Project Report

by
Josh Moore, Maria Colon, Theint Nwe Nyein, Leo Goldstein

Abstract

In this paper, an explanation of the assignment and program is given. Throughout the paper, the different aspects and difficulties of the program are also explained with their respective solutions that our group were able to produce. In general, the paper below recaps all the details of what was asked, what we did and how we did it.

Introduction

The program is to be designed in such a way that the user inputs vary strings while the program matches the inputted values against a dictionary table. The input can also contain different formatting that the program must recognize and act accordingly. Should the table contain the first word of the String input, then a displaying of the found String along with its classification are performed. Should the input not exist within the table, then a different display will be the outcome.

The program is designed to continuously ask the user for input until a specific keyword is given, that being 'exit'. This only applies if the keyword is the first word within the logical line, meaning this word itself can exit within the line without having the program exit, but is required at the beginning to terminate.

Process

For this project, being relatively simple in design, needed little external files other than the main file of the program. In this program, the user inputs their desired collection of characters and the program will incorporate such characters into what's called a "logical line". This line has the possibility of formatting, and as such needs processing to get what the current line means.

Different formats are implemented and need to be accounted for, such as the Pound Symbol (#). Having this within a logical line converts the information following the pound symbol as a 'comment', meaning the inputted characters can be ignored. Another formatting handler that was necessary was the 'new line' format. This would make the program accept both the current line and the following line as a single-set logical line. From this, the line would be processed and ran against the table within the program.

To properly handle these formats, the program contains metadata that splits the entire line by its characters. Following this, the first formatting handling is whitespace; ensuring that the input is no more than your single-space characters between inputted strings. After

completion of re-designing the logical line, the next format to check for is the Pound Symbol. Since this can be anywhere in the program, a dynamic search is needed to find the symbol. If found, then the preceding string input is kept while the rest is removed from the logical line. Should this, however, start at the beginning of the line, then the entirety of the line is ignored, and the program asks the user for another input.

Additionally, after all the previous processing has occurred, the ending of the line is not a comment and possess a ‘new-line’ keyword, then the program accepts the current line and combines it with the next following line(s) that the user enters. After the user has finished inputting all the characters and lines, this line is processed as the entire logical line; as stated previously, this will perform the processing and handling.

Conclusion

To complete, the program works as intended with requirements. There is, however, a small issue that is not addressed in the assignment that appears with the program. If the user inputs a comment as their input, then the program will take that empty string and still compare it. When trying to quickly fix the issue, the program would break. As such, the program will say the empty string is external, but performs all the actions correctly.

Appendix

Code of Program

josmoor@tesla:~/SP24/Operating_Systems/Project_1$ make

g++ main.cpp -o run

./run

ALPHA $ cat

cat: short

ALPHA $ CAT

CAT: short

ALPHA $ cAt

cAt: short

ALPHA $ dog

dog: external

ALPHA $ first Sentence\n

>>> Second Sentence

first: external

ALPHA $ part#comment

part: external

ALPHA $ cat#dog

cat: short

ALPHA $ cat# No New Line\n

cat: short

ALPHA $ exit

josmoor@tesla:~/SP24/Operating_Systems/Project_1$