GROUP HOMEWORK 1: TRIE RESEARCH

1 Research

Trie is an efficient *information re**Trie**val* data structure. Trie is usually pronounced "try", to distinguish it from other "tree" structures. You can research more about Trie at these references: link 1, link 2 (section 6.4).

Using *Trie*, searching operation can be brought to optimal limit (key length). Using the code given (tries_code.pdf) as the reference, students are requested to fulfill the following requirements:

1. Show the time complexity of the following operations (of *Trie*):
    - Adding a word.
    - Removing a word.
    - Searching a word.
    - Searching words which has the same prefix with length i.
2. Show the advantages of *Trie* comparing to other data structures (designed for searching) which you have learned: `AVL tree` and `Hash table`.

2 Programming

1. Students implement and build a *Trie* with words from the given files, including:
    - File *dict.txt* contains a list of English words sorted ascending. Each word locates on a single line. You have to use this list of words to build the Trie.
    - File *tries_code.pdf* contains the source code for operations on the Trie data structure. **You do not need to use source code from this file**.

2. With the built *Trie* (containing the provided English words), implement an auto-complete feature. The feature will generate a list of *valid English words*[1] which have **prefix** are letters from a given character list.

   - **Input:** List of letters use for generating valid English words. These letters must be on the same line and satisfied the following requirements:
     - Being in lowercase, not in CAPITAL.
     - May appear multiple times.
   - **Output:**
     - The 1st line: An integer `N` indicates the number of created words.
     - Next `N` lines: each line contains a generated word.
   - Example:

Input	**Output**
happ	11 happed happen happened happens happer happy happier happiest happify happily happing

[1] valid English words are words existed in the given dictionary file.

3 Submission

- The submission file must be in the following format: [**SID1_SID2_SID3.zip**] (SID stands for Student ID), is the compression of the [**SID1_SID2_SID3**] folder. This folder contains:
  - The report file must be presented as a document [**SID1_SID2_SID3.pdf**]. This file presented your research from section **1** and the information of code fragment (algorithms, functions) from section **2**.
    - Information (Names, Student IDs) must be declared clearly on the first page of your report. Your working progress (How much work have you completed?) should be demonstrated on this page, too.
    - The report file should be **structured, logical, clear** and **coherent**. The length of the submission should not exceed 20 pages for the document file.
    - All links and books related to your submission must be mentioned.
  - The programming folder [**code_SID1_SID2_SID3**]. This folder contains all the source code for section **2** (only `.cpp` and `.h` files). The code fragment must be clear, logical and commented.
- Submission with wrong regulation will result in a "0" (zero).
- Plagiarism and Cheating will result in a "0" (zero) for the entire course and will be subject to appropriate referral to the Management Board of the CLC program for further action.

THE END