**My Shell Documentation**

**Matthew Melendez**

**SUNY University at Albany**

**ICSI 412 – Operating Systems**

**System Documentation**

High-level Data Flow Diagram

Diagram

Description automatically generated

List/Routines and Their Brief Description

* void pipe\_execute(char \*\*commands, int n\_commands, char \*outfile);
  + This helper function is responsible for handling multiple arguments using pipes.
    - char \*\*commands – array of commands
    - int n\_commands – the number of commands
    - char \*outfile – the file for redirection

Implementation

To implement my Shell, I used the slides provided, stack overflow, emailing professor for help with pipes, and classmates advice.

**Test Documentation**

How I Tested My Program

The way I tested my program was the test cases provided in the Project 1 pdf. I also tested my program running ‘ls’, but I didn’t screenshot it since it wasn’t a required test case.

Additionally, I ran “cat country.txt city.txt” to test basic commands. However, I didn’t screenshot since it wasn’t required. The test cases below also test “cat country.txt city.txt”, but instead sends to a file.

I also ran “cat country.txt city.txt | egrep g” to test my pipes but didn’t screenshot since it wasn’t required. Also, the test cases below already test piping.

List of Test Sets

* cat country.txt city.txt | egrep g | sort | more > countryCitygSorted.txt

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

The only issue that I had with this test case is that my program has a weird issue where the “MyShPrompt>” doesn’t return to terminal. However, the program is still able to get input from the terminal but prints the output to the file specified by the User.

* cat country.txt city.txt | egrep ‘g’ | sort | wc -l > countryCitygCount.txt

Graphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

I have the same exact issue with this test case. I realized that this issue only happens with redirection but not piping.

**User Documentation**

How To Run My Program

To run my program, you just must compile the code using “gcc myShell.c -o mySh”. After it is compiled, enter the following,”. /mySh”, in the terminal. This will run the compiled program and now you’re in the Shell.

One important thing to mention is that if you want to fix the glitch with terminal outputting to a file instead of terminal, restart program.

Parameters

The parameters for the program are the same as the Linux terminal but there are two key differences. One is that my implementation of Shell can only run 10 commands whereas Linux terminal can run more than 10.

An additional parameter for my implementation of Shell is that egrep has an issue with anything in quotes or apostrophes whereas Linux terminal doesn’t. Therefore, when using egrep or grep, put the parameter without quotes or apostrophes.