Operating System Overview: Unix
EPITA
Homework as replacement for examination
Groupe 1 & 2

You must upload all scripts and answers inside teams before the 30/01/2022 at 23h59.
If you copy the answer of someone else, both persons will get the same mark (0/20).

Exercise 1

Please answer briefly to the following questions.

1.1) What the difference between Unix and Linux?

1.2) What is MINIX? Who is its author of MINIX?

1.3) Who is the author of Linux? What is the relationship between Linux and MINIX?

1.2) Give the command to display the version of your Linux kernel.

1) How to know if your kernel is stable or unstable?

1.3) Give the command to change your working directory to /etc.

2) Which kind of files are inside the /etc directory?

1.5) what is the utility of the options "-*ltr*" in the following command:

3) ls -ltr

1.6) What is the utility of the option "-r" in the following command and what is the difference between "-r" and "-R" in the following commands:

4) cp -r /home/osman/EPITA /home/backup/
5) cp -R /home/osman/EPITA /home/backup

1.7) what is the utility of the option "-n" used in the following command:

6) cat -n /etc/group

1.8) what is the difference between the following commands when applied to ″file.txt″ with initial permissions (-rwxrwxrwx). Give the permissions of ″file.txt″ after each command and explain the difference between both commands (without sequential execution):

7) chmod ugo+rw file.txt
8) chmod 666 file.txt

1.9) what is the difference between options "-a" and "-A" in the following commands:

9) ls -a
10) ls -A

1.10) what is the utility of the option "-p" in the following command:

11) mkdir -p dir1/subdir1/small

1.11) what is the utility of the option "%s" in the following command:

12) date +%s

1.12) Give the commands to:

13) Create a hard link called "hshortcut" for the file "file.txt"
14) Create a soft link called "ssorthcut" for the file "file.txt"

1.13) what is the utility of the options "-*lwc*" in the following command:

15) wc -lwc file.txt

1.14) Give the command to display only the before last line from a given file "file.txt"

1.15) What is the difference between the following commands when applied to ″file.txt″ with initial permissions (-rwxr--r--). Give the permissions of ″file.txt″ after each command and explain the difference between both commands (without sequential execution):

16) chmod ugo+rw file.txt
17) chmod 620 file.txt

1.16) Give 5 different syntaxes to write a loop in bash

1.17) What is the utility of the options "-s", "-n", "-t", "-p" in the following read commands:

1) read -s var
2) read -n 8 var
3) read -t 2 var
4) read -p "Could you please enter the value of the variable :" var

1.18) Explain the following the difference between ″;″, ″||″, ″&&″ in the following commands:

1) ls -l ; mkdir directory; cd directory; touch f{1..9}
2) ls -l file.txt || echo the file does not exist
3) ls -l file.txt && echo the file exist

Exercise 2

Give only the output of the following scripts:

1) cat script1.sh

```
#!/bin/bash
person=jenny
echo $person
echo '$person'
echo "$person"

PS > ./script1.sh
```

2) $cat script2.sh

```
#!/bin/bash
for i in 1 2 3 4 5
do
  echo "Looping ... number $i"
done

PS > ./script2.sh
```

3) $cat script3.sh

```
#!/bin/bash
for var in $(seq 7)
do
  echo -n "$var ... "
  t=$(( 2 * var + 20 ))
  echo $(( t % 7 + 1))
done
echo DONE!

PS > ./script3.sh
```

4) $cat script4.sh

```
#!/bin/bash
user=/home/fred/fredy/Siso/fred7.tar.gz
echo ${#user}
echo ${user1:=FREDO.tar.gz}
echo $user1
echo ${user2:-fredo.tar.gz}
echo $user2
echo ${user:6:4}
echo ${user/fred/jean}
echo ${user//fred/jean}
echo ${user/fred}
echo ${user//fred}
echo ${user#*.}
echo ${user##*/}
echo ${user%/*}
echo ${user%%/*}
echo ${user1^}
echo ${user1^^}
echo ${user1,}
echo ${user1,,}
echo ${user2,}
echo ${user2,,}
echo ${user~}
echo ${user~~}

m1info=(Ahmed Alexandre Alice Bob)
m1info[100]=Charlie
echo $m1info
echo ${m1info[*]}
echo ${#m1info[*]}
echo $m1info[1]
echo ${m1info[2]}
echo ${#m1info[2]}
echo ${!m1info[*]}
echo ${m1info[1]:2:3}
```

```
PS > ./script4.sh
```

5) $cat script5.sh

```
#!/bin/bash
funct()
{
    if [ $# = 0 ]; then
        echo 0
        return
    fi
    total=0

    for i
    do
        total=`expr $total + $i`
    done

    expr $total / $#
}
echo the result of 4, 8 and 21 is `funct 4 8 21`
echo the average of 4 and 14 is `funct 4 14`
echo the average of nothing is `funct`
```

```
PS > ./script5.sh
```

6) $cat script6.sh

```
#!/bin/bash
fonct()
{
        echo "$2 * $2" | bc > $1
        echo "$2 * $2 * $2" | bc >> $1
}
output=/tmp/tmp.$$
for i in 1 2 3 4 5
do
        fonct $output $i
        res1=`head -1 $output`
        res2=`tail -1 $output`
        echo "The res1 of $i is $res1"
        echo "The res2 of $i is $res2"
done
rm -f $output
```

```
PS > ./script6.sh
```

6) $cat script7.sh

```
#!/bin/bash
for i in *.doc
```

```
do
        mv $i ${i%.doc}.txt
done

PS > ./script7.sh
```

Exercise 3

3.1) Write a script bash that displays the following menu:

```
This script randomly generates the lottery numbers for you:
1)  Loto
2)  Euro Million
3)  Exit
Select your game:1
Numbers: 3 9 21 37 42 Lucky:5
Select your game:2
Numbers: 6 9 18 19 49 stars:2 5
Select your game:
```

In Loto, the script generates 5 different random numbers between 1 and 49, and a lucky number between 1 and 10. In EuroMillion, the script generates 5 different random numbers between 1 and 50, and two different numbers (denoted by stars/étoiles) between 1 and 12. The bold characters in previous example are either user input or output.

3.2) Write a bash script that displays the following menu when it is executed:

```
PS1> ./ex5.sh dir1 dir2 fichier1.txt fichier2.txt fichier3.txt
Ce script a pour objectif de réaliser une sauvegarde :
   1)   Avec une extension tar.gz
   2)   Avec une extension tar.bz2
   3)   Avec une extension tar.xz
Votre choix : 3
```

After entering the choice of the user (3 for example), the script will call the associated backup function, with the list of files and directories passed as arguments. You must write 3 functions for each case in the script and put them in separate files (one function by file).

The function will have to create the backup archive (with the command tar), whose name is incremental: backup**x**.tar.xz (with x is the date and time of backup), while keeping the previous backup files.

Do not forget to add a usage function and test the presence of the arguments before displaying the menu.

3.3) Write a script which loops by asking the user to enter a ″mark″ or press the ″q″ key to indicate the end of the entry. The script should save whole marks in a file and display:

a. The number of marks entered
b. The average
c. The maximum value
d. The minimum value
e. The number of marks below 10

3.4) Write a script that takes two operands and an operator (+, -, x, /, pow, mod) as argument or by prompting the user when no argument is provided, and displays the result of the desired operation in the terminal:

```
./script.sh 3 x 9
27
```

3.5) Write a script that takes as argument an integer number (n) and displays the factorial (n!). Verify that the number of arguments is equal to 1, otherwise, prompt the user to enter the value of n.

3.6) Write a script that prompt user to enter various integer values to fill an array calls the bubble sorting function (or any other sorting method) to sort the array and display the result.

3.7) Write a bash script that displays the number of characters in each line of a file passed as an argument.

3.8) Write a bash script that transforms the number passed into an argument into: binary, octal and hexadecimal.

3.9) Write a bash script that is used to identify and count the bash files in the current work directory. Note that some bash files do not have the extension sh.

3.10) Write a bash script that takes a word as an argument, and check if this word exists in the French language. The French dictionary is in: /usr/share/dict/French

3.11) Write a bash script that returns the number of days in month given as argument

3.12) Write a script that deletes duplicated files (same content with different names) in the current directory.

3.13) Write a script that deletes files with size greater than 100MB in the current directory

3.14) Write a script that takes as input an alpha-numerical chain from user (e.g. 123abc456) and transforms lower case letters in the chain into upper case letters.

Exercise 4

We consider the following array:

```
fruits=(apple banana pear orange)
```

Give and explain the output of each of the following commands:

```
echo ${#fruits}
echo ${#fruits[@]}
echo ${#fruits[0]}
echo ${#fruits[1]}
echo ${#fruits[2]}
echo ${#fruits[3]}
echo ${food[@]:3}
echo ${food[@]:2:3}
```

4.1) Give the bash commands for:

4.2) Add the element "aa" at the beginning of the table

4.3) Add the element "gg" at the end of the table

4.4) Add the element "dd" in the middle

Exercise 5

In this exercise, we use the squid configuration file located in *″/etc/squid/squid.conf″* as input file. You can install squid using the following command: apt install squid

1. Write a grep command that selects the lines from the *″squid.conf″* that have exactly three characters
2. Write a grep command that selects the lines from the *″squid.conf″* that have at least three characters.
3. Write a grep command that selects the lines from the *″squid.conf″* that have three or fewer characters
4. Write a grep command that counts the number blank lines in the *″squid.conf″*
5. Write a grep command that selects the lines from the *″squid.conf″* that do not start with A to G
6. Write a grep command that deletes the comments and the white lines from the file ″squid.conf″
7. Give the ″sed″ command that replaces the whole occurrence of the word ″squid″

by ″squid3″ only in the first 25 lines

8. Give the ″sed″ command that replaces the word ″squid″ by ″squid3″ in whole lines in the file *″squid.conf″*
9. Give the ″sed″ command that deletes whole lines containing the substring ″ACL″ (case insensitive)
10. Give the ″sed″ command that deletes from line number 10 to the last line
11. Give the ″sed″ command that deletes lines containing the pattern ″src″
12. Give the ″sed″ command that deletes whole comments and empty lines

Exercise 6

Write a script *AWK* to display the first name, the average, minimum and maximum marks for each student in the file *"marks.csv"*:

```
$head -7 marks.csv
Marwa;renard;9214;17;14;12;15;14;10
Gwendal;lay;8812;15;16;5;8;2;16
Sandrine;tureleti;9114;18;17;19;15;13;3
Edward;robinet;9337;7;8;5;2;0;10
Guillaume;sapin;2415;13;3;14;9;15;17
Yacine;bellaiche;9876;17;15;13;19;5;6
David;mitterand;4321;12;5;9;8;0;11
```

The file *"marks.csv"* contains more than 350 records (lines), with one line per student. The first field contains the first name of the student, the second field contains the last name of the student, the third field contains the student number (unique identifier), the other fields (4 →9) contains the notes (marks).