GEBZE TECHNICAL UNIVERSITY
COMPUTER ENGINEERING

SYSTEM PROGRAMMING LECTURE

HOMEWORK 1

REPORT 1

AHMET FURKAN KURBAN

1801042674

MARCH 18

SOLUTION OF THE PROBLEM

First of all, I divided the input entered into the terminal according to semicolons and tried to keep the before and after each semicolon in a double-dimensional array, and I sent each element of this array to the functions to be processed in while. After sending the character array, which is split according to the semicolon in each index, to the function, I split the string according to the slash and will change in the file, the string values to be replaced and the project-specific characters such as $, ^ , i separately, and then sent them to the function to make the following cases.

A case : First of all, I created a new array with malloc, after obtaining str1 and str2 separately, I walked around in the text and compared it with str1. When it is match, I saved the character as long as the length of str2 in my new array. If there was no match, I saved the first element of my text to the array I just opened and continued to compare it from the next character.

B case : I first reduced the big words in the str1 input to match the upper and lower case characters. Then I checked the comparisons I made in case A by adding + 32 to the text character, if it matched, I changed it.

C case : I first separated them by semicolons and saved them in an array, as I explained in the beginning of the report.

D case : While I was navigating over the input entered for the d state, if I came across the [character, I proceeded through while until I came across the] character, and at this time, I continued to compare the values in it with the character in my text, if there was a match, I set the control variable to 1. and progressed until input was '/0'. In case of any mismatch, I exited by making a break and did not change, otherwise I continued by saving the str2 input to the array.

E case : For this case, if a char does not match in the first word I search, I exit directly with break and do not check other words until \n. I repeat the same things again in the next line.

F case : For this case, when I saw a dollar sign only, it was to delete the dollar sign from the str1 input and assign \n to it, so the basic replacement, which works under normal conditions, worked.

G case : I scrolled through input str1 until I saw an asterisk. After seeing the asterisk, I checked how many times the previous character was repeated in the text. I advanced the text that far and continued to compare from the next character. If the previous value is], I compared the value that was previously match in slash with my text and tried to move forward in my text. I replaced it with str2 when it was match. When there was no match, I saved the character in the text and continued to compare.

I used the codes we saw in the lesson for file operations. I did the same in the lock operation.

Design Decisions

In terms of design, my primary task was to break down the inputs given at the beginning of the project according to certain symbols. Then I did the parts of whether the entered arguments are sufficient and reading from the file. After these, I saw that for string manipulation, firstly, star and brackets form the main body in the project, and symbols such as case sensitive , $ and ^ can be easily combined with this main body later on. I tried to fulfill what was requested and made it final.

Requirements that I Achieved :

→ Case a , b , c, d , e , f , g working separately and together.

→ There is no memory leak (controlled with Valgrind) .

→ File I/O process is working.

→ Lock process is working.

Requirements that I Failed :

→ There is bug sometimes when cases is working together.

→ Since I haven't written so much code for a long time, the code structure and writing did not appeal to the eye.