Matthew Lieberman

mrl196

Extensions used:

3.2 Home Directory – if a user does ‘cd ~/Desktop’ the program knows that ~=$HOME

3.3 Directory Wildcards – If a user were to call a script, they could just run ‘./mysh */*.sh’ to find a .sh file and run

Here's a summary of each function in this shell program:

1. execute_builtin_command: This function handles built-in commands such as cd, pwd, echo, and ls. For example, if the user types "cd /home/user", this function changes the current directory to /home/user.
2. search_file: This function searches for a file with a given filename in a list of directories. If it finds an executable file with the specified name, it returns the full path to that file.
3. expand_wildcards: This function expands wildcards in command-line arguments. For example, if the user types "./*mysh */*.sh", this function expands the wildcard "*.sh" to a list of matching files and updates the argument list accordingly.
4. execute_command: This function executes a command by forking a new process and running the command in the child process. It handles various features of shell commands, such as input/output redirection, piping, and wildcard expansion. It also calls search_file to find the executable file for the command.

To test the functionality of this shell program, we have the following test plan:

1. Test basic commands: Try commands like ls, pwd, echo, cd, and exit to ensure that the program executes them correctly.
2. Test input/output redirection: Test redirection by using a command like "echo 'Hello, world!' > output.txt" to ensure that the program redirects output to a file as expected.
3. Test wildcard expansion: Test wildcard expansion by using commands like "./mysh */*.sh" or "./mysh */*/*.sh" to see if the program can expand wildcards correctly, and travel through directories and subdirectories.
4. Test environment variables: Test environment variables by using commands like "echo $PATH" or "echo $HOME" to see if the program can read and display these variables.
5. Test error handling: Test error handling by trying invalid commands, arguments, or syntax to see if the program handles them.
6. Test batch mode: Run a script containing a series of commands to ensure that the program can run in batch mode and execute multiple commands in sequence.