Linux – lab 4 - Bash

Your lab reports with notes from previous laborations and related material could be of great help during the laboration. Hint: the book contains good starters.

Prepare as usual by creating a copy of the Ubuntu server virtual machine and start it.

Preparation – read about the following:

shebang/ hashbang ____________________

if ____________________

if-else ____________________

for ____________________

-ne, -eq, -gt, -ge, -lt, -le ____________________

____

-d, -e, -f, -x, -r, -w, -s ____________________

____

while ____________________

variables, assignment and use ____________________

____

$#, $0, $1, … ____________________

____

shift ____________________

1. Create a skeleton for a bash script. To avoid overwriting the template, write protect the file. We also want it to be executable. Save it and use it as a template. Specify the mode:

```
chmod __ __ __ template.sh
```

2. Create a script that prints “Hello world!” to the display.
3. Modify #1 to store “Hello world!” in a variable and then write the variable to the screen.
4. Create a script that asks for the users name and prints “Hello <the users name>! How are you?”. Example (output in italics):

Enter your name: Mattias
Hello Mattias! How are you?

5. Create a script that prints the arguments $0 - $3.
6. Create a script that prints the number of arguments.

7. Create a script that asks for two numbers, a and b, and print “a > b” if a>b or “a < b” otherwise using the if-construct. Example:

```
Enter a: 7
Enter b: 3
a > b
```

8. Update the script in #7 and make it output “a = b” if a = b. Use the if-construct.

9. Create a script that prints all arguments, no matter how many arguments there are. Use a while loop, the “number of arguments”-variable and the shift command. Don’t forget $0! Example:

```
assignment_7.sh this is a “little test”
assignment_7.sh
this
is
a
little test
```

10. Make a script that lists all files and folders in the current folder using a for loop and a suitable wildcard.

11. Update the script so it lists files only in the current folder.

12. Update the script so it lists files according to a wildcard given as an argument to the script. Example:

```
./assignment_12.sh *.sh
assignment_10.sh
assignment_11.sh
assignment_12.sh
```

> Hint: Print the number of arguments. To get a better look at what is happening, run the script #9 with a few different wildcards. Eg. *.sh, *, or something similar.

13. Update the script so it performs md5sum for each file instead of printing the filename to the screen. The syntax for this is

```
md5sum filename
```

You have now written a very useful script that can generate lists of checksums for files only that can be used to detect if the content of a file has changed. (Compare the output from your script and md5sum given the wildcard * and a folder of files and subfolders.)