First Shell Scripts

Login into the server of the subject (Linux)! Create a directory for your work!

Exercises

Remember:
Use an optional Unix text editor (vi, mcedit etc.) for program code writing
or
edit the files on Windows (e.g. Notepad++) and transfer them on to the server!

Give permission for execution to the file – chmod u+x filename

Execute it by typing: *bash filename* or *./filename* *(first line: #!/bin/bash)*

First shell scripts

a) Create the first.sh script, which writes out „Hello”! Remember to switch on execution permission by using chmod! Execute it!

Tip: Use remarks to write readable codes. The syntax of a remark is a # character!

b) Write a script which writes out the text „Hungary”! Give appropriate permissions and execute it!
c) Write a script which writes out the following joke:
Q: What do you get when you cross a fish and drumsticks?
A: Fishsticks.
d) Modify the above written script writing out the following joke in one line:
Q: What do you get when you cross a fish and drumsticks?A: Fishsticks. (echo –n)
e) Create a .profile file with writing out a „Hello” text! (It should be placed in your main directory!) Log in again and check the result!
f) Modify your .profile file to write out the following joke!
Q: What do you call a dog on the beach in the Summer?
A: A hot dog!
Log in again and check the result!
g) Modify your .profile file to write out the actual date and time! Log in again and check the result!
h) Modify your .profile file to change directory to your subdirectory which you have created just for today! Log in again and check the result!

Remember:
Use env command to list the environment variables! Use set to see all of the variables!
Use unset command to delete a variable!

Variables, parameteres, keyboard input

Environment variables

a) Rewrite PATH variable (in command line) with „”!. Try to run e.g. ls command! What happens? (to correct it, log in once more or write back the original path!
b) Add „ . ” (the actual directory) directory to PATH variable! Write it into .profile!
c) Change the PS1 prompt to the text ***hello:***!
d) Make an error while changing the prompt back (e.g. forget to type the ending „). You shall get the value of PS2 as the prompt!
(/home/**your_username**/bin:/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games)
e) Restore your prompt (\u@\h:\w>) or simply log in once more!

User defined variables

Constant values

a) Create an own variable! The variable name should be „*name*” and the value of it the last name of your best friend! Create a simple script in which you greets your friend by writing out a hello and the content of the variable! Run the script! (Do not forget about x permission!) Export the variable at command line and execute the script once more! At the end unset the variable!
b) Create an own variable! The variable name should be „*myluckynumber*” and the value of it should be your lucky number (e.g. 17)! Create a simple script in which you writes out a text „My lucky number is: „ and the content of the variable! Run the script! (Do not forget about x permission!) Export the variable at command line and execute the script once more! At the end unset the variable!
c) Create a script to greet your native country - with using a new variable inside it! The variable name should be „*mycountry”*, the value of it the name of the country. Write ot a greeting formula (eg. Hello everybody in $mycountry). Execute the script! (Do not forget about x permission!) Check whether it is available from command line or not! (dot sourcing!)
d) Create a script with to variables (A=”What do you get when you cross a piece of paper and scissors?”,Q=”Confetti.”) which tells a joke (writes out the variables value). Execute the script!
e) Create a script with to variables (A=” What do you get if you cross a kangaroo and a snake?”,Q=” A jump rope!”) which tells a joke (writes out the variables value). Execute the script!

Expressions, variable assignments

Remember:
Use **expr** command for evaluating a mathematical expression!
Do not forget about the required spaces between the operators and operands (values)!
In the case of an assigment you have to use command substitution!

a) Write a shell script which gives back the sum of 5 and 3! (echo, expr)
b) Modify the above written script and assign the result of the expression to a user defined variable! (echo, expr, ``)
c) Write a shell script which gives back the difference of 5 and 3. Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)
d) Write a shell script which gives back the product of 5 and 3! Firstly write out the result simply, secondly use a custom variable! (echo, expr, \,``)

e) Write a shell script which gives back the result of division 5 by 3! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)
f) Write a shell script which gives back the remainder dividing 5 by 3! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)
g) Write a shell script which gives back the result: 5 + 3 + 2! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)
h) Write a shell script which gives back the result: (5 - 3) * 2! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)
i) Write a shell script which gives back the result: (5 + 3) % 2! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)

Remember:

$0 – the name of the script
$1,$2…$9 – the parameters from the command line in order of typing
$# - the number of parameters
$* - each parameter
$$ - the PID number of the running script

j) Write a shell script which gives back the result: (6 * 3) / 2! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)
k) Write a shell script which gives back the result: (5+3) % 2! Firstly write out the result simply, secondly use a custom variable! (echo, expr, ``)

Parameters

a) Create a script which writes out the number of parameters given in the command line! Execute the script! (echo, $#)
b) Create a new script or modify the above created one! Write out all of the parameters too! Execute the script!
c) Create a new script or modify the above created one! Write out the first parameter! What will happen if there was no parameter given at the command line? Execute the script!
d) Create a new script or modify the above created one! Write out the PID number of the running script! Execute the script!
e) Create a script which writes out everything: the name, the number of parameters, each of the parameter and the PID number!
f) Write a shell script which gives back the quadrat of the first parameter! (echo, expr, $1, \)
g) Write a shell script which gives back the sum of the first and the second parameter! (echo, expr, $1,$2)
h) Write a shell script which gives back the product of the first and the second parameter! (echo, expr, \, $1,$2)
i) Write a shell script which gives back the sum of the first and the second parameter multiplied by the third parameter! ($1+$2)*$3 (echo, expr, $1,$2)

Keyboard input

a) Create a script which reads in a name into *name* variable from the keyboard and then writes out Hello $name! (read, echo)
b) Create a new script! Read in the name of your favourite country into the *country* variable and then write it out on the screen! (read, echo)
c) Modify the above created script to read in your favourite town as well! (read, echo)

d) Create a new script which reads in the user's first and last name and then writes them out on the screen! (read, echo)
e) Create a new script which reads in the user's name and date of birth and then writes them out on the screen! (read, echo)
f) Create a script which reads from the keyboard a number into a user defined variable and writes out it's value multiplied by 2! (echo, expr, read, \)
g) Create a script which reads from the keyboard a number into a user defined variable and writes out it's value multiplied by 2! (echo, expr, read, \)
h) Create a script which reads from the keyboard a number into a user defined variable **x** and writes out the result of the following expression: x*x+2x+3! (echo, expr, read, \)

Pipes, filters (cut, more, head, tail, wc, sort, grep)

Firts,last,more,sort

a) Write a shell script which lists out the first n line of a file. The filename and the number n should be parameters. (cat, first)
b) Write a shell script which lists out the last n line of a file. The filename and the number n should be parameters. (cat, tail)
c) Write a shell script which lists out the first and last n line of a file. The filename and the number n should be parameters. (cat, first, tail)
d) Write a shell script which lists out the first and last n line of a file in an alphabetical order. The filename and the number n should be parameters. (cat, first, last, sort)
e) Write a shell script which lists out the first and last n line of a file in an alphabetical order. The filename and the number n should be parameters. Think of paging as well! (cat, first, last, sort, more)

Wc

a) Write a shell script which counts how many files and directories are in your actual directory! (ls, wc)
b) Write a shell script which counts how many lines are in a file! The filename should be given by a parameter! (cat, wc)
c) Write a shell script which counts how many characters are in a file! The filename should be given by a parameter! (cat, wc)
d) Write a shell script which counts how many users are logged in the server actually! (who, wc)
e) Write a shell script which counts how many characters long is a word given by a parameter! (echo, wc)
f) Write a shell script which counts how many users are logged in the server actually! (getent passwd, wc)

Cut, tee

a) Write a shell script which gives back the first character of a word given by a parameter! (echo, cut)
b) Write a shell script which gives back the first and last parameter of it! (echo, cut, $*,$#,$1)
c) Write a shell script which adds the value of the first and the last parameter! (echo, cut, $*,$#,$1, expr)
d) Write a shell script which lists out only the login name of all of the users! (getent passwd, cut)
e) Write a shell script which lists out only the login name of all of the users! At the same time write the result into a file too! (getent passwd, cut, tee)

f) Write a shell script which lists out only the first word from each line of a file. (There are spaces between the words.) (cat, cut,$1)
g) Write a shell script which lists out only the first word from each line of a file. At the same time write the result into a file too! (There are spaces between the words.) (cat, cut,$1,tee)
h) Write a shell script which lists out the Nth word from each line of a file. (N is a number given by a parameter! The filename is given by a parameter too!) (cat, cut,$1,$2)
i) Write a shell script which lists out the characters between the Nth and Mth position of each line of a file. (N and M are numbers given by parameters! The filename is given by a parameter too!) (cat, cut, $1,$2,$3)

Grep, wc,

a) Write a shell script which gives back that user logins which starts with character „m”! (who, grep)
b) Write a shell script which gives back that user logins which starts with character C! C is a paraméter. (who, grep, $1)
c) Write a shell script which gives back that user logins which contains the characters given in S! S is a paraméter. (who, grep, $1)
d) Write a shell script which counts how many temporarly deleted users are in your server! Temporarly deleted users are signed by ':666:' in the result of getent passwd command. (getent passwd, grep, wc)
e) Write a shell script listing out only that lines of a file which contains a given word! The filename and the word is given by parameters! (cat, $1,$2, grep,wc)
f) Write a shell script which counts how many lines of a file contains a given word! The filename and the word is given by parameters! (cat, $1,$2, grep,wc)

Redirection of input and output

Output redirection, >

a) What does it mean? echo apple >&2
b) Redirect the output to the dust-bin! (`/dev/null` is a special filesystem object that throws away everything written into it)
c) Redirect the errors to the dust-bin! (`/dev/null` is a special filesystem object that

Remember:
> output redirection
>> output redirection with append feature
< input redirection
<< *here* document

throws away everything written into it)
d) What does it mean?
cat <<apple
    <title> Shell script 1. </title>
    <body> My programs: </body>
    </html>
    apple
e) What does it mean? 2>&1
f) What does it mean? 2>/dev/null

Make it funny!

a) Use colors inside the prompt, e.g. typing
```
export PS1=”\033[01;32m\]\u@\h\[\033[01;34m\] \w
\$\[\033[00m\]”
```
you shall get the prompt in blue and green! The ANSI code-table can be read here: http://en.wikipedia.org/wiki/ANSI_escape_code

b) Modify your .profile file with this colour prompt!

Finish your work and logout from everywhere!