Lab2 Sorting

Problem Description

The problem is similar to Exercise2, but you need to sort words instead of numbers.

Devise and implement an algorithm that sorts words in O(*n*) complexity, where *n* is the number of words in the input file (not counting the first word that has special meaning described below).

The input file contains words separated by whitespaces. Each word consists of English letters a-z. There are 1, 2, or 3 letters in each word, the same letter can be used multiple times.

The first word in the input file is so called *maxword.* It is some word that is equal to or greater than any other word in the file.

Program

Assume that you have *n* arbitrary words in range **a**..*maxword*. The words may be repeated. Sort them using the algorithm of O(*n*) complexity. Also sort them using any other algorithm of your choice.

The program has to be case insensitive. For example, letters **a** and **A** has to be treated as equal.

Sort all words from the file in alphabetical case insensitive order (for example, **a** precedes **aa**, **ab, b, ba,** and **bb**). Ignore the letters case and output words in small case only.

Write a program does both of the following

- reads words from input file **in_abc*X*.txt**, sorts them using O(*n*) algorithm, and output the sorted words in small letters to output file **out_abc*X*a.txt** (where *X* is 1, 2, …)
- reads words again from the same input file, sorts them using any other algorithm of your choice, and output the sorted numbers to output file **out*X*b.txt**

There are two test input files provided – **in_abc10.txt** and **in_abc100.txt**. There is also one sample output file corresponding to the first input file – **out_abc10_sample.txt** (file **out_abc100_sample.txt** is not provided).

Run your program with the test input files. For each input file, two corresponding output files (produced by your program) have to be the same.

Also compare the produced output file **out_abc10a.txt** with the sample file **out_abc10_sample.txt**. Submit the test input and the produced output files together with the program source code.

Hint

Replace characters with integers and use algorithm from Exercise2. For example, a is 01, b is 02, etc., and z is 26.

For instance, word **abc** can be encoded as **10203** (may you skip leading zero? why yes or why not?). Word **a** is **10000**, word **aa** is **10100,** word **z** is **260000,** word **yz** is **252600**.

To figure out the array size *k* encode the *maxword*. For example, if *maxword* is **aaa**, then k is **10101**.

The numbers in these examples can have up to 6 digits. Can you devise other encoding that require only 5 digits?

Input File Format

The input file contains words separated by whitespaces

- the first word is *maxword (maximum possible word below)*
- the follow up words are the words that you need to sort (count these words to figure out *n*)

For example,

zzz // so *maxword* is zzz, the deduced *k* is 262626
ab ijk x zdd // so *n* is 4

In this example, the actual maximum word is **zdd** that can be calculated after all words are read. But we have to allocate array at the beginning, so we specify *maxword* == **zzz** (and the deduced *k* is 262626).

Example

If your input file contains

ddd
Aa aab Abc cb bcd

The output has to be

aa aab abc bcd cd

