LINUX LAB 4: BASH SCRIPTING

Prepared by Jack G. Zheng for IT 4423

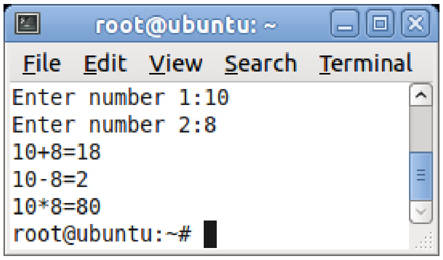
This exercise will let you practice Bash shell script basics.

INSTRUCTIONS

1. First of all, go through all examples in the lecture notes and the example source files to get familiar with Bash script basics.

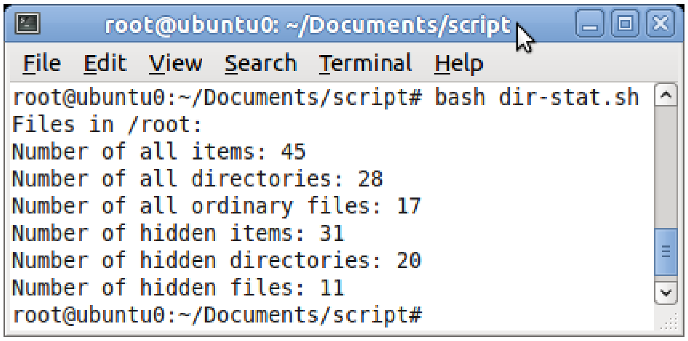
2. Write a script to check command arguments (3 arguments maximum). Display the argument one by one. If there is no argument provided, remind users about the mistake. If there an easy way to use a loop to get all arguments?

3. Practice a simple command line calculator. See the sample output on the right.
   - Use the read command twice to prompt users to enter two integer numbers. Save them to two variables.
   - Display the sum, difference, and product of these two numbers. - Hint: use $[ ] to force numeric calculation; example: $[$num1+$num2]



```
Enter number 1:10
Enter number 2:8
10+8=18
10-8=2
10*8=80
root@ubuntu:~#
```

4. Write a script to do a little file count report for a given directory, as shown in the following figure.
   a) You may use a command argument to supply the directory you want to report
   b) Hint: use ls, grep, wc, regex, and pipe "|"
   c) You may see that scripts can help do something that is usually not provided by the OS or any utilities.


root@ubuntu0: ~/Documents/script
File Edit View Search Terminal Help

```
root@ubuntu0:~/Documents/script# bash dir-stat.sh
Files in /root:
Number of all items: 45
Number of all directories: 28
Number of all ordinary files: 17
Number of hidden items: 31
Number of hidden directories: 20
Number of hidden files: 11
root@ubuntu0:~/Documents/script#
```

5. Write a script to determine if a number is even or odd.

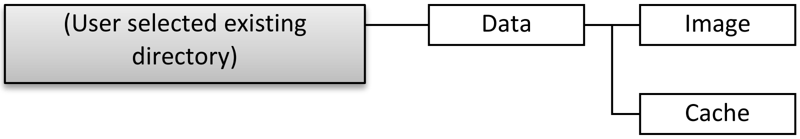
6. Write a script to display numbers from 1 to n, where n is an integer provided by users (if not, default to 10). Hint: use "read" command to accept user input.

7. Write a script to display the following patterns on the screen. Number of rows and columns are taken from the command arguments; if they are missing, set default to 3 (rows) and 4 (columns). Hint: you will use a nested loop.

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

8. Create x empty files in a given directory (x is a number), following a naming format like this: myfile1, myfile2, etc. Ask the user to enter the first part file name and the number of files he/she wants to create. Hint: you may use the “touch” command to quickly create empty files. Take two screenshots:
   a) Display the source code in an editor (**#4-1**)
   b) Execute your script in the terminal, and display the command and the result (**#4-2**)

9. Write a script to create the following directory structure in a directory of user’s choice. The user can supply the directory of choice as an argument; if missing, prompt the user to enter one from the command line. Take two screenshots:
   a) Display the source code in an editor (**#4-3**)
   b) Execute your script in the terminal, and display the command and the result (**#4-4**)



10. Edit the script in the prior question.
    a) Check if the user input (an existing directory) is valid; if not, ask the user to enter it again.
    b) If the “Data” directory exists, then prompt for user actions: cancel or overwrite. Hint: use directory operators; refer to lecture notes slide #8 and #10.

Command line script on one line:

```
read -p "Enter a number: " max; sum=0; for ((i=1; i<=$max; i++)); do let sum+=i; done; echo "The sum from 1 to $max is $sum"
```