ITEC 371: Operating Systems

Project 1: RUSH – The RU SHell

In doing this project, abide by the RU Honor Code or you will fail the assignment and the course. This is not a group assignment

---

Description

For this project, you will build upon your binary file system from project 1. You will create an interactive shell that can be used to query information about the binary file system, or add to it. Your program takes a single command line argument, the binary file to use for the shell session. When the program starts it is located in the root directory for the binary file system, displays a prompt, waits for user input, reports on user input, and repeats until stopped by a command. Note: if your program is started with a file name that does not exist, your shell should create an empty binary files system that will be stored in that file.

Commands the RUSH shell responds to

pwd – Print the absolute path of the current directory from the root directory
ls – List the names of all of the files and directories stored in the current directory, include the type (file, directory, etc). Also list the name of the directory.
cd <dirname> - Change the current working directory to the specified directory
cd .. – Change the current working directory to the previous working directory (or ignore if in the root director)
mkdir <name> - Create a directory in the current working directory
cat <filename> - Print the contents of the text file to the screen or print out invalid file type
createFile <filename> - Create a text file or program of the particular name, and prompt the user for its contents (Note the .t or .p must be included at the end of the filename)
run <program> - For now, print out running program <name>
start <program> - For now, print out starting program <name>
step <program> - For now, print out stepping in program <name>
printInfo – Print out the information produced after the quit command in Project 1
quit – Stop the execution of the RUSH shell and return to the linux shell.

Sample usage scenario (on an empty binary file called sample.bin)

```
./RUSH sample.bin
EnterCommand>ls
Directory Name:  root
EnterCommand>mkdir test
EnterCommand>cd test
EnterCommand>pwd
Current directory is root/test
EnterCommand>createFile test.t
Enter file contents>A test file
```

```
EnterCommand>ls
Directory Name:  test
Filename:  test.t Type:  Text file
EnterCommand>cat test.t
Text file contents:
A test file
EnterCommand>cd ..
EnterCommand>pwd
Current directory is root
EnterCommand>quit
```

Note: your code must account for extraneous spaces or newlines. If it does not handle these input cases, a grade deduction will be made. Your program will be tested by inputting the commands listed earlier into a file, then redirected into your program via ./NAME < inputFile . If your program does not work in this manner, you will not receive a passing grade on the project.

Submission requirements:

Include all of your files in a directory name project2_RUEmailLogin_ along with a Makefile that can build your program and produce the RUSH executable. Tar and gzip this directory into the file project2_RUEMailLogin.tar.gz and submit it to D2L.

When I untar the file with tar xfzv project2_RUEMailLogin.tar.gz I expect one directory to be produced (project2_RUEMailLogin_) that contains your source code and a Makefile that will build your program (i.e. I will type make in that directory and it should produce a program called RUSH).

Your submission will be tested on rucs.radford.edu. Please make sure it works on that system before you submit to D2L. I will not accept programs that only work on your machine and not on rucs.radford.edu.

Rubric

Submission requirements: If not met, you receive a 0 for your submission. Homework/Project 1 should have provided feedback necessary to successfully submit this project.

Runtime:

To get a C on your submission it must take the input from the specification and

produce the output from the specification (albeit it cannot be hardcoded).

To get a B on your submission it must produce correct output for at least ½ of the

additional test cases I will use on your program.

To get an A on your submission it must produce correct output for all of the additional test cases I will use on your program.

Design: Can lower your grade by one letter. Deductions are made for choices such as all of your code in one function. Use what you have learned from previous courses and you should have no issues with this deduction.

Comments: Can lower your grade by one letter if your submission doesn't contain comments on:

Variables – purpose, type, scope

Function headers (purpose, parameters, return type, called by / calls)

Explanation for blocks of code (inline comments)

Readme file for program containing name, what it is, how to use it.

Header for every source file containing purpose and functions / classes contained / where to start looking.

Note: If you score a B or higher on this assignment, I will replace your Project 1 score with your Project 2 score (and this will not count as your one substitution for the semester per the syllabus). For those who scored a B or above on P1, you can use the substitution policy in the syllabus to replace any grade with any other grade in the same category (i.e. you can substitute P1's grade for P4's grade). Again, the previous substitution policy can only be used by those who scored a B or higher on P1.