cse15l-lab-reports

Competition Task (from Lab Report 4) Using A Bash Script

Inspired by the prompts provided. For lab report 4 (about doing commands quickly), how could you have done the task very quickly by writing a bash script (even if that was against the rules)

Lab Report 4

Link to My Lab Report 4 My Strategy was to copy paste a series of commands from a Google Doc (or any convenient note-taking program). So, essentially bending the rules. In practice, I was able to do it within around 15 seconds as I did not have to wait to press Enter after every single command. Visual Studio Code's built in terminal executed multiple different-line commands at once. However, one drawback was that I had to split up the commands into multiple blocks because if I copy-pasted all the commands at once, it did not work. The terminal lagged out and did not execute any of them. It was not evident whether it was the terminal's problem or ieng6's. Regardless, it could not be optimized and did have the limitation of my typing speed.

Writing the Bash Script

While choosing this prompt, it seemed obvious that this would be among the easier ones. However, I did not account for editing the document from the command line. I had no idea how that would work. So, taking advantage of this course's ChatGPT policy, I asked ChatGPT.

ChatGPT Prompt 1

```
How do I edit text files from using a bash script?
```

ChatGPT's Response 1

To edit text files using a bash script, you can use one of several command-line text editors available on most Unix-based systems. Here are a few examples:

1. Nano: Nano is a simple and easy-to-use command-line text editor. You can use it to edit text files by specifying the file name as an argument. For example, to edit a file called "example.txt", you can run the following command:

   ```
   nano example.txt
   ```

   This will open the file in the nano text editor. You can then make any changes you want, save the file by pressing `Ctrl+O`, and exit the editor by pressing `Ctrl+X`.

2. Vim: Vim is a more advanced text editor that offers many powerful features. You can use it to edit text files by running the following command:

   ```
   vim example.txt
   ```

   This will open the file in the vim text editor. You can then make any changes you want, save the file by typing `:w` and pressing `Enter`, and exit the editor by typing `:q` and pressing `Enter`. If you make changes and want to save and exit at the same time, you can type `:wq` and press `Enter`.

3. Emacs: Emacs is another powerful text editor that can be used from the command line. You can use it to edit text files by running the following command:

   ```
   emacs example.txt
   ```

   This will open the file in the emacs text editor. You can then make any changes you want, save the file by typing `Ctrl+x Ctrl+s`, and exit the editor by typing `Ctrl+x Ctrl+c`.

To automate the editing of text files from a bash script, you can use the command-line options of the text editor you choose to use. For example, to edit a file with vim in a script and save it without any user interaction, you can use the following command:

```
vim -c ":wq" example.txt
```

This will open the file in vim, save it, and quit the editor without any user interaction. You can replace "example.txt" with the name of the file you want to edit.

Overall Answer 1

I think the answer ChatGPT gave was very redundant giving multiple possibilities. However, some of them were not applicable as it would not work in a bash script. The last part of the response was nice, though, because it gave me the answer I was looking for (the Automation part). It pointed me in the right direction - using `vim -c :wq`. So, the logical next step was asking it to expand on the automating process.

ChatGPT Prompt 2

```
Can you expand on automating the editing of text files?
```

ChatGPT's Response 2

Sure, I'd be happy to expand on automating the editing of text files using a bash script.

When editing text files using a text editor from the command line, you can use various command-line options and features to automate the editing process. Here are some examples of how you can automate the editing of text files using different command-line text editors:

1. Nano: To edit a file with nano and save it without user interaction, you can use the following command:

   ```
   echo "new text" > example.txt && nano example.txt -S
   ```

   This command first adds the text "new text" to the file "example.txt" using the echo command, and then opens the file in nano with the `-S` option. The `-S` option tells nano to save the file without prompting the user for confirmation. Once you're done editing the file, you can simply exit nano and the changes will be saved automatically.

2. Vim: To edit a file with vim and save it without user interaction, you can use the following command:

   ```
   echo "new text" > example.txt && vim -c ":wq" example.txt
   ```

   This command first adds the text "new text" to the file "example.txt" using the `echo` command, and then opens the file in vim with the `-c` option. The `-c` option tells vim to run the command `:wq` (write and quit) automatically. Once you're done editing the file, you can simply exit vim and the changes will be saved automatically.

3. Sed: Sed is a stream editor that can be used to edit text files non-interactively. To replace a string in a file using sed, you can use the following command:

   ```
   sed -i 's/old text/new text/g' example.txt
   ```

   This command replaces all occurrences of "old text" with "new text" in the file "example.txt" using the `-i` option, which tells sed to edit the file in place. These are just a few examples of how you can automate the editing of text files using a bash script. The possibilities are endless, and you can use any command-line text editor or tool that fits your needs.

Overall Answer 2

This answer was useful, but did not mention how to edit it automatically. The response assumed creating a new file and then editing it, which might be useful as an example. Lastly, I needed to learn how to edit it at a specific location. So, I asked ChatGPT yet again.

ChatGPT Prompt 3

```
Similarly, how do I edit a file at a particular location?
```

ChatGPT Response 3

To edit a file at a particular location, you can use various command-line text editors and tools available on most Unix-based systems. Here are some examples of how you can edit a file at a particular location using different command-line text editors:

4. Nano: To edit a file at a particular location using nano, you can use the following command:

   ```
   nano +line_number,column_number file_name
   ```

   This command opens the file "file_name" in nano and places the cursor at the specified location, which is specified by "line_number" and "column_number". For example, to open "example.txt" and place the cursor at line 10, column 5, you can use the following command:

   ```
   nano +10,5 example.txt
   ```

5. Vim: To edit a file at a particular location using vim, you can use the following command:

   ```
   vim +line_number file_name
   ```

   This command opens the file "file_name" in vim and places the cursor at the specified line number, which is specified by "line_number". For example, to open "example.txt" and place the cursor at line 10, you can use the following command:

   ```
   vim +10 example.txt
   ```

6. Sed: To edit a file at a particular location using sed, you can use the following command:

   ```
   sed -i 'line_numberss/old_text/new_text/g' file_name
   ```

This command replaces all occurrences of “old_text” with “new_text” in the specified line range “line_numbers” of the file “file_name”. For example, to replace “old_text” with “new_text” in lines 10-20 of “example.txt”, you can use the following command:

```
sed -i '10,20s/old_text/new_text/g' example.txt
```

These are just a few examples of how you can edit a file at a particular location using different command-line text editors and tools. The specific syntax and options may vary depending on the tool you use, so make sure to refer to the documentation or help files for the tool you are using.

Overall Answer 3

Trusting its accuracy, I will use `vim` with the `-c` and `+line_number file_name` modifiers.

Trying to Edit a File using vim and the options using a Bash script

On my local computer (to avoid messing with the host copy of the previous lab report), I will try to find the location where the changes need to be made. Using VisualStudioCode, I found the location of where the error is at: line 43 col 12, where it is `index1` instead of `index2`. So, I could go to line 43 and col 13, enter insert mode and use backspace to delete the character, and insert `index2 += 1;`. So, I need to formulate a command to open vim at line 43. Trusting ChatGPT, I tried:

```
vim +43 ListExamples.java
```

It worked as expected. However, I could not figure out how to do the same, with a specific column. I used google, and found this command -

```
vim "+call cursor(<Line>, <Col>)"
```

So, using this, I ran -

```
vim "+call cursos(43,13)" ListExamples.java
```

It worked as expected. Now, I needed to enter insert mode, input `backspace` and `2`. However, after countless hours (2 to be precise), I found that it was very difficult to save and exit using vim as the in-line text editor. Eventually, I ended up using `sed` because it was very straightforward and the text substitution could be done easily.

```
sed -i.bu '43 s/index1/index2/' ListExamples.java
```

The `-i` flag means insert. The `.bu` part means a backup file is made without the changes made (necessary because of different `sed` versions). The '43 ___' means to edit only on line 43. The 's/' means to substitute. The first / signals the word to search for and the second one means which one to replace with. The last argument is the file to be edited. So, I successfully found a way to edit the file in a way comaptible with a bash scipt.

Actually Creating the Bash Script

I had the commands I ran previously in a Google Doc (as part of my strategy). And now, I just had to populate all the comamnds into the bash file. Create the bash script file using `vim` and call is `a.sh` because it's easy to type.

```
vim a.sh
```

Then, copy paste all these lines into a.sh

```
ssh cs15lwi23aja@ieng6.ucsd.edu
git clone git@github.com:adityaparmarr/lab7.git
cd lab7
javac -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar *.java
java -cp .:lib/hamcrest-core-1.3.jar:lib/junit-4.13.2.jar org.junit.runner.JUnitCo
sed -i.bu '43 s/index1/index2/' ListExamples.java
git add ListExamples.java
git commit -m "a"
git push
```

Then, finally, just run the bash scipt -

```
bash a.sh
```