27760 Utku Alkan PA4 Report

Firstly, I defined a struct as Person so that I can save the names, surnames, and genders from the database to an array called people. I have a linecount global variable to hold how many lines are in the database file.

I will continue with my main file. Firstly, I opened the database file with the read option. Then, I read them line by line and put them into the "people" array I just said. I had to clean the last word while reading the file because it was having newline character in it. Also, linecount is stored too. Then, I called my myfunct function with the "." string. Then I simply close the file and finish the execution.

The function called myfunct is actually doing all of the job. First, from the parameter "path" it initializes a directory. Also, with the help of the directory entry (dirent) header, we were able to loop in every file in that directory in a while loop. Since we are calling readdir it will just make the "entry" variable equal to the next directory or file in that directory. In our while loop, firstly I wanted to update our actual path for each file (or directory) in our current directory. Then, I am checking if the entry is a regular file. This will help us to eliminate directories or some weird files but it is not enough to reach for .txt files. So with the help of strrchr, we were able to search the extension of the file by looking after the ".". If it is then we will check if it is a database.txt file which we shouldn't look at. If it is, we will continue with the next file. As a note, if we reach database.txt in a directory; since the checker will be updated to 1, we will do the correction. Then, since we know it is a txt file we can open it with r+ (read and write). After, while looping through every word of that txt file; we will search for the matching names. If any name is matching in the file with our Person structure (from database.txt) we will do these:

Since we have a structure like Mr. "name" "surname"

1) fseek for => SEEK_CUR - 4 - strlen(worder)

will bring us to the beginning of the gender part (Mr. or Ms.). The reason is when we subtract the word's length we will arrive at the beginning of the name. Subtracting 4 is for Mr. (or Ms. which is also 3) plus one space between name and gender.

2) Since we are at the beginning of gender we can update the gender. Whether it is correct or not we can overwrite what is written there so calling fputs with the correct gender solves the problem of putting 3 letters instead of the 3 old gender letters.

3) Now, it's time to update the surname. We first add the length of the name back so we just removed what we did in the first element. Then all we need to do is add two more to reach the surname's first letter with an additional space again.

4) Then the same thing happens as the second element. We just update it with the correct surname.

This will continue until the entire file is scanned. Then we close the file.

There is one last problem. What if the entry is a directory? For this, the program will check if the entry type is DT_DIR. Then we need to check if it is "." or "..". The reason is, we don't want to search for these because "." is representing the current directory, and ".." is representing the parent directory. We don't need to check those since the parent directory is already checked (except the code's directory) and the current directory is being checked right now. After that check, we are assigning 1 to the checker which says that we will not be in the same directory with the code anymore. This will allow us to check for database.txt files in directories too. And then, we will call myfunct again recursively with our new path. All of the above will apply for our new path recursively so no file or directory can escape from the corrector.