CS 213: System Software Lab

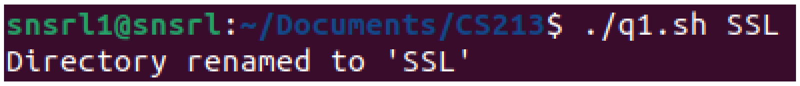
Autumn 2024, IIT Dharwad

Exercise#2 Questions

Topic: Linux and Bash

---

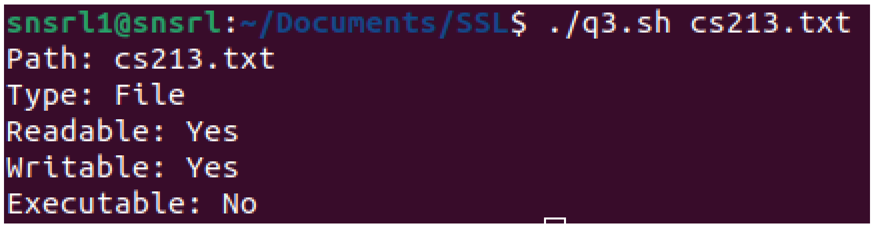
1. Write a shell script to rename the current working directory with the given name
Steps: (i) Create directory name CS213 within the script; (ii) Check if a new name is provided as a command-line argument if not, throw an exception; (iii) Check if the directory was created successfully if not, throw an exception; (iv) Rename the directory with the command line provided name as shown in below figure.



```
snsrl1@snsrl:~/Documents/CS213$ ./q1.sh SSL
Directory renamed to 'SSL'
```

2. Create a bash script to assess the status of a given file or directory. The script should offer insights into various attributes of the provided path, including its existence, type (file or directory), readability, writability, and executability. The script should take the path as an argument and provide informative responses about these attributes.

   Steps: (i) Check if the input arguments are not equal to 1, throw an exception; (ii) Check if the provided input is file or not, if not throw an exception; (iii) Check if the file exit or not, if not throw an exception; (iv) Check for read (-r), write (-w) and executable (-w), if file has permissions and you can echo them accordingly.

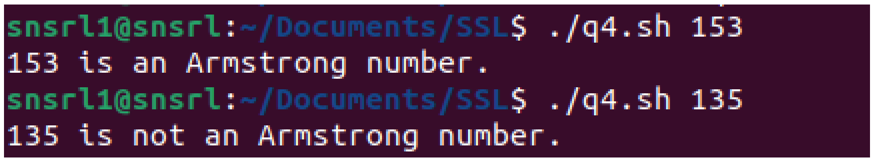


```
snsrl1@snsrl:~/Documents/SSL$ ./q3.sh cs213.txt
Path: cs213.txt
Type: File
Readable: Yes
Writable: Yes
Executable: No
```

3. Write a bash script to determine whether a given number is an Armstrong number. Your script should take the provided number as input and determine whether that number is an Armstrong number or not

Note: An Armstrong number, also known as a narcissistic number or pluperfect digital invariant, is a number that is equal to the sum of its own digits raised to the power of the number of digits.
For instance, 153 is an Armstrong number because 1^3 + 5^3 + 3^3 = 153.

Steps: (i) Check if the input arguments are not equal to 1, throw an exception; (ii) Check how to get the number of digits in the given number by using special variables; (iii) Use loops/any other mechanism to do the above specified operation

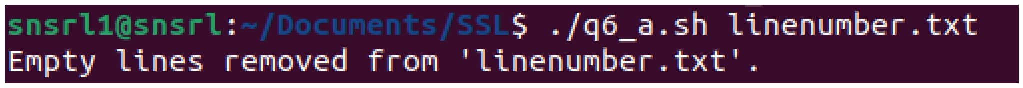


```
snsrl1@snsrl:~/Documents/SSL$ ./q4.sh 153
153 is an Armstrong number.
snsrl1@snsrl:~/Documents/SSL$ ./q4.sh 135
135 is not an Armstrong number.
```

4. Write a script to perform the following operations on a text file that contains both non-empty and empty lines.
   (a) Delete the empty line from the file
   (b) Delete the non-empty lines from the file

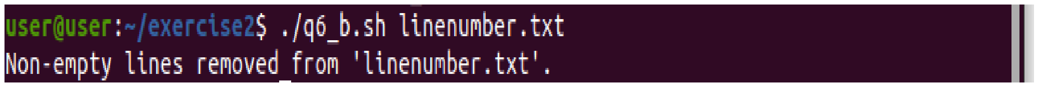
   Steps: (i) Check if the input arguments are not equal to 1, throw an exception; (ii) Check if the provided input is file or not, if not, throw an exception; (iii) To check if the file contains empty line and to remove for this you could explore “sed” utility and you can use redirection operators

   (a)



```
snsrl1@snsrl:~/Documents/SSL$ ./q6_a.sh linenumber.txt
Empty lines removed from 'linenumber.txt'.
```

   (b)



```
user@user:~/exercise2$ ./q6_b.sh linenumber.txt
Non-empty lines removed from 'linenumber.txt'.
```

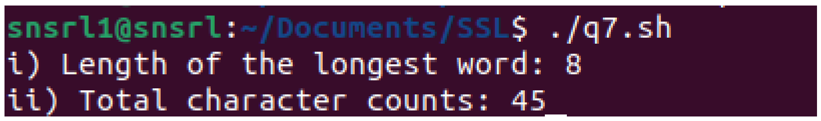
5. Create a new file, *f3.txt,* using nano. Write each in a single line sequentially
   12
   one
   three

Four
1
Laptop
fortyone
seventy

Answer the following questions using bash program code with respect to *f3.txt*:

i) Write the code to get the length of the longest word. Verify the answer by manual calculation.

ii) Write the code to get the character counts. Verify that 2 + 3 + 5 + 4 + 1 + 6 + 8 + 7 + 8(total no.of words) + 1 (End of File)) = 45

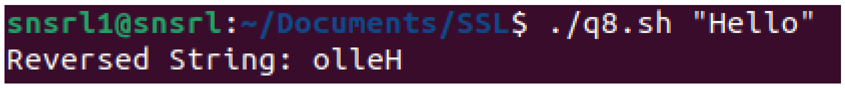


```
snsrl1@snsrl:~/Documents/SSL$ ./q7.sh
i) Length of the longest word: 8
ii) Total character counts: 45
```

6. Write a script that takes a string as input and determines its reverse.
   Steps: (i) Check if the input arguments are not equal to 1, throw an exception; (ii) You can explore "rev" utility in the script to reverse the provided string character by character

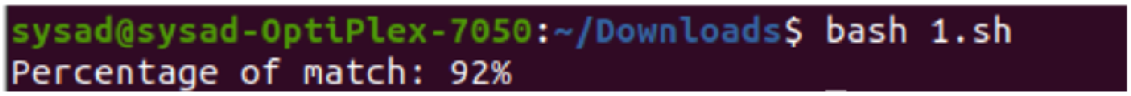
   Sample Input: Hello, World!

   Sample Output: !dlroW ,olleH



```
snsrl1@snsrl:~/Documents/SSL$ ./q8.sh "Hello"
Reversed String: olleH
```

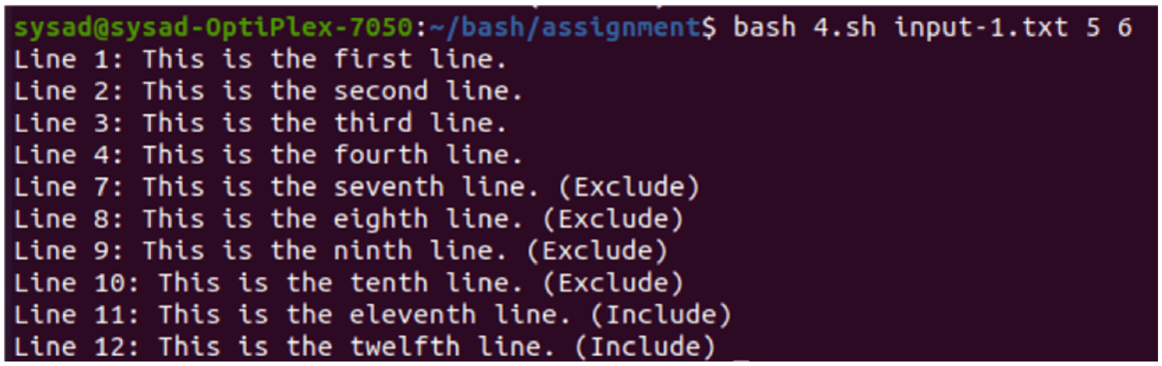
7. Write a shell script to generate two text files containing Fibonacci and prime series within the range of 0 to 500. Compare contents of both the files using the "diff" command, and display the percentage of similarity between them.



```
sysad@sysad-OptiPlex-7050:~/Downloads$ bash 1.sh
Percentage of match: 92%
```

8. Create a shell script that prints the entire content of a file while excluding the specified range of lines.
   [Note: Utilize the file we provided named “input-1.txt”.]

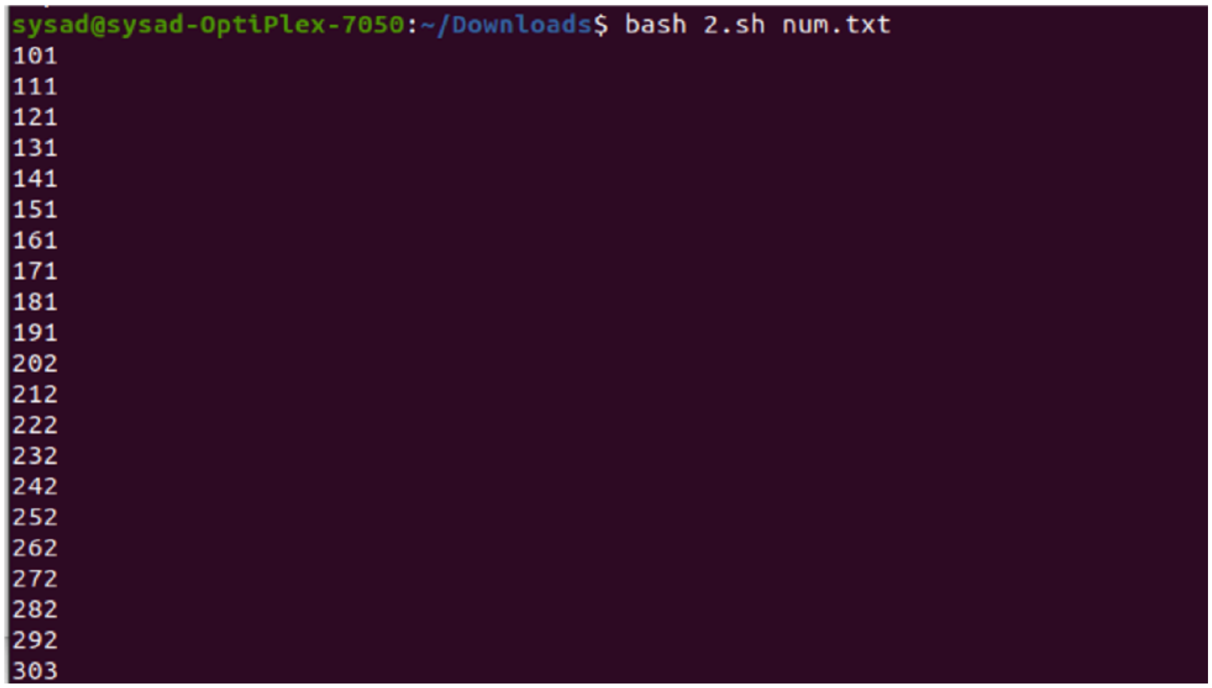
   Steps: (i) Check for the correct number of arguments if not, throw an exception; (ii) Check if the input file exists if not, throw an exception; (iii) Check if start_line and end_line are valid integers if not, throw an exception; (iv) Use sed to print the entire content of the file except for the specified range



```
sysad@sysad-OptiPlex-7050:~/bash/assignment$ bash 4.sh input-1.txt 5 6
Line 1: This is the first line.
Line 2: This is the second line.
Line 3: This is the third line.
Line 4: This is the fourth line.
Line 7: This is the seventh line. (Exclude)
Line 8: This is the eighth line. (Exclude)
Line 9: This is the ninth line. (Exclude)
Line 10: This is the tenth line. (Exclude)
Line 11: This is the eleventh line. (Include)
Line 12: This is the twelfth line. (Include)
```

9. Create a shell script that employs the “awk” command to extract and display all palindrome numbers from a specified “num.txt” file.
   [Note: Utilize the file we provided named “num.txt”.]

   Steps: (i) Check if the input file exists, if not, throw an exception; (ii) Use awk to print all palindrome numbers



```
sysad@sysad-OptiPlex-7050:~/Downloads$ bash 2.sh num.txt
101
111
121
131
141
151
161
171
181
191
202
212
222
232
242
252
262
272
282
292
303
```