Word Composition Problem Solution

Problem Statement

The given problem statement is all about identifying the

1. longest compounded word
2. second-longest compounded word
3. time taken to process the input file

that given with the problem statement containing alphabetically sorted words. A compounded word is one that can be constructed by concatenating shorter words also found in the same file. The solution processes two input files,

1. `Input_01.txt` (small dataset) and
2. `Input_02.txt` (large dataset) .

Approach

1) The program uses a "Trie" to store words efficiently and to quickly determine prefixes. This reduces the time complexity of searching for compound words.
2) Trie Construction: Words from the input file are inserted into the Trie.
3) Queue Processing: A queue is used to store word-suffix pairs for efficient processing of potential compounded words.
4) Longest and Second-Longest Identification: Words are evaluated as compounded if their suffixes are either directly present in the Trie or can themselves be further broken into valid words using the Trie.
5) By leveraging the Trie and queue, the program ensures efficient prefix lookups and avoids redundant computations.
6) Execution time is measured using `std::chrono` for precise performance tracking in milliseconds.

Note :

- Before Executing the program please make sure that both the input files are in same directory as the program.
- Make sure you a g++ compiler installed in your system.

Steps to Execute

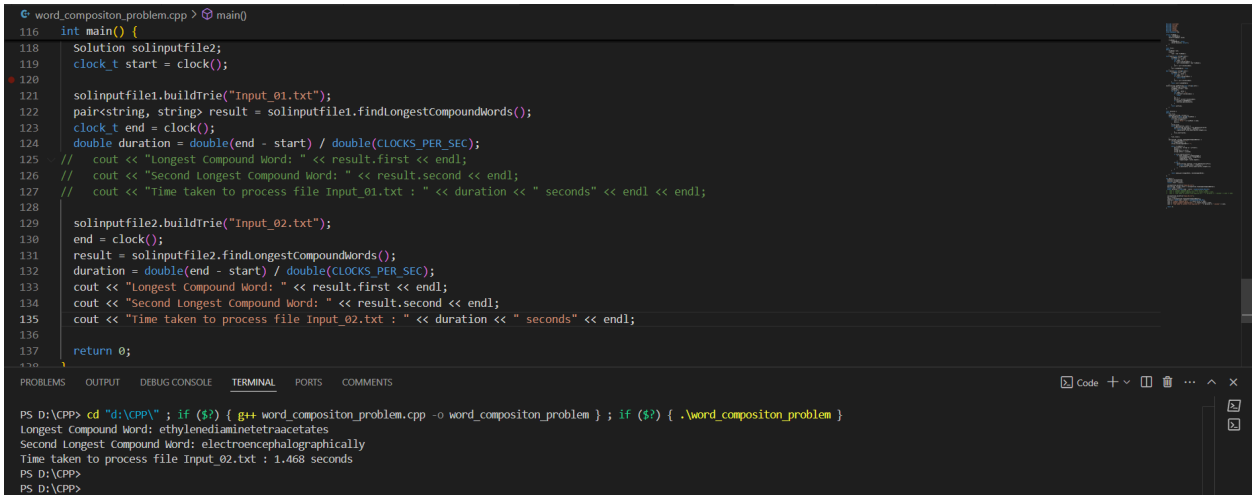
- Open terminal in same directory where your solution is stored.
- To compile the program write following command in terminal/powershell "g++ Solution.cpp" (like “word_compositon_problem.cpp").
- To execute the program write the following command in terminal/powershell "./a.exe" or the name of the compiled file.

Sample Output

For the given sample input files:

Input_01.txt

Input_02.txt



Note: Time taken can be different every time and on any system while compile.

SOLUTION BY ETI GARG