Shell script III.

Login into the server of the subject (Linux) and create a subdirectory!

Review, scripts, program structure

a) Create a script which calculates the factorial value of a number N! (N!) N is given by a parameter. Be careful with * using in an expression… Check whether you get a parameter or not! (if, $#, for, seq, $1, expr, ``)
b) Create a script which reads in words till you get the word „end”. The words has to be written into a file in alphabetical order! Use a temporary file! (read, while, >>, cat, sort)
c) Create a script which gets several numbers (we do not know how many) through the parameter list (maximum 9 parameters) - the result should be the sum of them! ($*, expr, ``, for)
d) Try to solve the above given task in an other way too! (Tipp: echo $* | cut)
e) Try to solve the above mentioned task without using $*! (Tipp: use SHIFT!)
f) Try to solve the above mentioned task with using parameter lista s an array! (Tipp: ${*[$i]})
g) Modify one of your solutions and check whether the number of parameters are between 1-9! (if, $#)
h) Modify your solution by writing out a small help if there there is no parameter or the first parameter is –help! (if, $#, test, $1)
i) Create a script which gets a file and garbles it into two. Each odd-th lines copy to odd.txt and each of the even-th lines copy into even.txt! The filename is given by a parameter! Check whether you get a parameter or not and check the existence of the file as well! (if, test, while, read, expr, >>)

Tip: To read a file line by line you can use the following structure:

```
while read line
do
  Instructions
done < file
```

j) Write a script which gets a file. The filename is given by a parameter. In each lines of the file there is a number. Writes out the number and a * if it is a prim number!

To read a file line by line you can use the following structure as well!

```
cat file | while read line
do
  Instructions
done
```

k) Explain the main difference between the two solutions! (Tip: try both of them with writing out the result outside of the cycle!)

l) Write a script which will run in the background and checks in every 0.5 minutes whether a given user is logged in! The required login name is given by a parameter. If it is logged in, then stop execution! (sleep, while, test, grep, $1, wc, if)
m) Modify the above written script and check if there is a parameter or not! (if, test, $#)
n) Check if the required user name is a real one or not! (Tip: use getent passwd)
o) Login through two terminal windows in the same time! Create a script which will write a message in each 10 sec to the other terminal window during 1 minutes! (Tip: echo message >/dev/pts/2 ; if the other terminal window is pts/2 or use write <login> command)

Filters: grep, sed, regular expressions

Remember some rule about regular expressions:
[a-z] – means the character set between 'a' and 'z'
^ - means it must fits to the first position, $ - means the same with the last position
+ - means that it iterates once or more, * - means the iteration can be 0 as well
( ) – creates a group
\ - can change the default meaning of a character
There are a lot of rules!
See them e.g. at http://www.regular-expressions.info/tutorial.html

a) Modify our script which can calculate the factorial of a number (N!)! The new version should have be used as a filter too, if it does not get a parameter! (expr, $1, test, if ,$#, for, seq, ``, cat)
b) Create a phone register program which can store names and phone numbers in a text file! In the text file line by line there is a name and a phone number diveded by a ;. Use switches (parameters) to decide what to do with the „database". If you use „-a" you want to add a new name and phone number. If you use –d then it means you want to delete some data form the file. If you give the parameter –s the program should search a person and the given phone numbers! With parameter –c count the number of persons in the register, and at –o writes out the names in alphabetical order! Sometimes a person has several phone numbers! (grep, sort, cut, >, uniq)
c) Modify our script which can add several numbers given by parameters to be able to use as a filter too! ($*, for, expr, ``, if ,cat)
d) Write a script which rename the files given by parameters from small letters to capital! ($*, tr)
e) Modify the above written script to be able to decide whether you want to change from small to capital letter or vica versa! Give the „direction" by a parameter! ($1,if, test, $*, shift, tr)
f) Write a script which gets a lot of coded e-mail addresses like *hugo at inf dot elte dot hu*. The filename is given by a parameter. Transform this e-mail addresses to the standard syntax hugo@inf.elte.hu. The result must be written out on the screen and into a file int he same time! Use temporary file if you need! (sed, cat, tee)
g) Modify the script with checking the existence of the file! (if, test)
h) Modify the script with checking whether the coded e-mail addresses are good or not! The lines can contains characters, numbers and _ besides spaces. In each line must be

something before „ at „, and must be at least one „ dot ” and the end of the line must be at least one character. (In real life it is more complicated to check.) (grep, while, read, wc, if, test)

Finish your work and logout from everywhere!