Assignment 3: Bash Scripting
IT Workshop - 1

- Deadline for submission is: **9:00 PM, 23 October 2017.**

- Submission of this assignment will be via a Git repository on ***GitLab***. GitLab is similar to GitHub, except it allows you to create private repositories.

- Submission details:
    - Create an account on **gitlab.com** if you don't already have one.
    - Create a **private** repository on gitlab, so that other's can't see your code, with the name "**Assignment3-ITWS**".
    - Give the TA user **itws.ta** "**master**" access to this repository.
      How to: (https://stackoverflow.com/questions/31908222/giving-access-to-private-gitlab-repository)
    - In your repository, there should be at most 15 files:
        - If not specified, Each question should have a script file with name format: q<n>.sh
          "q1.sh" for question 1, for example.
        - There should also be a readme file in your repository, with any additional information.
        - Do not add any other files or directories in the repository.

- **It is compulsory to do in Bash**. If you also attempt these in any other language other than Bash/C , you will get Bonus marks.

- Make sure to give the **TA user account access to your repository**. Otherwise, we will not be able to access it and evaluate your work, leading to a 0 in this assignment for you.

- **Plagiarism will earn you a straight ZERO in this assignment. Automatic plagiarism detection scripts will be run on all your submissions.**

1) Given a scrambled word, check if it exists as an executable command. If yes, print the allowed arguments of the command
Print the 1st valid occurrence with arguments.
If there are multiple valid commands for the scrambled word, You may print that as well but it is not mandatory.
Format:

- **Input =>**
  - <Scrambled Word>
- **Output =>**
  - <Command_Name>
  - Arguments =>
  - <Flag>
  - <Function>

Example:
➜ bash q1.sh
Input => tac
Output => yes
cat
Arguments =>
-A, --show-all
equivalent to -vET

-b, --number-nonblank
number nonempty output lines, overrides -n

-e equivalent to -vE

-E, --show-ends
display $ at end of each line

-n, --number
number all output lines

-s, --squeeze-blank
suppress repeated empty output lines

-t equivalent to -vT

-T, --show-tabs
display TAB characters as ^I

-u (ignored)

-v, --show-nonprinting
use ^ and M- notation, except for LFD and TAB

--help display this help and exit

--version
output version information and exit

➜ bash q1.sh
Input => hy
Output => No

2) List the commands that have been used on your terminal in the descending order of usage. The result should contain command name and the count. If some commands have same count, print any order.
Format:
Input => None
Output =>
<Command Name> <Number of TIme>
Let suppose you use cat 2 times, gedit 5 times, echo 10 times
➜ bash q2.sh
Output =>
echo 10
gedit 5
cat 2

3) Have a crontab that checks the history of your profile and cleans it up every 1st Monday of the month.
Format: submit whatever you have done as crontab operation submit it in crontab_q3

4) Write a script that checks if Crontab is properly formatted.
Cr min hr dom mon dow CMD
-> cat crontab_file
30 08 10 06 * /home/ramesh/full-backup
-> bash q4.sh crontab_file
Yes
-> cat crontab_file2
30 08 10 cat * /home/ramesh/full-backup
-> bash q4.sh crontab_file2
No

5). Write a script that takes 2 file names as input arguments (use these 2 files only so that we are consistent in terms of result) and merges the data and then sort the data based on the nth column passed as an argument. There can be more than 1 argument for sorting, that would mean sort the file by first argument and then followed by 2nd argument and so on for all arguments that are given after the file names.
If there are multiple answers, print any.
Format=> bash q5.sh file1 file2 <column_no1> <column_no2>
file1=>

5 4 2 1
2 1 3 4
2 10 11 401

file2 =>

3 8 7 6
2 1 5 5
8 10 22 33

➜ bash q5.sh file1 file2 1 2
output=>

2 1 3 4
2 1 5 5
2 10 11 401
3 8 7 6
5 4 2 1
8 10 22 33

Explanation:
after merging files:

5 4 2 1
2 1 3 4
2 10 11 401
3 8 7 6
2 1 5 5
8 10 22 33

after sorting by 1st column:

2 1 3 4
2 10 11 401
2 1 5 5
3 8 7 6
5 4 2 1
8 10 22 33

after sorting by 2nd column for the numbers which are same in 1st column: [2 2 2], it will check its 2nd column [1 10 1] which in sort is [1 1 10] so the final output is:

2 1 3 4

2 1 5 5
2 10 11 401
3 8 7 6
5 4 2 1
8 10 22 33

So, firstly input is sorted by 1st argument, where the numbers are same, it checks by column given in next argument.

6). Write a script that takes a list as an argument and flattens the data in the list. Any type of data can be inside list example doubles, numbers etc.
E.g (1 (2 (3 4 5)))) ==> (1 2 3 4 5)
➜ bash q6.sh
Example: Input => Output

1. (1 (2) ((3)) (5) (((7 8 9)) ()) ==> (1 2 3 5 7 8 9)
2. ((1 2)(3 (4))) ==> (1 2 3 4)
3. ((1 1.2)()(3.5 4)) ==> (1 1.2 3.5 4)

7) Take a string and check if the value is a palindrome. Output Yes / No.
➜ bash q7.sh

1. madam => Yes
2. Madam => Yes (M is upper case but still palindrome)
3. max => No

8) Write a program that provides an interface for read, write, update and delete a record in the file. The record contains Employee Number, Name and Salary. Identify the duplicate records in a file.For duplicate, Check whether full record is same or not with case sensitive.
Format: Employee data in file name employee.txt, and program.sh is your script name
Read

- **Input**
  - **Read by employee number -> bash program.sh read eno <no>**
  - **Read by employee name -> bash program.sh read ename <name>**
  - **Read by employee salary -> bash program.sh read esalary <salary>**
- **Output**
  - **eno,ename,esalary**

Write

- **Input**
  - **bash program.sh write <eno> <ename> <esalary>**
- **Output**
  - **Done**

Update only by eno

- **Input**
  - **bash program.sh update <eno> <ename> <esalary>**

- **Output**
    - **Done**

Delete only by eno

- **Input**
    - **bash program.sh delete <eno>**
- **Output**
    - **Done**

Example:

➜ cat employee.txt
eno,ename,salary
1,Shivam,1001
2,Rahul,999
6,Ramesh,500
➜ bash program.sh write 9 Arjun 800
Done
➜ cat employee.txt
1,Shivam,1001
2,Rahul,999
6,Ramesh,500
9,Arjun,800
➜ bash program.sh read eno 6
6,Ramesh,500
➜ bash program.sh read name Ramesh
6,Ramesh,500
update only by eno
➜ bash program.sh update 6 Arjun 700
Done
➜ cat employee.txt
1,Shivam,1001
2,Rahul,999
6,Arjun,700
9,Arjun,800
➜ bash program.sh delete 6
Done
➜ cat employee.txt
1,Shivam,1001
2,Rahul,999
9,Arjun,800
➜ bash program.sh write 9 Rahul 700
Done
➜ cat employee.txt
1,Shivam,1001
2,Rahul,999

6,Arjun,700
9,Arjun,800
9,Rahul,700
➜ bash program.sh write 9 Rahul 700
Done
➜ bash program.sh write 9 rahul 700
Done
➜ bash program.sh write 9 Shyam 1700
Done
➜ cat employee.txt
1,Shivam,1001
2,Rahul,999
6,Arjun,700
9,Arjun,800
9,Rahul,700
9,Rahul,700
9,rahul,700
9,Shyam,1700
➜ bash program.sh duplicate
9,Rahul,700

9) Identify the nth highest salary in the file. If the input is '10', it should print employee number, name and salary of all employee(s) that have 10th highest salary.
Format: use the same script of question 8 program.sh and implements in that.
nthsalary

- **Input**
  - **bash program.sh nthsalary <no>**
- **Output**
  - **Multiple rows <eno,ename,esalary> seperated by newline**

Example:
➜ cat employee.txt
1,Shivam,1001
2,Rahul,999
3,Arjun,999
4,XYZ,2000
5,ABC,1234
➜ bash program.sh nthsalary 2
5,ABC,1234
➜ bash program.sh nthsalary 4
2,Rahul,999
3,Arjun,999

10) Write a program that can take any number of arguments and performs an exponential value of that number.

➜ bash q10.sh 2 3 4
4096
Explanation => 2^3^4 => 8^4 => 4096
➜ bash q10.sh 2 3 4 5
1152921504606846976

11) Build a Basic calculator that takes operator and "n" operands for basic arithmetic operations

There will be 1 operation and multiple operands. Allowed Operations are +, -, *, /.
All operations are from left to right. Use 4 precision for decimal numbers.

Format:
<Operation Type>
<Number of operands>
<Operand>
<Operand>
<Operand>
.
.

Input:

+
5
2
20
35
7
12

Output:

76

Input:

*
5
1
2
3
4
5

Output:

120

Input:

/
3
4
2
3

Output:

0.6667

Input:

-
3
4
2
7

Output:

-5

12) Takes a date in Word Format (it can be Oct, October, etc.. Standard month and date nomenclature) and asks for the date format to be printed - MM/DD/YYYY or DD-MM-YYYY) and then prints in the numeric format. Also check whether it is a valid date or not. Example: feb 29, 2019 is not a valid date. [Print"Invalid"]
Format: first 3 characters of the month name or complete month name. eg: jan or january
Input=>

january 21, 2016

Output=>

01/21/2016
21-01-2016

13) Create a cronjob that checks for every alternate month 3rd Wednesday 8:23 PM for finding files that have been modified in the last 3 months and then does a back-up into a different server (mirage account)
Format: crontab_q13

14) Create a HTML file that contains a table of all directories and files in home directory of user account.

Format: q14.html and q14.sh that opens your html file in firefox
Example:

ITWS		
Name	**Size**	**File/Dir**
Lab1	30 K	Dir
Lab2	50 K	Dir
help.txt	100 K	File
ITWS\Lab1		
Name	**Size**	**File/Dir**
commands.txt	15 K	File
unix.txt	15K	File
ITWS\Lab2		
Name	**Size**	**File/Dir**
bash.txt	50 K	File

15) Download at least 2 IIIT webpages and sort the words based on their frequency.

Format:

- **q15_url.txt that contains webpage address separated by newline.**
- **q15_ans.txt that contains sorted word in descending order with frequency**
- **q15.sh that executes your program and take 2 webpages url as argument and 3rd argument as file_name in which you want to store the output.**
- **word with same frequency should be printed in lexicographical order in ascending order.**

Output Format:

<word> <frequency>

example of q15_ans.txt:

iiit 25

india 25

hyderabad 12

college 5