CMPE322

OPERATING SYSTEMS

PROJECT 1

STUDENT

EYMEN ESAD ÇELİKTÜRK

2020400165

SUBMISSION DATE

13.12.2023

1 Introduction

MyShell is a Unix shell implementation written in C. It provides a command-line interface where users can execute commands, create aliases, and perform basic system operations. The shell supports various features, including command history, background processing, and output redirection.

2 Compilation

To use this makefile, save it as a file named makefile (or Makefile) in the same directory as your myshell.c. Open a terminal and run make to compile the myshell executable. Use make clean to remove the compiled executable.

3 Structure

parseInput(char* input, char* commandArgs):**

This function takes a user-input string and parses it into an array of strings representing the command and its arguments. It tokenizes the input string using spaces as delimiters, allocates memory for an array of strings, and populates it with the parsed tokens. The resulting array of strings is then terminated with a NULL pointer.

saveAliasesToFile():

The saveAliasesToFile function writes the current aliases stored in the AliasList data structure to a file named "aliases.txt". It opens the file in write mode, iterates through the aliases, and writes them in the format "alias = command" to the file. This function is crucial for persisting aliases across different sessions.

addAlias(char* alias, char* command):

The addAlias function adds a new alias and its corresponding command to the AliasList. It checks if the maximum number of aliases has been reached and prints an error message if so. If not, it copies the provided alias and command into the respective arrays within the AliasList and increments the alias count.

loadAliasesFromFile():

The loadAliasesFromFile function reads aliases and their corresponding commands from the "aliases.txt" file and adds them to the AliasList. It prevents duplicate aliases by checking if an alias already exists in the list before adding it. This function is called during the initialization of the shell to load previously defined aliases.

substituteAlias(char* input, char* commandArgs):**

This function checks if the given input corresponds to an alias in the AliasList. If a match is found, it substitutes the alias with its associated command in the provided commandArgs array. It then parses the substituted command and returns a success flag. If no alias is found, it returns a failure flag.

executeCommand(char* command, char args, int background, char* outputFile, int append, int isAlias, int invertOrder):**

The executeCommand function serves as a most critical component in the MyShell implementation, responsible for executing user commands, handling various features such as background execution, output redirection, and the specialized ">>>" (invert order) functionality. Upon receiving a command and its arguments, the function initiates a fork to create a child process. In the child process, it first checks for output redirection; if an output file is specified, the standard output is redirected to that file. Subsequently, it either directly executes the command or, in the case of invert order, sets up a pipeline between the child and a grandchild process to reverse the output before displaying it.

For background execution, the function forks another process to run the command independently, allowing the shell prompt to regain responsiveness. The function also manages the history file, appending each executed command along with its arguments for future reference.

The handling of aliases is incorporated into this function, where it distinguishes between a regular command and an alias. If an alias is encountered, the associated command is executed in place. If the command involves invert order or background execution, the function adapts accordingly, creating additional processes and managing pipes for communication.

getLastCommand(char* lastCommand):

The getLastCommand function reads the last executed command from the ".myshell_history" file and stores it in the provided lastCommand buffer. This information is utilized in the "bello" command to display the last executed command.

belloCommand(int processCount):

The belloCommand function displays various system-related information, including username, hostname, last executed command, TTY, current shell name, home location, current time and date, and the number of processes being executed. It also handles the cleanup of terminated child processes (zombies).

main():

The main function is the entry point of the MyShell program. It contains the main loop for accepting user input, parsing and executing commands, handling aliases, and managing the overall flow of the shell. It also includes the "exit" command to terminate the shell and the "bello" command for displaying system information.

4 Difficulties Encountered

After implementing the core functionalities of MyShell, I dedicated considerable attention to addressing edge cases, such as scenarios involving ">>>&" or aliases with redirection.