# Sabancı University
# Faculty of Engineering and Natural Sciences

## CS 300 Data Structures

**Homework 4**

**Assigned: Dec 9, 2021 Due: Dec 20, 2021 at 11:59pm**

**PLEASE NOTE:**

**SOLUTIONS HAVE TO BE YOUR OWN. NO COLLABORATION OR COOPERATION AMONG STUDENTS IS PERMITTED.**

**10% PENALTY WILL BE INCURRED FOR EACH DAY OF OVERTIME. SUBMISSIONS THAT ARE LATE MORE THAN 3 DAYS WILL NOT GET ANY CREDITS.**

**SUBMISSIONS WILL BE MADE TO THE SUCOURSE SERVER. NO OTHER METHOD OF SUBMISSION WILL BE ACCEPTED.**

# The Problem

In this homework, we will solve a puzzle that involves a list of words. You will be given a *long* list (English*)* words. Then, you will be given two words A and B and be asked if you can start from A and transform it to B by a series of one-character substitutions, one character insertion or one character deletions, and if so, output the corresponding sequence of words. *Note that all intermediate words must be in the list of words given*. For instance, we can convert *bleed* to *blood* by going through a sequence of transformations giving us the sequence of words *bleed, blend, blond, blood*. We can also convert *can* into *pale* by going through a sequence of words *can, man, mane, male, pale*.

# The Input and the Output

You will read from a file called *words.txt* the words that you will use as your database. The words will not be in any particular order and may contain characters such as "-" or apostrophe (') – these should not cause any problems. Each word will be in a separate line in the file. All the characters will be in lowercase (i.e., no capital letters). Once the file is read and processed, you will then go into a loop that does the following:

1. Read two words from the standard input (cin). The words will be separated by a space character.
2. If the first word starts with the character '*' then exit the loop and terminate the program.
3. Make sure both words are in the list; print an error message if either one is not in the list

4. If both words are in the list, find out if you can transform the first word into the second by a sequence of one letter substitutions, insertions, and deletions. You should find the shortest sequence of such transformations. Print a suitable error message if no sequence of one letter transformations exists.
5. If there is such a sequence then, print the first word, the sequence of intermediate words and the last word along with the transformations that indicates what you do to the previous word to find the current word, one word to a line. For example, for the 2nd example above, your output should look like the following

```
can
man (change c at position 1 to m)
mane (insert e after position 3)
male (change n at position 3 to l)
pale (change m at position 1 to p)
```

   In the case of adding a character at the beginning of a word, for example, when going from the word ree to **t**ree, print the following output:

```
tree (insert t after position 0)
```

   For deletion you should write something like "(delete m at position 4)", etc.

6. Go to 1

You should construct your *words.txt* file to test your program. We will use separate list of words to test your programs.

# The Submission

This problem involves the use of a number of data structures and algorithms that we studied in this course. We provided only basic requirements above and there is a LOT of details that you have to decide on by yourself. For instance, you will have to develop a class for representing and manipulating graphs. You should also figure out how you find out if one word can be obtained from the other using one character insertions, deletions, or substitutions, and how you represent the database of words and the transformations between them in a graph.

You should turn in a fully documented source code listing for your main application program source with any additional classes and procedures. Make sure your programs accept input or produce outputs in the specified format and way.

You should follow the following steps:

- Name the folder containing your source files as **XXXX-NameLastname** where **XXXX** is your student number. Make sure you do NOT use any Turkish characters in the folder name. You should remove any folders containing executables (Debug or Release), since they take up too much space.

- Use the Winzip or an equivalent program to compress your folders to a compressed file named, for example, **5432-AliMehmetoglu.zip**. After you compress, please make sure it uncompresses properly and reproduces your folder exactly. We will NOT accept any excuses for files not appearing in your submitted files.

- You then submit this compressed file in accordance with the deadlines above.

Your homework will be graded in the following way:

- If the source code does NOT compile you get 0 points and that is it.

- We will run your program with one or more different *words.txt* files that we will prepare. If your output is completely correct for all the tests you will get 100 points maximum.

- Note that your outputs should be in the exact same format we described above. If they are not, you will lose points.

Good luck