**Project 1: Unix/Linux Command Line Interpreter**

Asher Shores, Philip Varkey

College of Science, Engineering, and Technology, Grand Canyon University

CST-315: Operating Systems Lecture & Lab

Dr. Ricardo Citro

6 February 2022

**Objectives**:

1. Learn the basics of Unix/Linux programming.
2. Review C programming.
3. Explore shell functionality.
4. Implement basic process management.

**Project Description:**

In this project, implement a *command**line interpreter*as a first step towards developing a *shell***.**In many ways, the command line interpreter is a mini shell. The working CLI will demonstrate functionality for a number of commands. Returning to prompt after each completion.

**Methodology/Approach:**

The approach to this project involved creating a single cpp file to host the code for running the program and allowing for command functionality. The basic goal is to have a console interface in which the user can prompt character specific commands (from a set list) that are fed through a constantly running main loop. The commands are executed and the evidence for such is printed and the user is returned to the prompt.

Several screenshots are included throughout demonstrating these processes.

**Algorithm:**

As discussed above, the program is run entirely through one main.cpp file. Once the program initializes, it enters a while(true) loop that can only be broken via quit command. The while loop creates several buffers and prompts the user for command input which is then processed to remove spaces and determine if multiple commands have been entered. If so, there is a nested for loop to iterate through every command as separated via semicolons. Then, in the main line, the program runs through potential matches to the input and finds and executes the appropriate command(s). If no matches are found to the user inputted string, the program outputs a command not found catch and returns to the beginning of the while loop.

**Key Code Segments:**

A screenshot of a computer

Description automatically generated with medium confidence

Initialization and beginning of main loop

Text

Description automatically generated

If statements to test user input for command matches

Text

Description automatically generated

Additional if statements and the catch-all and return to beginning of loop(s).

Text

Description automatically generated

Example of executed code

**GitHub Link:**

[**https://github.com/asherShores5/Project-1-Unix-Linux-Command-Line-Interpreter-**](https://github.com/asherShores5/Project-1-Unix-Linux-Command-Line-Interpreter-)

References

Brennan, S. (2015, January 16). *Tutorial - Write a Shell in C - Stephen Brennan*. Stephen Brennan’s Blog. Retrieved February 4, 2022, from https://brennan.io/2015/01/16/write-a-shell-in-c/

GeeksforGeeks. (2020, June 7). *Making your own Linux Shell in C*. Retrieved February 4, 2022, from https://www.geeksforgeeks.org/making-linux-shell-c/