



C Piscine

Shell 01

Summary: This document contains the instructions for Shell module 01 of the C Piscine @ 42

Version: 7.6

Contents

I	Instructions	2
II	Foreword	3
III	Exercise 00: Exam	4
IV	Exercise 01: print_groups	5
V	Exercise 02: find_sh	6
VI	Exercise 03: count_files	7
VII	Exercise 04: MAC	8
VIII	Exercise 05: Can you create it ?	9
IX	Exercise 06: Skip	11
X	Exercise 07: r_dwssap	12
XI	Exercise 08: add_chelou	13
XII	Submission and peer-evaluation	14

Chapter I

Instructions

- Only this page will serve as reference; do not trust rumors.
- Watch out! This document may change up to an hour before submission.
- These exercises are carefully arranged in order of difficulty, from easiest to hardest. We will not consider a successfully completed harder exercise if an easier one is not perfectly functional.
- Ensure that you have the appropriate permissions on your files and directories.
- You must follow the **submission procedures** for every exercise.
- Your exercises will be checked and graded by your fellow classmates.
- Additionally, your exercises will be checked and graded by a program called **Moulinette**.
- **Moulinette** is extremely meticulous and strict in its evaluation. It is entirely automated, and there is no way to negotiate with it. To avoid unpleasant surprises, be as thorough as possible.
- Shell exercises must be executable with `/bin/sh`.
- You must not leave any additional files in your directory other than those specified in the assignment.
- Have a question? Ask the peer on your right. If not, try the peer on your left.
- Your reference guide is called **Google / man / the Internet / ...**
- Examine the examples carefully. They may contain details that are not explicitly mentioned in the assignment.

Chapter II

Foreword

Here's what *Wikipedia* says about otters:

The European otter (*Lutra lutra*), also known as the Eurasian otter, Eurasian river otter, common otter and Old World otter, is a European and Asian member of the Lutrinae or otter subfamily, and is typical of freshwater otters.

The European otter is a typical species of the otter subfamily. Brown above and cream below, these long, slender creatures are well-equipped for their aquatic habits. Its bones show osteosclerosis, increasing their density to reduce buoyancy.

This otter differs from the North American river otter by its shorter neck, broader visage, the greater space between the ears and its longer tail.

However, the European otter is the only otter in its range, so it cannot be confused for any other animal. Normally, this species is 57 to 95 cm (23-37 in) long, not counting a tail of 35-45 cm (14-18 in).

The female is shorter than the male.

The otter's average body weight is 7 to 12 kg (15.4-26.4 lbs), although occasionally a large old male may reach up to 17 kg (37 lbs).

The record-sized specimen, reported by a reliable source but not verified, weighed over 24 kg (53 lbs).

The European otter is the most widely distributed otter species, its range including parts of Asia and Africa, as well as being spread across Europe. Though currently believed to be extinct in Liechtenstein, and Switzerland, they are now very common in Latvia, along the coast of Norway and across Great Britain, especially Shetland, where 12% of the UK breeding population exist. Ireland has the highest density of Eurasian otters in Europe.


In Italy, they can be found in southern parts of the peninsula.

The South Korean population is endangered.

Otters are cute.

Chapter III


Exercise 00: Exam

	Exercise : 00
Exam	

- During the week, you will be able to sign up for Friday's exam in the agenda, don't forget!
- You must also register for the **Exam00** project.
- Double-check that you are registered for both the exam event and the project!
- Triple-check that you are definitely registered for both the exam event and the project, yes, both!.

Chapter IV

Exercise 01: print_groups

	Exercise 01
	print_groups.sh
	Turn-in directory : <i>ex01/</i>
	Files to turn in : print_groups.sh
	Allowed functions : None

- Write a command line that displays the list of groups the user (defined in the environment variable `FT_USER`) belongs to.
- The output should be comma-separated, without spaces.
- Examples:

- for `FT_USER=nours`:

```
$>./print_groups.sh  
god,root,admin,master,nours,bocal$>
```

- for `FT_USER=daemon`:


```
$>./print_groups.sh  
daemon,bin$>
```



- `man id`
- Get inspired by others, but do the work yourself!

Chapter V

Exercise 02: find_sh

	Exercise 02
	find_sh.sh
	Turn-in directory : <i>ex02/</i>
	Files to turn in : find_sh.sh
	Allowed functions : None

- Write a command line that searches for all files ending with `.sh` in the current directory and all subdirectories.
- The output should display only the file names without the `.sh` extension.
- Example output:


```
$>./find_sh.sh | cat -e
find_sh$
file1$
file2$
file3$
$>
```



Don't blindly trust sources-always test, verify, and validate your results yourself!

Chapter VI

Exercise 03: count_files

	Exercise 03
	count_files.sh
	Turn-in directory : <i>ex03/</i>
	Files to turn in : count_files.sh
	Allowed functions : None

- Write a command line that counts and displays the total number of regular files and directories in the current directory and all its subdirectories.
- The count should include "." (the starting directory).
- Example output:


```
$>./count_files.sh | cat -e
42$
$>
```



Failure is part of your learning journey-keep testing and improving!

Chapter VII

Exercise 04: MAC

	Exercise 04
	MAC.sh
	Turn-in directory : <i>ex04/</i>
	Files to turn in : MAC.sh
	Allowed functions : None


- Write a command line that displays your machine's MAC addresses, with each address followed by a line break.



- `man ifconfig`
- Collaboration is key to success!

Chapter VIII

Exercise 05: Can you create it ?

	Exercise 05
Can you create it ?	
Turn-in directory : <i>ex05/</i>	
Files to turn in : "\?\${*}'MaRViN'*\$?\\"	
Allowed functions : None	

- Create a file containing **only** "42", and **nothing** else.
- The file name must be:

```
"\?${*}'MaRViN'*$?\\"
```

- Example output:

```
$>ls -lRa *MaRV* | cat -e
-rw---xr-- 1 75355 32015 2 Oct 2 12:21 "\?${*}'MaRViN'*$?\\"$
$>
```

Milestone Achieved, Keep Going!

You've completed the mandatory exercises for this project. Now, you have a choice:


- Continue with the **optional exercises** to explore more.
- Move on to your **next project**.

Both paths will introduce you to useful concepts. Consider the following before making your decision:

- Your first exam, as well as the end-of-week rush, will focus on C programming. It might therefore be useful to gain experience in this field beforehand. (You'll learn more about the rush soon).
- Your performance in this Piscine is evaluated on multiple factors:
 - Project completion is one aspect.
 - Overall progress through the full list of Piscine projects is another. Choose wisely to maximize your results.
- You can retry the same project in a few days or weeks until the end of the Piscine.
- Staying in sync with your peers promotes better collaboration.

Chapter IX

Exercise 06: Skip

	Exercise 06
skip.sh	
Turn-in directory : <i>ex06/</i>	
Files to turn in : skip.sh	
Allowed functions : None	

- Write a command line that executes `ls -l` but displays only every second line, starting from the first line.
- Example output:

```
$>ls -l | cat -e
total 4$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:46 skip.sh$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:41 tata$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:41 titi$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:41 toto$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:41 tutu$
$>
```


```
$>./skip.sh | cat -e
total 4$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:41 tata$
-rw-rw-r-- 1 eagle eagle ** ** 15 15:41 toto$
$>
```



Git push regularly!

Chapter X

Exercise 07: r_dwssap

	Exercise 07
r_dwssap.sh	
Turn-in directory : <i>ex07/</i>	
Files to turn in : r_dwssap.sh	
Allowed functions : None	

- Write a command line that processes the output of `cat /etc/passwd` with the following modifications:
 - Remove comments.
 - Keep every other line, starting from the second line.
 - Reverse each login name.
 - Sort the results in reverse alphabetical order.
 - Keep only logins between the environment variables `FT_LINE1` and `FT_LINE2` (inclusive).
 - Join them in a single line, separated by ", ".
 - End the output with a "."
- Example Output (for lines 7 to 15):

```
$> ./r_dwssap.sh
sstq_, sorebrek_brk_, soibten_, sergtsop_, scodved_, rlaxcm_, rgmecived_, revreswodniw_,
revressta_.$>
```



- Follow the steps in the exact order given!
- Did you check with your left-side neighbor?

Exercise 08: add_chelou



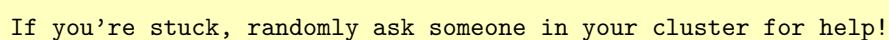
- Example 1:

```
FT_NBR1='\'?'\"'\"'\
FT_NBR2=rcrdmddd
```

- Salut

- ```
FT_NBR1="\\"!"!\""!\\"!"!\""!\\"!"!\""!
FT_NBR2=dcrmcmmooododmrrrmorcmcrmomo
```

- ```
Segmentation fault
```



Chapter XII

Submission and peer-evaluation

Submit your assignment to your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Make sure to double-check the filenames to ensure they are correct.



You must submit only the files explicitly required by the project instructions.