Operating Systems

Session: July-Dec 2020

LAB Assignment-2 : System Programming Part-I

Deadline for Submission: 12th September, 2020, Satday, Midnight (Hard)

Q. 1. Create your own SHELL-commands to simulate (mimic) the functionality of the well-known SHELL commands:-

- **A.** mycp *source-file destination-file*
- **B.** mycp *file(1) file(2)... file(n) destination-directory*
- **C.** mymv *file(1) file(2)... file(n) destination-directory*
- **D.** myhead [-n] *count filename*
- **E.** mytail [-n] *count filename*
- **F.** mycut [-d] "*delimiter*" [-f] *field-number filename*
- **G.** mypaste [-d] "*delimiter*" *file(1) file(2)*
- **H.** mypaste [-s] *file(1) file(2)*
- **I.** mygrep *'string' filename*
- **J.** mywc [-l] [-w] [-c] *filename*

The above commands should act exactly the same way as the corresponding SHELL command does. For example, in question no. A, 'mycp' should act as the 'cp' command:-

```
$  ./mycp /tmp/file1.txt /etc/file2.txt
        same as
$  cp /tmp/file1.txt /etc/file2.txt
```

However, a SHELL command may be more powerful than your simulated command in the sense that a SHELL command may take several optional arguments which your simulated command may not accept. You are required only to implement those options which have been mentioned in your question. For example, in question no. D, you are required to implement 'myhead' command only with -n option, although the corresponding 'head' command can take several other options like -c, -q, -v etc.

But, please note that if your command takes an option, the user running the command may or may not give that option (just like in the corresponding command). For example, in question no. D, the user can run the command in both of the following two ways:-

```
$ ./myhead /tmp/file1.txt
        Or
$ ./myhead -n 5 /tmp/file1.txt
```

In both cases, your command should work correctly and as expected.

NOTE-1: You have to implement your commands using only C/C++ programs. <u>You cannot use any SHELL script</u> to implement the commands.

NOTE-2: In order to do file-I/O operations (excluding I/O with standard input-output devices), you MUST use System Calls, NOT C library functions.

Q. 2. Write SHELL scripts inside which you should execute, along with other SHELL commands, your own command, which you have already created in the previous question.

A. Write a SHELL Script which will take a source pathname/filename and a destination pathname/filename from the user, check whether the source file and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, copies the source file to the specified destination using your **mycp** command.

B. Write a SHELL Script which will take multiple pathname/filename-s as source files, and one destination pathname from the user, check whether the source files and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives all correct inputs, copies all the source files to the specified destination directory using your **mycp** command.

C. Write a SHELL Script which will take multiple pathname/filename as source files, and one destination pathname from the user, check whether the source files and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives all correct inputs, moves all the source files to the specified destination directory using your **mymv** command.

D. Write a SHELL Script which takes a source pathname/filename and a destination pathname from the user, checks whether the source file and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, asks the user to input a value for count. If the count is 10 (default value for the SHELL 'head' command), it executes your **myhead** command without the -n option; otherwise executes the **myhead** command with -n option with the specified value of count on the input file. The output of the execution is not displayed on the terminal; rather, it is redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

E. Write a SHELL Script which takes a source pathname/filename and a destination pathname from the user, checks whether the source file and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, asks the user to input a value for count. If the count is 10 (default value for the SHELL 'head' command), it executes your **mytail** command without the -n option; otherwise executes the **mytail** command with -n option with the specified value of count on the input file. The output of the execution is not displayed on the terminal; rather, it is redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

F. Write a SHELL Script which takes a source pathname/filename and a destination pathname from the user, checks whether the source file and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, asks the user to input values for a delimiter and a field-number. It then executes your **mycut** command with the -d and -f options with the specified values of delimiter and field-number on the input file. The output of the execution is not displayed on the terminal; rather, it is redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

G. Write a SHELL Script which takes two pathname/filename-s for two source files and a destination pathname from the user, checks whether the two source files and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, asks the user to input the value for a delimiter. It then executes your **mypaste** command with the -d option with the specified delimiter value on the input files. The output of the execution is not displayed on the terminal; rather, it is redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

H. Write a SHELL Script which takes two pathname/filename-s for two source files and a destination pathname from the user, checks whether the two source files and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, executes your **mypaste** command with the -s option on the input files. The output of the execution is not displayed on the terminal; rather, it is redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

I. Write a SHELL Script which takes a source pathname/filename and a destination pathname from the user, checks whether the source file and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, asks the user to input a string. It then executes your **mygrep** command with the specified string on the specified file. The output of the execution is not displayed on the terminal; rather, it is redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

J. Write a SHELL Script which takes a source pathname/filename and a destination pathname from the user, checks whether the source file and the destination pathname (directory) exists. If not, prompts to the user to correctly provide the inputs. When it receives correct inputs, executes your **mywc** command with all the -l, -w and -c options on the specified file. The output of the executions are not displayed on the terminal; rather, redirected to a newly created file called output which is placed in the destination directory as mentioned by the user.

NOTE: Please listen to my video lectures on System Programming (specifically the last part) before trying these assignments.

What you have to Submit?

1. The program files (one C, one bash script)
2. Sample Input and Output (Screenshots)
3. A PPT explaining your programs with proper flow chart etc. Each slide of the PPT must include an Audio clip describing the content of that slide.
4. Comments for the evaluator (TA), if any, in PDF format

How to Submit?

Mail all the above mentioned documents by the given deadline to the TA-incharge **Mr. FAHIEM (PCL2015006).**

Q. No. to Group-ID mapping:-

Group-ID	Q. No.		Group-ID	Q. No.
1	1A, 2A		10	1J, 2J
2	1B, 2B		11	1B, 2B
3	1C, 2C		12	1C, 2C
4	1D, 2D		13	1D, 2D
5	1E, 2E		14	1E, 2E
6	1F, 2F		15	1F, 2F
7	1G, 2G		16	1G, 2G
8	1H, 2H		17	1H, 2H
9	1I, 2I		18	1I, 2I
			19	1J, 2J