Programming with Data – Coursework II
Problem solving and Scalability

The goal of this coursework is to make you apply the concepts and general methods seen in class and understand the scalability of your code. This assignment requires you to upload a short Python source code and an essay/technical annex that describes i) your analysis of the problems described below, ii) your algorithmic solution and the Python code for it and iii) the scalability analysis of your code, tested against random instances of the problem. Work is organized in phases, as described below.

Phase 1: MyHealthcare device: Vital signs simulator (20 marks)

MyHealthcare device is a wearable device for collecting vital sign data. Develop a Python function to simulate the **MyHealthcare device** that generates data for "n" vital sign records (e.g. 1000, 2000 etc.) of a person. The data should be generated according to the rules as presented in the following table. Measures that are out of the normal range are considered abnormal e.g. for temperature normal values are: 37 and 38 and abnormal values are 36 and 39.

lists or dictionary,
import random,

	Temperature	Heart rate	Pulse	Blood pressure	Respiratory rate	Oxygen saturation	pH
Values to generate	36-39 (int)	55-100 (int)	55-100 (int)	120-121 (int)	11-17 (int)	93-100 (int)	7.1-7.6 (float, 1 decimal)
Normal rates	37-38	60-99	60-99	120	12-16	95-100	7.3-7.5
Abnormal rates	36, 39	55-59, 100	55-59, 100	121	11,17	93,94	7.1, 7.2, 7.6

The table below shows example data for three records, n=3. Each data row is called record and consists of 8 values (namely as timestamp, temperature, heart rate, pulse, blood pressure, respiratory rate, oxygen saturation and ph). In this example, abnormal values are highlighted with red font. Note, "ts" is a timestamp (feel free to use your own timestamp or a Python timestamp).

ts	temp	hr	pulse	bloodpr	resrate	oxsat	ph
101	37	61	88	120	**11**	95	7.3
102	**36**	77	75	**121**	13	100	7.4
103	37	76	**54**	120	14	**94**	**7**.2
…							

Notes: To ensure consistency of results please use "seed(404)". Generate data for thousand records, n = 1000. Name the function: **myHealthcare(…)**

Phase 2: Run analytics (30 marks)

For phase 2 or 3 develop 2 different functions and compare (e.g. linear and binary then see which is more efficient with a TS)

Develop a Python function for each of the following analytics:

a) Find abnormal values for pulse or blood pressure.
   - Select a small sample e.g. 50 records and count the instances where a vital sign was out of the normal range for a selected value. Return selected values for each timestamp. Example output for pulse with 3 abnormal values could be [pulse, 3, [[105,56], [109,57], [125,59]]] or {" abnormal_pulse_count": 3, "abnormal_values": [[105,56], [109,57], [125,59]]}, where [105,56] is [timestamp,value]. Feel free to create your own data structure.

Whisker plot scatter bar charts Histograms

b) Present a frequency histogram of pulse rates.
   - Select a small sample e.g. 50 records, find the frequency for pulse rate values. Example output could be: [[55,2],[56,6],[57,4],[59,12],…]

c) Plot the results for 2a and 2b and briefly discuss your observations. What is the complexity of your algorithm?

Notes: Present diagrams and discussions in the report. Name the functions as abnormalSignAnalytics(…), frequencyAnalytics(…).

Phase 3: Search for heart rates using the HealthAnalyzer (30 marks)

Develop a function called HealthAnalyzer. HealthAnalyzer provides a query mechanism to search for a particular sign value, for example it could search for records where the pulse value is 56.

a) Design a solution (including one or more algoritms) to search for a particular pulse rate value (e.g. 56), the algorithm should return a multidimensional list with all the records associated with this value (ts, temp, hr, pulse, bloodpr, resrate, oxsat). Keep in mind that a value might exist more than 1 time. For example, when searching for value 56, the following 3 records are selected.
   [121, 37, 61, **56**, 120, 11, 95, 7.3],
   [126, 38, 62, **56**, 120, 11, 95, 7.3],
   [131, 37, 62, **56**, 120, 11, 94, 7.2]
   For this example, the output could be a list such as: [[121, 37, 61, 56, 120, 11, 95, 7.3], [126, 38, 62, 56, 120, 11, 95, 7.3], [131, 37, 62, 56, 120, 11, 94, 7.2]].
b) What is the complexity of your solution?
c) Plot the heart rate values for records having pulse rate 56.

Notes: Present diagrams and discussions in the report. Name the function: healthAnalyzer (…)

Phase 4: Testing scalability of your algorithm (20 marks)

Benchmark the MyHealthData application simulating n = 1000, 2500, 5000, 7500 and 10000 records from phase 1 (MyHealthcare device).

a) Measure the running time and plot the results for different n values.
b) Present diagrams and discussions in the report.

Notes: Name the function: benchmarking(myHealthcare(…))

Annex: provide short and clear descriptions of your solutions, what is the complexity of your code, why, what is the trade off if you use other algorithms. Just looking for a critical evaluation of how you understand your code.

General guidelines:

- Provide comments in the code to explain functionality.
- Please follow the instructions on your Moodle page for time and mode of submission of the solution.

Please note:

- We expect you to provide more than one solution and compare your findings in terms of "what is the better algorithm to use?" (for example, why you might use linear search or binary search or interpolation search for implementing your algorithms, what are the trade-offs?).
- Extra marks will be awarded for "home made functions" with clear description on algorithmic complexity (in comparison to the build in Python functions with sometimes unknown low level detail and complexity).

Submission: Please upload (a) the Python source codes of your solution and (b) a technical annex (report). About the technical annex: please use your judgement on the right amount of data and length of presentation for a technical description of your solution. In this instructor's opinion, two pages should suffice. Please submit your technical annex in a PDF format. Please use an easy-to-read style similar to that of this document (Times New Roman font or similar, size 12, 1.15 line spacing or higher, justified alignment).

Plagiarism: please be advised that Moodle deploys a state-of-the-art plagiarism detection software[1] to evaluate coursework submissions against both Web sources and other submissions, past and present. Each submission will be scored for originality; submissions with low originality might be discarded or penalized.
It is however possible to insert quotations by using appropriate typographic style and providing the reference:

> *this phrase is an example of a typographic style for citation and reference: it will be discounted by Turnitin analysis.*

Please make use of a formalized citation system and report articles and books you refer to, e.g. [Narayanan et al., 2011] and [Shenoy et al., 2011].

References

[Shenoy et al., 2011]
G. G. Shenoy, M. A. Wagle, A. Shaikh, 2017
Kaggle Competition: Expedia Hotel Recommendations
https://arxiv.org/abs/1703.02915

[Narayanan et al., 2011]
A. Narayanan, E. Shi, B. I. P. Rubinstein, 2011.
Link Prediction by De-anonymization: How We Won the Kaggle Social Network Challenge
Proc. of the 2011 Int'l Joint Conference on Neural Networks.
https://arxiv.org/abs/1102.4374

1 https://turnitin.com/gateway/index.html