Programming Assignment #2

Due Date: 07/11/2023 at 12.15

WARNING: YOU ARE NOT ALLOWED TO USE ANY EXISTING LIBRARIES, UNLESS THE LINK FOR IT IS EXPLICITLY GIVEN THROUGHOUT THIS DOCUMENT (Exception PART II).

In this assignment, you will be working with a variety of well-known sorting algorithms.

You are expected to sort disclosed software vulnerabilities listed by National Institute of Standards and Technology (NIST). NIST provides a well-scaling API to allow researchers and developers to access entries in their database. These entries consist of a brief description of disclosed vulnerabilities along with an impact factor and a severity score.

Part I: Fetching the data from the NIST database

As discussed earlier, you can use NIST's API to access their database. The documentation of the API can be found from the given URL:

`https://nvd.nist.gov/developers/vulnerabilities`

The base URL of the API is:

`https://services.nvd.nist.gov/rest/json/cves/2.0`

The API lets you to download only 2000 entries per request from their database that contains more than 180.000 entries, but you can make multiple requests to reach more data. To do that, you need to set two parameters, which are `startIndex` and

`resultsPerPage`, to create a chain of data fetching. For example, after downloading the first 2000 entries with

`https://services.nvd.nist.gov/rest/json/cves/2.0/?resultsPerPage=2000&startIndex=0`

you can request the second 2000 entries with

`https://services.nvd.nist.gov/rest/json/cves/2.0/?resultsPerPage=2000&startIndex=2000`.

You are expected to download 50.000 entries for this assignment. The entries will be in JSON format, which something you can learn more about from

`https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Objects/JSON#:~:text=JavaScript%20Object%20Notation%20(JSON)%20is,page%2C%20or%20vice%20versa).`

Part II: Parsing the JSON entries into Java Objects

There are many tools to convert JSON format into Java objects, but I provide you one of them. You can use any tool to parse the entries or existing libraries in this section.

`https://json2csharp.com/code-converters/json-to-pojo`

Do not forget to implement `Comparable` interface to be able to compare your Java Objects.

Part III: Sorting entries

After parsing entries, you downloaded from the NIST database, you will sort them based on three criteria: `baseScore`, `impactScore`, and `exploitabilityScore`. The `baseScore` will be the most important criteria, which means the bigger `baseScore` will provide more priority. If the `baseScore` of two vulnerabilities are the same, then you need to compare their `impactScore`. If the `baseScore` and `impactScore` of two

vulnerabilities are the same, then you need to compare their `exploitabilityScore.` If all three scores are the same, the lower `CVE-ID` wins.

Sorting algorithms: Merge Sort, Insertion Sort, Heap Sort, AVL Sort and Quick Sort.

With every given sorting algorithm, you sort vulnerability entries. You will implement the sorting algorithms by yourself (Please do not use any existing library). You will measure the sorting time as you did in Assignment #1 and report it.
Important Note: If Insertion Sort takes too much time, reduce the number of entries to 10.000 and report it in your README file.

Part IV: Submission

You will submit the executable .jar file, the README file that explains how to run your code, and the screenshots of performance of each sorting algorithm to Aybuzem.