Shell

February 22, 2020

1 Shell

Implement a basic shell. Your shell implementation should not execute the existing shell program. You have to build these functionalities by yourself (using fork, exec, wait, etc., as discussed in the class). By default, the shell program waits for user input from the stdin. After the user enters some command, the shell program parses the input to interpret I/O redirection, pipe, etc. After interpreting the command as described below, the shell program again waits for user input until the user enters the exit command. Below is the list of features you need to implement.

Syntax	Meaning
command	execute the command and wait for the command to finish, print error message if the command is invalid
command > filename	redirect stdout to file filename. If the file does not exist create one, otherwise, overwrite the existing file
command >> filename	If the filename already exists append the stdout output, otherwise, create a new file
1>filename	redirect stdout to filename
2>filename	redirect stderr to filename
2>&1	redirect stderr to stdout
command < filename	use file descriptor 0 (stdin) for filename. If command tries to read from stdin, effectively it will read from filename.
|	pipe command (as discussed in class)
exit	exit from the shell program

2 Implementation

Implement everything in the "shell.c" file. Your program should be able to handle the nested commands, e.g.,

```
‘‘/bin/ls | /bin/sort | /bin/uniq | /usr/bin/wc -l 2>&1 1>output.txt’’
```

3 References

Read the man page of `pipe`, `fork`, `read`, `write`, `open`, `close`, `dup`, `exec`, and `wait` for more details about these system calls.

4 Design documentation

You also have to submit design documentation along with your implementation. In your design documentation, explain which part of your shell program will get invoked when you execute "/bin/ls | /bin/sort | /bin/uniq" command. If you are using a Linux distribution where `ls`, `sort`, and `uniq` commands are not present in the `/bin` folder, then find the correct paths and use them to execute the command. Write the pseudocode of all components of your program that will execute during the execution of the above program. The pseudocode should be a high-level description of your actual submission. Your assignment will not be graded if your design documentation is incomplete, or your pseudocode is different from your actual implementation.

5 How to submit.

To be done individually. Submit a zip folder that contains two files: "shell.c" and design documentation (in pdf format). Please make sure that your implementation is not printing any debug messages before submitting the final code. The submission link is on backpack.