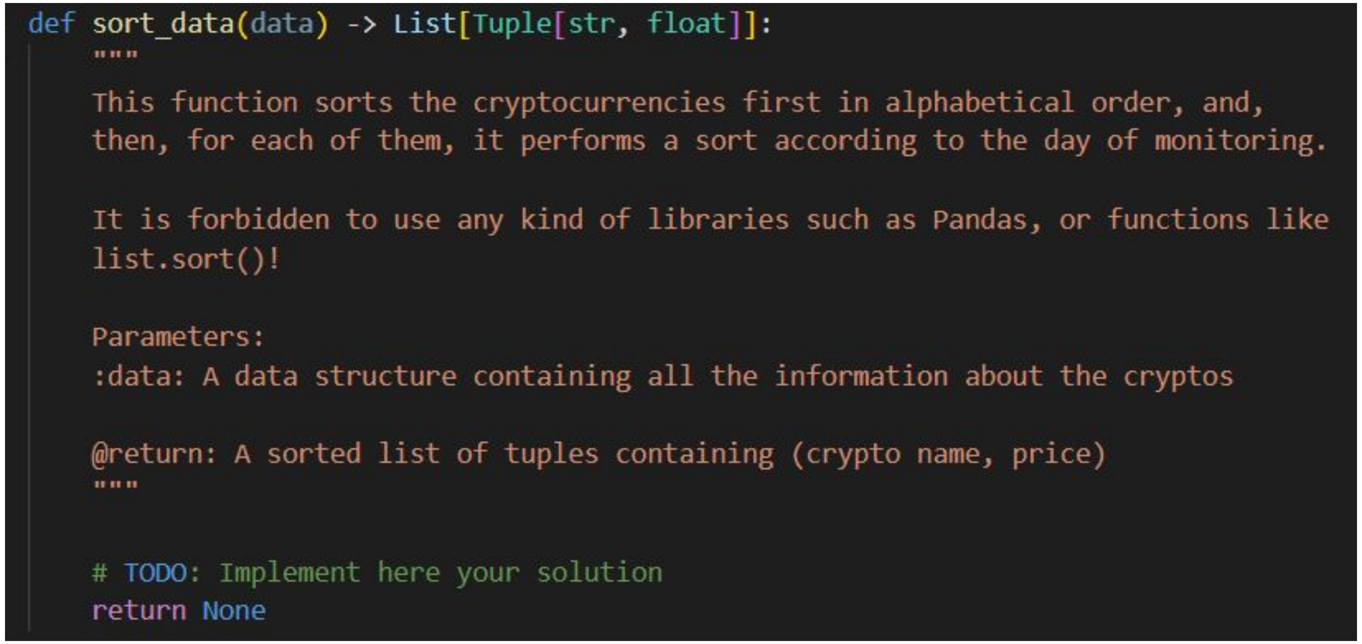
In particular, you have to implement **“group0.py”**. You can use this file to also add your helper functions.

You can download the code from Luiss-learn or Github:
https://github.com/flaat/Algorithms-2022-2023-Project

# Group Project Work : Implementation

The python file **"group0.py":**

The function **sort_data** is called on the data structure that you return with **read_file**



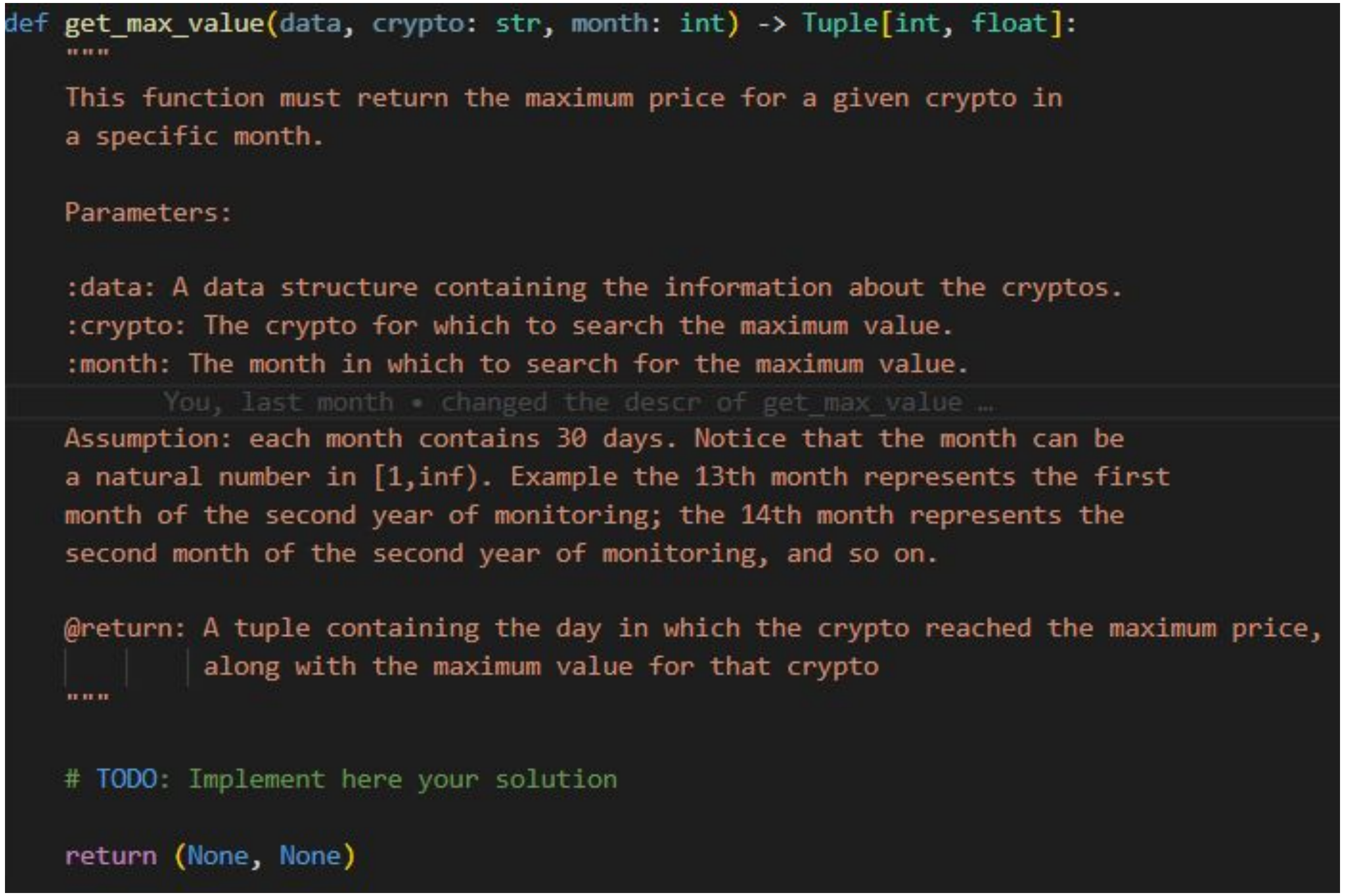
```python
def sort_data(data) -> List[Tuple[str, float]]:
    """
    This function sorts the cryptocurrencies first in alphabetical order, and,
    then, for each of them, it performs a sort according to the day of monitoring.

    It is forbidden to use any kind of libraries such as Pandas, or functions like
    list.sort()!

    Parameters:
    :data: A data structure containing all the information about the cryptos

    @return: A sorted list of tuples containing (crypto name, price)
    """


    # TODO: Implement here your solution
    return None
```

LUISS

# Group Project Work : Implementation

The python file **“group0.py”:**

The function **get_max_value** receives as input the crypto name (e.g., *”Gala”*) and the month, it outputs the maximum price of the crypto in that particular month and the day in which it was reached.



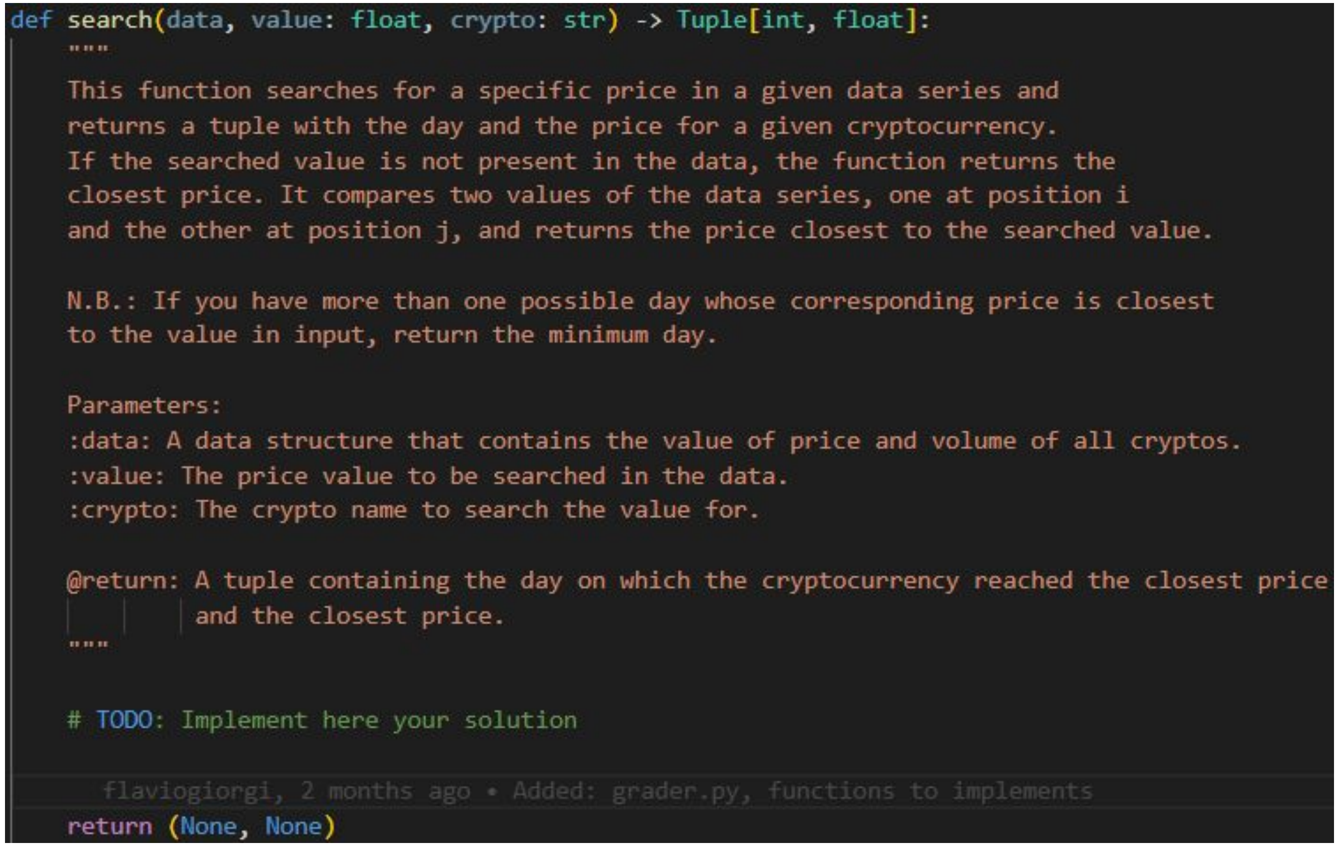
```
def get_max_value(data, crypto: str, month: int) -> Tuple[int, float]:
    """
    This function must return the maximum price for a given crypto in
    a specific month.

    Parameters:

    :data: A data structure containing the information about the cryptos.
    :crypto: The crypto for which to search the maximum value.
    :month: The month in which to search for the maximum value.
            You, last month • changed the descr of get_max_value ...
    Assumption: each month contains 30 days. Notice that the month can be
    a natural number in [1,inf). Example the 13th month represents the first
    month of the second year of monitoring; the 14th month represents the
    second month of the second year of monitoring, and so on.

    @return: A tuple containing the day in which the crypto reached the maximum price,
             along with the maximum value for that crypto
    """

    # TODO: Implement here your solution

    return (None, None)
```

LUISS

# Group Project Work : Implementation

The python file **“group0.py”:**

The function **search** performs the modified searching algorithm described before! It takes in input the sorted data structure from **sort_data**



```
def search(data, value: float, crypto: str) -> Tuple[int, float]:
    """
    This function searches for a specific price in a given data series and
    returns a tuple with the day and the price for a given cryptocurrency.
    If the searched value is not present in the data, the function returns the
    closest price. It compares two values of the data series, one at position i
    and the other at position j, and returns the price closest to the searched value.

    N.B.: If you have more than one possible day whose corresponding price is closest
    to the value in input, return the minimum day.

    Parameters:
    :data: A data structure that contains the value of price and volume of all cryptos.
    :value: The price value to be searched in the data.
    :crypto: The crypto name to search the value for.

    @return: A tuple containing the day on which the cryptocurrency reached the closest price
             and the closest price.
    """


    # TODO: Implement here your solution

      flaviogiorgi, 2 months ago • Added: grader.py, functions to implements
    return (None, None)
```

LUISS

# Group Project Work

## Report:

- For the first two parts, you can write a simple presentation about your implementation (about 2 slides). NOT MANDATORY, but HIGHLY SUGGESTED!

- The final release is MANDATORY TO PASS the Project, and you must provide a presentation with at most 8 slides.

Describe your algorithmic idea, main implementation details, and experiments. You should try to analyze the asymptotic cost of your implementation.

# Group Project Work

## Score:

- The project contributes up to 8 points (added to the theory score).
- If you miss the May deadline, the maximum grade is lowered: you can achieve max 6 points (if you deliver the project by the second exam session), max 5 points (third session), max 4 points (fourth - and last - session)
- For **top projects**, we might consider assigning an extra **1-point bonus**.

You should work in groups of 3 students.

More details on Luiss-learn and Project guidelines pdf.

Thank you!