IIIT Hyderabad
Computer & Scripting Environment
Assignment 4 - Shell scripts

IIIT-H

Assignment 4 - Shell scripts

Practice questions

These questions are optional and for your practice only. You do not need to submit solution to these questions on courses website.

1. Take two numbers and print their sum
2. Take number and check it is even or odd
3. Take number and print number of significant digits in number. For example if user enters 00234 print 3.
4. Write script to print table of 2 in below format:

   ```
   2 * 1 = 2
   2 * 2 = 4
   ...
   2 * 10 = 20
   ```

5. Write script that takes a number as command line argument and prints its table in format used in (4) above.
6. Display following menu to user:
   ```
   (A) Add
   (B) Subtract
   (C) Multiply
   (D) Divide
   (E) Modulus
   (F) Exponentiation
   ```
   Then ask user for choice (A-F). After taking users choice ask user for two numbers and perform chosen operation on those two numbers.
7. Add choice `(G) Exit` in above menu. The program should keep asking for choice / numbers unless user chooses ‘G’ or ‘g’ for exit. This time support both ‘a’ or ‘A’ for Add, ‘B’ or ‘b’ for Subtract, etc.

8. Take word and file from user as argument and display '`File <filename> contains <word>`' or '`File <filename> does not contains <word>`' based on whether file actually has the given word or not. You should replace <filename> and <word> in above messages with actual filename and word, supplied by user.

   You can assume that word will not contain any special characters. It will only have alphabets or digits. Print error message when user gives less or more number of arguments than what is required for script to work.

9. Write script that takes any number of files / folders as command-line argument and creates tar file containing all those folders and files. The name of tar file should be based on current date and time. Use following format for naming tar file - `YYYY-mm-dd-HH-mm-ss.tar`. For example 2011-09-27-18-43-45.tar

   You must support more than nine command line arguments. (Hint: Use `shift`)

10. Write script that takes one filename as argument. The file given as command line argument contains list of absolute path of files, one on each line. (You can assume there are no spaces in path.) Touch all the files whose names are specified.

    You must not pass more than one argument to touch anywhere in script. Here argument does not refers to shell variable, here argument refers to number of arguments given to touch. Hence `touch $A` where `A='a.txt b.txt'` is not allowed. You are also not allowed to use xargs anywhere in the script.

Assignment questions

Since courses website will allow only one file submission, you have to 'tar' programs together and upload the final tar file as solution.

1. Take numerator and denominator for a fraction from user. Then print the same fraction with lowest possible integer numerator and denominator such that both fractions are same.

   For example if user enters 4, 10 then output should be 2, 5 as $\frac{4}{10} = \frac{2}{5}$. Similarly if user enters 3, 9 then output should be 1, 3 as $\frac{3}{9} = \frac{1}{3}$.

2. Again take numerator and denominator from user. In this case if numerator is less than denominator then print smallest possible integer numerator and denominator for same fraction. However if numerator is bigger than denominator then print mixed fractions where fraction part again should have smallest possible integer numerator and denominator.

   For example if user enters 44, 10 then output should be 4, 2, 5 as $\frac{44}{10} = 4\frac{2}{5}$. Similarly if user enters 5, 3 then output should be 1, 2, 3 as $\frac{5}{3} = 1\frac{2}{3}$.

   (Hint: Try to reuse code written for problem (1) above.

3. Take number 'n' from user and print n^{th} prime number. Note that 2 is first prime number.

4. Write function factorial which calculates factorial using recursion. Use the function in a script which takes number whose factorial needs to be calculated from user.

5. Take a number 'n' from user and print diamond made of '*' on screen based on n. For example for n=2 display:

```
 *
***
 *
```

   Similarly for n=3 display:

```
  *
 ***
*****
 ***
  *
```

You are required to infer relation between n and shape printed based on above examples.

6. Take word and number (n) from user and then display diamond of size n made by using word given by user. For example if user gives values hello and 2 then display:

   ```
      hello
   hello hello
      hello
   ```

   You can do small adjustments for words with even length to achieve something that looks like diamond.

7. Write script that takes one filename as argument. The file given as command line argument contains list of absolute path of files, one on each line. Do not assume that there are no spaces in path. Touch all the files whose names are specified.

   You must not pass more than one argument to touch anywhere in script. Here argument does not refers to shell variable, here argument refers to number of arguments given to touch. Hence `touch $A` where `A='a.txt b.txt'` is not allowed. You are also not allowed to use xargs anywhere in the script.

8. Create list of all `.sh` files in current folder. Check first line of each file in this list and if file **starts with** '`#!/bin/sh`' then replace that with '`#!/bin/bash`'.

   Only modify files that have '`#!/bin/sh`', do not blindly replace first line of all `.sh` files with '`#!/bin/bash`'. Also the replacement needs to be done only to starting sequence. If file has '`#!/bin/sh`' somewhere in between do not replace that. You do not have to search for `.sh` recursively in sub-folders, do replacement only in files in current folder.

9. Write script that takes two arguments <search> and <replace>. The script should replaces all occurrences of first argument in all text files in current folder with second argument.

   You can assume both <search> and <replace> are alphanumeric without any special characters. You do not have to search recursively in sub-folders, do replacement only in files in current folder.

   Your program must not modify itself while doing <search> and <replace>. For example if your program/script is called with arguments '`echo` `printf`' then it should replace all echo's with printf's in script itself. Hence program has to be intelligent enough to detect that file it is operating on is itself and avoid modifying it. You can use trivial and not absolutely perfect ways of detecting whether some file is same as script being run or not. However the better the test, the better is the program.

(Hint: Looks at hash functions / checksums like md5sum, sha1sum. Trying to use hash functions without thinking them through may lead to chicken/egg problem.)

10. For a series of symbolically linked files, find out what is the final actual file being referred to. For example if folder has:

    ```
    a.txt --> b.txt
    b.txt --> c.txt
    c.txt --> d.txt
    d.txt --> e.txt
    ```

    then any input to your program among '`a.txt`', '`b.txt`', '`c.txt`', '`d.txt`' and '`e.txt`' must give output as '`e.txt`'.

11. Solve problem (10) without making any assumptions on location of links. For example in this problem links can point to locations like '`../a/../../b.txt`', '`/home/user/Desktop/c.txt`' etc.

    You can output '`/home/user/Desktop/../../a.txt`' instead of '`/home/a.txt`'. In other words you do not have to normalize the output path.

12. Write script that takes filename as argument. The file given as argument would contain absolute path of files / folders in current system to be backed up. Create folder in current working directory with name `YYYY-mm-dd-HH-mm-ss` and create copy of all folders and files with ownership, timestamp, permissions and full path information in created folder.

    For example if argument is '`backup.txt`' and `backup.txt` contains:

    ```
    /etc/hosts
    /var/lib/mysql
    ```

    then create folder '`2011-09-27-18-52-41`' and create sub-folder '`etc`' and sub-tree '`var/lib/mysql`' inside folder '`2011-09-27-18-52-41`'. Then copy file '`/etc/hosts`' and complete contents of folder '`/var/lib/mysql`' to newly created '`etc`' and '`mysql`' folders. This is all assuming user tried to run script file for backup at 06:52:41pm on 27th September, 2011.

You can use above script after doing assignment for actually taking backup of your files. You can automate daily backup by calling script using cron. Once you automate calling of script you will soon require another script to delete very older copies of backup leaving just last 3/4 copies.

The Linux system also grew like this where people created scripts and later on additional scripts to support/extend previous ones. Hence actual system scripts are very good learning points for learning how to organize/write scripts.

13. Create set of scripts to implement 'trash' functionality on command line. Requirements for these set of scripts are:

    - All scripts should take parameter from config file
    - Config file should get stored in folder `~/.my_trash`
    - Name of config file should be '`my_trash.conf`'
    - Config file should one parameter 'Location' specified as:

      ```
      Location=~/.my_trash/deleted_items
      ```

    - All deleted files should be stored / restored / listed using Location parameter specified in config file.
    - While using any script if config folder or file do not exist, they should get created. Default value of Location in config file should be `~/.my_trash/deleted_items`
    - There should be script named '`mytrash-list.sh`' which lists deleted items in following manner:

      ```
      01.  abc
      02.  a.conf
      03.  b.txt
      ```

      Note preceding zeros in above output to make output neater. (Will your script work if more than 100 or 1000 files are present in Location folder?)
    - There should be script named '`mytrash-delete.sh`' which takes any number of arguments and moves all the files / folders specified as command line arguments to folder specified in Location

in my_trash config file. Nicely display error message if some error occurs due to missing files, lack of permissions, etc.

- There should be script named '`mytrash-restore.sh`' which takes any number of arguments and restores all those files / folders from Location specified in configuration file to current folder. Nicely display error message if some error occurs due to missing files, lack of permissions etc.
- You can assume user will not delete two files / folders with same name. (I hope you realize that this is huge assumption).
- In case of missing files / permission problems you should not abort at first missing file / permission problem. You should anyway process all arguments and print error messages for files that could not be deleted/restored.
- Scripts should support more than 9 arguments. (Hint: You can use `shift`) to handle more than 9 arguments.

Hint: You anyway have to divide program into at least three files. You might as well create few more supporting programs and source them using dot ('.') so that number of functions / lines of code in a single file is manageable. This would also allow reuse of code.

For example there is not much point in maintaining three copies of code that verifies that config file and folder exist. Similarly there is not much point in keeping three copies of code that can read location. Trying to do this without using functions may prove suicidal given that you are required to print very good and information error messages.

Challenge questions

These are for ambitious students only. It is not required for everyone to solve these. You do not have to upload solutions to these questions on courses website:

1. A file contains list of roll numbers of students who copied assignment from each other. Write program which reads the file and groups of all students together who copied assignments. For example, if file has:

   ```
   300106040    300294658
   300106040    300107059
   300107059    300294658
   300107059    400605824
   400605824    300107059
   300109527    300109529
   300109527    300642821
   300106040    300754628
   300892421    300992421
   ```

   then output would be:

   ```
   * 300892421 300992421
   * 300106040 300107059 300294658 400605824 300754628
   * 300109527 300109529 300642821
   ```

   If solving this in shell script is a problem, then give it a try in language of your choice. This belongs to category of problems known as 'connectivity problems'. If roll numbers are treated as vertices of graphs and set of roll numbers as edge between two vertices. Then the problem is asking for groups of list of all connected vertices.

2. Find all hard-links of givenn fiel without using '`find`' or any similar command which can directly find hard-linked files. Your script should search entire partition for files which are hard-linked to given file.

3. Solve assignment problem (11) such that output paths are normalized. You should show both absolute path and relative path of actual file which is being referred. Relative path should be with respect to current working directory.

4. Extend my_trash scripts created in assignment problem (13) so that they support following additional features:

- The config file should take two more parameters 'Email' and 'Size'
- If size of deleted items exceeds size then files deleted earlier can be removed permanently until size of deleted items is under specified Size. Email should go to specified Email when some files are deleted permanently.
- Size can be absolute 10GB, 500MB etc. or relative 10%. In case of relative use partition size as reference.
- Support deletion of more than one file / folder with same name.

Note that all thre three / four requirements specified above are non-trivial. You should be able to appreciate that adding any of these to assignment problem would make assignment problem very complex.