T.C.
DOKUZ EYLUL UNIVERSITY

FACULTY OF ENGINEERING

DEPARTMENT OF COMPUTER ENGINEERING

2024 - 2025
FALL SEMESTER

CME 3205
OPERATING SYSTEMS

ASSIGNMENT 1
TEXT ANALYSIS SERVER

DUE DATE
23:55 – 30.12.2024

In this assignment you are asked to create a simple text analysis server, which will take a single string (that contains one or more words) from a port connection using telnet program (telnet console command that has been used in lab sessions), perform Levenshtein Distance calculation on the inputted words by comparing them to a list of words given in “basic_english_2000.txt” text file.

The text file “basic_english_2000.txt” can be downloaded from the following link:
https://people.sc.fsu.edu/~jburkardt/datasets/words/words.html

You are required to use the file above for your program as a dictionary file, download it and examine it.

Your program should be written in C programming language and it should work on a Linux computer. You should create a simple virtual Linux operating system inside VirtualBox software (the same setup we use in our lab sessions) for development and testing of your assignment.

You are also required to use following global variables to make your code more understandable, modifiable and standardized.

```
int INPUT_CHARACTER_LIMIT = 100;
```

The maximum number of input string size and integer array size your program should work for.

```
int OUTPUT_CHARACTER_LIMIT = 200;
```

The maximum number of output string size and integer array size your program should work for.

```
int PORT_NUMBER = 60000;
```

The port number that will be used by your program. You may need to change it after closing your program, because OS thinks it is still in use by your previous execution.

```
int LEVENSHTEIN_LIST_LIMIT = 5;
```

The number of words your similarity list should contain at most. Basically the 5 closest words, compared to the input, you have found from “basic_english_2000.txt”.

These are the basic variables and their names you should use on your program. A standard pseudo code algorithm/operation of your program is given below.

TEXT ANALYSIS SERVER BASIC EXECUTION STEPS:

First your server starts operation and checks if file “basic_english_2000.txt” exists or not.

If it does not exist, it outputs an error (e.g. ERROR: Dictionary file “basic_english_2000.txt” not found!) and shuts down.

If it is found, the server reads the file and stores its contents for Levenshtein Distance calculation.

Then server starts to wait for a telnet connection.

When a telnet connection is received, the communication steps that has been given in this document are performed.

Your server receives the first string and stores it. It checks if the input string is longer than “INPUT_CHARACTER_LIMIT” global variable. If this is the case, the server will output an error (e.g. ERROR: Input string is longer than 100 (INPUT_CHARACTER_LIMIT) variable!) and shuts down.

If the input string is shorter than “INPUT_CHARACTER_LIMIT”, then server checks if the input string contains characters other than empty space and alphabetical characters. If this is the case, the server will output an error (e.g. ERROR: Input string contains unsupported characters!) and shuts down.

If the input string passes this test as well, then your main process/thread divides the input string into words (assuming this is a sentence and contains more than one words). Then it creates threads that will take each word as input.

These threads then will calculate Levenshtein Distance between this word and all words in dictionary, and return LEVENSHTEIN_LIST_LIMIT (which is 5) closest word matches (the 5 (LEVENSHTEIN_LIST_LIMIT) words with minimum Levenshtein Distance) to the main thread.

After sub threads are finished, the main thread outputs the results for every word given in input text, printing 5 (LEVENSHTEIN_LIST_LIMIT) closest matched words for each word with their Levenshtein Distance value.

While closest words for every word is printed, if the word is not present in dictionary, the user will be prompted to ask if they want to add the word to dictionary or now. If user answer yes, the word is added to the dictionary and original “basic_english_2000.txt” file is updated with new word (You can update this file one word at a time or one time for all words at the end of execution, you are free to choose). If user answer no, then closest (and possibly, alphabetically first) word is chosen to replace original input word.

Lastly, it will print out the input sting and output string (where every word either find in dictionary, added to dictionary or replaced by closest word available in dictionary).

After this, the main process/thread terminates and ends connection, by releasing port 60000 back to the OS, to be able to use it the next time.

A sample port/socket communication between your assignment and telnet program is given below with extra information and comments.

```
telnet localhost 60000
```

With this command, you will be connected to port 60000 of your localhost (your own computer). Assuming this is the port that your assignment is currently working on, you should receive the following message.

```
Hello, this is Text Analysis Server!

Please enter your input string:
Hello their how are yu
```

As you can see we purposefully give an incorrect input, to allow it to fix it.

```
WORD 01: hello
MATCHES: bell (2), cell (2), help (2), hill (2), hollow (2)
```

“Hello” is not present in out text file so we will get closest possible words. Your program should output 5 closest matches, according to global variable LEVENSHTEIN_LIST_LIMIT.

```
WORD hello is not present in dictionary.
Do you want to add this word to dictionary? (y/N): y
```

Because “hello” is not found, the user will be asked if they want to add this word to dictionary or not, just like office software we use today. If user enters “y” or “Y”, word hello will be added to the dictionary and no word change will be done. If user enters “n”, “N” or “” (empty), first closest word (in this case “bell”) will be used in place of “hello”.

```
WORD 02: their
MATCHES: chair (2), hair (2), other (2), shear (2), tear (2)
```

Similar to above “hello”, “their” is not present in dictionary, so we get closest possible words.

```
WORD their is not present in dictionary.
Do you want to add this word to dictionary? (y/N): n
```

Because “their” is not found, the user will be asked if they want to add this word to dictionary or not, just like office software we use today. If user enters “y” or “Y”, word their will be added to the dictionary and no word change will be done. If user enters “n”, “N” or “” (empty), first closest word (in this case “chair”) will be used in place of “their”.

```
WORD 03: how
MATCHES: how (0), cow (1), low (1), now (1), show (1)
```

Unlike others, “how” is present in dictionary, so no word substitution will be performed.

```
WORD 04: are
MATCHES: age (1), arc (1), area (1), arm (1), art (1)
```

“are” is not present in dictionary, so we get closest possible words.

```
WORD are is not present in dictionary.
Do you want to add this word to dictionary? (y/N): y
```

Because “are” is not found, the user will be asked if they want to add this word to dictionary or not, just like office software we use today. If user enters “y” or “Y”, word their will be added to the dictionary and no word change will be done. If user enters “n”, “N” or “” (empty), first closest word (in this case “age”) will be used in place of “are”.

```
WORD 05: yu
MATCHES: you (1), a (2), as (2), at (2), be (2)
```

“yu” is not present in dictionary, so we get closest possible words.

```
WORD yu is not present in dictionary.
Do you want to add this word to dictionary? (y/N): n
```

Because “yu” is not found, the user will be asked if they want to add this word to dictionary or not, just like office software we use today. If user enters “y” or “Y”, word their will be added to the dictionary and no word change will be done. If user enters “n”, “N” or “” (empty), first closest word (in this case “you”) will be used in place of “yu”.

```
INPUT : hello their how are yu
OUTPUT: hello chair how are you
```

As you can see, “chair” is wrong but because we did not add it to dictionary and our dictionary is limited, this was the best result we could get.

```
Thank you for using Text Analysis Server! Good Bye!
```

With this message, the connection is closed by the server and communication ends.

The above example shows how a communication between your server and telnet command should be handled. In addition to this you should also send previously mentioned error messages. There could be more error messages you may need to send, to make your program more user friendly.

If there is an error that is not covered here, you can write your own error message with similar style to given error messages above. After sending an error message, you should close the connection and wait for new telnet connection to be made. You are not required to ask for correct input again, however if you wish, you are allowed to create a program that works like that, rather than closing down.

As you can see in example execution above, you should convert uppercase letters to lowercase before processing, because dictionary file contains words that are lower case.

You should also check if it is possible for you to detect if the input string was over limit (more than 100 characters assuming the default value, you can lower it to test it) or not. If you can detect this, you should return an error explaining to problem to the telnet connection. You should also explain it on your comments and code control and show it. If you cannot detect this, again show and explain it on your comments and code control, what have you tried to detect it and how did it failed to work.

You are advised but not required to write your code from simplest implementation step by step. For example, if you know Python or another high level language, write this program there first. This will allow you to quickly develop correct working program. After that, translate that program to C language, because you know your design is correct, all emerging errors will be caused by C language rules, data types and functionalities. After that add multi threading and debug emerged errors. Lastly, add port communication to finish your program and satisfy the requirements.

You are also required to come to code control during your lab sessions in 02.01.2025, 08.01.2025 and 09.01.2025. The code control appointment table is included with this assignment paper, please check it and make sure you attend to your code control appointment. If you do not come to code control, even if you made an upload with perfectly working code, you grade will be zero.

We are planing to do your code controls in Lab 3 and 4 (the labs were “CME 1251 – Project Based Learning – I” class is held), however if it there is a change of location, you will be informed in Sakai announcements. If there is an extreme scenario and you will not be able to come to your code control, notify us immediately of this situation. You are also advised to remind your group mates of your code control to prevent them from forgetting and getting a zero.

You are also required to create a video, a timed automatic presentation or something similar in functionality which will show your code and program structure in a timely manner, helping you presenting your code in 5 minutes in total (approximately 1.5 minutes per student). You are not required to prepare a full presentation with slides, just your code and some explanation to help you explain it. This should be done by all group members equally, everyone should take equal time to explain their code and program.

Your code control will be 15 minutes in total, 5 minutes for presenting your code, 5 minutes for executing and showing different outputs and 5 minutes for questions and evaluation by your teacher. Your code control must be finished within given time periods.

GRADING REQUIREMENTS:

File and Code Criteria:

- Correct naming of upload file.
- Correct English variable names.
- Correct English comments.
- Correct code quality, indentation and readability.
- Using required global variables with correct names.
- Using correct port number.

Error Detection Criteria:

- Correct error detection and output for non alphabetical and space characters.
  - (including dots, commas, question marks, etc).
- Correct error detection and output if the input string is larger than supported maximum.
  - (given as 100 above stored in INPUT_CHARACTER_LIMIT).
- Explanation of error detection and output on your comments and code control.

Execution Criteria:

- Opening port communication to port 60000.
- Receiving telnet connection by user.
- Sending required information to telnet.
- Receiving input string from telnet.
- Checking for errors inside input string.
- Outputting errors if they are present.
- Using threads for every word inside input string.
- Correct Levenshtein Distance calculation.
- Correct closest matching word list generation.
- Outputting resulting closest matching word list.
- If input word is not in dictionary, asking user for which action to take.
- Adding input word to dictionary or replacing input word.
- Outputting correct output string with input string.
- Closing the port connection after finished operation.

Code Control/Presentation Criteria:

- Individual Criteria:
  - Preparing a video, timed presentation or etc for code explanation for their part of code.
  - Being able to explain the code and program in correct manner.
  - Being able to answer questions in correct manner.
- Group Criteria:
  - Program working correctly.
  - Correct behavior with different and/or wrong inputs.
  - Checking edge cases.

UPLOAD REQUIREMENTS:

You are required to upload the C language code file you have written to the SAKAI. You should compile and test it to make sure it works before upload. You can use a IDE during development but your code must work correctly using console compilation and execution commands (using “gcc” and “./”). If we cannot correctly compile and execute your code in our computers, your grade will be significantly reduced due to not being able to see results to evaluate. You do not need to upload a complied version of your code, just your C language code, because we cannot prove if it is a original compilation or not. For this reason, uploading a complied file is not necessary nor it is requested.

The file you are required to upload are given below with an explanation and an example. You are free to do this assignment as a group of two (2) or three (3) students. Please do not use empty spaces or Turkish language characters in your file name.

For 3 Student Groups:
GROUP_(G_NO)_(S_NUMBER_1)_(S_NAME_1)_(S_NUMBER_2)_(S_NAME_2)_(S_NUMBER_3)_(S_NAME_3).c
Example: GROUP_01_2024510123_fatih_dicle_2024510124_muharremcan_gulye_2024510125_yunus_dogan.c
(Source code you have written in C language)

For 2 Student Groups:
GROUP_(G_NO)_(S_NUMBER_1)_(S_NAME_1)_(S_NUMBER_2)_(S_NAME_2).c
Example: GROUP_01_2024510123_fatih_dicle_2024510124_muharremcan_gulye.c
(Source code you have written in C language)

Late or no submissions will be graded zero. In a rare case of system or upload failure, send your code file to research assistants as an email. This does not guarantee that your code will be accepted or graded but it will be a record of your problem and your code. For this reason, you should finish your code one or two days before your deadline and make an upload, instead of leaving it to last day and hour.

If you want to improve upon it before deadline and a new upload, just delete the original upload and make a new one and add a date time string with format “YYYYMMDD_HHMM” to the end of your file. An example is given below.

For 2 Student Groups:
GROUP_(G_NO)_(S_NUMBER_1)_(S_NAME_1)_(S_NUMBER_2)_(S_NAME_2)_YYYYMMDD_HHMM.c
Example: GROUP_01_2024510123_fatih_dicle_2024510124_muharremcan_gulye_20241212_1920.c
(Source code you have written in C language)

Your uploads will be analyzed for cheating and plagiarism and if it is detected, your assignment will be graded as zero. You can search and learn from online sources for this assignment but write your own source code instead of copy pasting. If you do not, that will be considered as plagiarism.

If you have any questions or problems regarding this lab paper, you can ask about it in our lab sessions. This way every student will hear and note the answer, eliminating need for asking it later.

However, please do not send emails because that would lead to asking the same questions over and over again.

GOOD LUCK TO YOU ALL!