Playing with statistics on municipalities in France

Problem-based learning (PBL) for students of of the Advanced Algorithms 1 module

Lille
729 km
Paris
514 km
Rennes
660 km
Strasbourg
536 km
Nantes
590 km
Genève
220 km
Lyon
64 km
Bordeaux
521 km
Saint-Etienne
Toulouse
527 km
Nice
462 km
Marseille
309 km

Required work

The subject is based on the data contained in the

https://www.data.gouv.fr/fr/datasets/r/f5df602b-3800-44d7-b2df-fa40a0350325 , which you can download from this link or via Moodle (in the same folder as your project).

The aim of APP is to provide algorithms for sorting France's communes according to certain criteria. The data is in .csv (*comma separated values*) format. You need to import the data into a python script.

You are asked to implement algorithms to sort the communes of France in ascending order of distance from a given town. The aim is to come up with the fastest possible algorithm, choosing from the two fastest sorting algorithms you've studied.

The implementation you propose must not use either the *sorted* function or the *sort* function or the *pandas* or similar library and their sorting facilities.

Your program will also give the following statistics: *min* (nearest city to the city passed in parameter), *first quartile*, *median*, *max distance*. The implementation of a function giving the quantile of order

You're a team of trainee engineers, and as part of the overall project to create a social GPS application, you've been asked to use the quickest "home-made" sorters to rank French towns in order of proximity to a reference town. Your code will then be integrated into a larger application

When the reference commune is the first arrondissement of Paris, a possible (but not obligatory) output is :

```
                min :       0.82 km pour PARIS 02 (75002)
   premier quartile :     187.15 km pour PONT SUR SAMBRE (59138)
            médiane :     318.88 km pour BELLEFONTAINE (88370)
 troisième quartile :     459.81 km pour QUEYSSAC (24140)
                max :    9422.32 km pour ST PHILIPPE (97442)
```

Your mission is to design a tool that allows you to :

1. **Load data :**

    - Read data from a csv file containing all possible commune information, including GPS coordinates, and correct any errors and inconsistencies.

    - Introduce data in a suitable structure to facilitate handling and searching

2. **Putting data in the right format :**
    - Implement intermediate functions to :
        - Organize data in a clear, comprehensible structure.
        - Display complete data for a city.
        - Retrieve unique information about a town (name, town name, GPS coordinates, latitude, longitude, etc.).

3. **Sort and display data :**
    - Implement and adapt the sorting algorithms from the course to classify cities:
        - Adapt merge sorting and quick sorting to classify French towns in order of proximity to a city.
        - Display this data with quantiles (nearest city, first quartile, median, farthest city).
        - Highlight (via a dedicated function) and choose the attributes that will enable you to estimate the distance between two towns (GPS coordinates of the town hall? of the town center? Haversine formula?).

(*continued on next page*)

4. **Compare sorting algorithms**
    - Enable the user to :
        - Carry out a comparative study of sorting algorithms (merge, fast and insert), with graphical display using the ***matplotlib*** Python library. Students will plot the duration of different sorts as a function of the size of the data to be sorted.
        - Using your data and an argument supported by your graphs, answer the following question: "Is there any point in using insertion sorting?

Resources for dealing with the problem situation

- Course 3 and TD 3 "Sorting algorithms: Sorting by insertion, Sorting by selection, Sorting by bubbles".
- Course 4 and TD 4 "Sorting algorithms 2: Merge sorting, fast sorting)
- The "files" course (1st semester course: "Python" module)
- Optional: "Tkinter graphical interfaces" course (1st semester course: "Python" module)
- The dataset available on Moodle

APA learning objectives :

- Understand and master sorting algorithms.
- Test and compare sorting algorithms and identify the fastest algorithm.
- Allocate roles and tasks within the APP.
- Relevant comments on programs.
- Summarize the group's work.
- Use Python to handle a large-scale project involving government data
- Import, use and test a dataset from a CSV file.
- Understand the link between the theoretical complexity calculated in class and the actual time taken by the various sorting operations studied this year.