



Informatique

Travaux Pratiques

Axel LE BOT



Copyright © 2017-2018 Axel LE BOT

Licensed under the Creative Commons Attribution-NonCommercial 3.0 Unported License (the “License”). You may not use this file except in compliance with the License. You may obtain a copy of the License at <http://creativecommons.org/licenses/by-nc/3.0>. Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an “AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.



Table des matières

I

Shell

1	TP1 : Manipulations de l'environnement et des fichiers sous UNIX	11
1.1	Exercice 1 : Découverte de quelques commandes d'archivage	11
1.2	Exercice 2 : Utilisation des masques de création de fichiers	12
1.3	Exercice 3 : Manipulation du Systeme de fichier et des droits de navigation	13
1.4	Exercice 4 : Manipulation d'expression régulière	13
2	TP2 : Scripts Shell	17
2.1	Exercice 5 : Un premier script	17
2.2	Exercice 6 : Comptage des paramètres	17
2.3	Exercice 7 : Portée des variables	18

II

C++

3	TP1+TP2 : Tableaux, matrices et Fonctions recursives ..	21
3.1	Exercice sur des tableaux	21
3.1.1	Fonction sur les tableaux non triés	21
3.1.2	Algorithmes de tri de tableaux	21

3.1.3	Fonctions sur les tableaux triés	21
3.1.4	Exercice sur les Fonctions récursives	21
4	TP3+TP4 : Manipulation des arbres	23
4.1	Algorithmes sur arborescences binaires de recherche (ABR) non équilibrées	23
4.1.1	Exercice 1 : Mise en place d'ABR et premiers algorithmes	23
4.1.2	Exercice 2 : Algorithmes récursifs sur arborescences	23
4.2	Modification et parcours d'ABR non équilibrées	23
4.2.1	Exercice 3 : Insertion et suppression de valeurs dans une arborescence	23
4.3	Parcours d'arborescences binaires	23
4.3.1	Exercice 4 : Parcours sur arbres	23
5	TP5+TP6 : Hiérarchie de processus, signaux	25
5.1	Gestion des signaux : envoi et reception	25
5.1.1	Exercice 1 : Droits et signaux	25
5.1.2	Exercice 2 : capture de signal et traduction en langage C	25
5.1.3	Exercice 3 : Capture de signaux et redirections (exercice difficile)	25
5.1.4	Exercice 4 : envoi multiples et capture de signal en C	25
5.2	Gestion des processus	25
5.2.1	Exercice 5 : Processus en premier-plan / Arriere-plan	25
5.2.2	Exercice 6 : Duplication et recouvrement de processus	25
5.3	Gestion des processus - Suite	25
5.3.1	Exercice 7 : Duplication de processus	25
5.3.2	Exercice 8 : Creation et destruction de processus	25
5.3.3	Exercice 9 : Evaluation du nombre de processus	25
5.3.4	Exercice 10 : Conjonctions, Disjonctions, et Duplication	25
5.3.5	Exercice 11 : Terminaison normale de processus	25
6	TP7+TP8 : Communication socket	27
6.1	Communication distante en utilisant l'outil netcat	27
6.1.1	Exercice 1 : Découverte de la commande nc : netcat	27
6.1.2	Exercice 2 : Utilisation de la commande nc : netcat pour le transfert de fichier et l'évaluation de la bande passante	27
6.1.3	Exercice 3 : Une histoire de serveurs concurrents ...	27
6.1.4	Exercice 4 : Comprendre une requête HTTP	27
6.2	Développement d'un client et d'un serveur en C	27
6.2.1	Exercice 5 : Mise en place d'une communication en mode non connecte	27
6.2.2	Exercice 6 : Création d'une architecture (client UDP) - (relai UDP-TCP)-(serveur TCP)	27

6.3	Exercices bonus	27
6.3.1	Exercice 7 : Résolution de noms	27
6.3.2	Exercice 8 : Serveur multi-client en mode connecte	27
7	TP9+TP10 : Héritage multiple et modélisation	29
7.1	Exercice 1 : Organisation d'un jeu de combat au tour par tour	29



Table des figures

1.1	Permissions Unix	12
-----	------------------------	----

1 TP1 : Manipulations de l'environnement et des fichiers sous UNIX .. 11

- 1.1 Exercice 1 : Découverte de quelques commandes d'archivage
- 1.2 Exercice 2 : Utilisation des masques de création de fichiers
- 1.3 Exercice 3 : Manipulation du Systeme de fichier et des droits de navigation
- 1.4 Exercice 4 : Manipulation d'expression régulière

2 TP2 : Scripts Shell 17

- 2.1 Exercice 5 : Un premier script
- 2.2 Exercice 6 : Comptage des paramètres
- 2.3 Exercice 7 : Portée des variables



1. TP1 : Manipulations de l'environnement

1.1 Exercice 1 : Découverte de quelques commandes d'archivage

L'objectif de cet exercice est de découvrir et manipuler les commandes de téléchargement, d'archivage, de compression et de décompression de fichier

1. Récupération et décompression d'une archive

La commande `wget <url>` permet de télécharger un fichier présent à cette adresse. Ici nous récupérons une archive que nous pouvons manipuler avec `tar`. `tar` a trois options intéressante :

- L'option `-x` permet de restaurer les fichiers contenus dans une archive.
- L'option `-c` permet de créer une nouvelle archive.
- L'option `-f` permet d'utiliser le fichier archive `F` ou le périphérique `F` (par défaut `/dev/rmt0`).

On découvre que l'archive téléchargée contient 9 fichiers.

2. Manipulation de fichiers

- La commande `file <filename>` nous permet de savoir le format d'un fichier.
- La commande `mv <filename> <filename2>` me permet de déplacer mais aussi de renommer un "fichier1" en "fichier2". Ici j'ai utilisé la commande suivante : `mv image4.jpg image4.jpg2`.
- La commande `ls -lh <filename>` permet d'afficher les informations plus détaillées lisibles par l'humain. Ici la commande nous apprend que le fichier `script.txt` fait 170Ko.
- La commande `gzip <filename>` permet de compresser un fichier. Ici le fichier `script.txt` a été compressé. Il fait maintenant 65Ko. La compression est donc d'environ 38.235%.

- La commande `gunzip <filename>` permet de décompresser un fichier. Ici le fichier `script.txt` fait maintenant 170Ko, qui est bien la taille initial du fichier.

3. Création d'une nouvelle archive

- La commande `tar` permet de créer une archive sans la compresser. On peut tout de même utiliser `tar` pour archiver puis compresser des fichiers
- La commande `tar -z` permet de compresser l'archive au format `gzip`.
- La commande `tar -cz *.jpg *.txt *.jp2` n'est pas exécutée car il est impossible d'écrire des données compressées dans le terminal. Pour cela il faut rajouter l'option `-f`.
- La commande `tar -cz *.jpg *.txt *.jp2 > nouvelleArchive3.tar.gz` redirige bien le résultat dans un fichier. En effet le symbole `>` permet de rediriger la sortie vers un fichier, symbole `>`.

La redirection du flux dans un fichier recrée une archive compressée "archive3" similaire à "archive2" créé. En conclusion l'archive 2 et 3 donne le même résultat et sont plus petit que l'archive 1 puisqu'elles sont compressées.

1.2 Exercice 2 : Utilisation des masques de création de fichiers

Cet exercice permet de comprendre les droits UNIX à la création de fichier à l'aide des masque utilisateur. La commande `umask` permet de définir les permissions par défaut de fichiers ou de répertoires créées. Pour cela il faut préciser les droits que l'on veut supprimer. Ci-dessous un tableau de droits UNIX pour rappel.

Humain	Base 8	Base 2
---	0	000
--X	1	001
-W-	2	010
-WX	3	011
r--	4	100
r-X	5	101
rw-	6	110
rwX	7	111

FIGURE 1.1 – Permissions Unix

1.

Afin de créer des fichiers avec les droits adéquates, il faudra effectuer les commandes suivantes :

```
touch Raphael.txt
umask 0666
```

```
touch Donatello.txt
umask 0331
touch Michelangelo.txt
umask 0661
touch Leonardo.txt
umask 0000
```

2. et 3.

Après plusieurs test, on découvre qu'il n'est pas possible de donner plus de droit que la limitation par défaut du systeme.

- `umask 666` sur les fichiers
- `umask 777` sur les répertoires

1.3 Exercice 3 : Manipulation du Systeme de fichier et des droits de navigation

L'objectif de cet exercice est de manipuler les commandes de navigation dans l'arborescence et de création de fichier.

- La commande `mkdir <dirname>` permet de créer un répertoire.

1.

L'archive contient 5 images.

3.

Le chemin absolue d'un fichier ou d'un repertoire correspond au chemin vers le fichier ou repertoire depuis la racine. Par exemple : `/home/axel/Documents/Shell/TPs/TP1/Ex3/images/Chin`

4.

Le chemin relatif d'un fichier ou d'un repertoire correspond au chemin vers le fichier ou repertoire par rapport à un autre repertoire. Par exemple : `../P-Z/Vamporc.png`

6.

La commande `tar -xczf ITC313_TP_Shell_lebot.axel.tar.gz` permettra de décompresser et extraire l'archive.

7.

Après transfert de l'archive compressé toutes les permissions sont conservés.

1.4 Exercice 4 : Manipulation d'expression régulière

L'objectif de cet exercice est de manipuler les basiques des expressions régulière.

1.

La commande `wget https://cloud.infotro.fr/ITC313/unGrosBordel.txt` permettra de récupérer le fichier à analyser.

Chapitre 1. TP1 : Manipulations de l'environnement et des fichiers sous UNIX

14

2.

Les lignes affichées par `cat unGrosBordel.txt | grep -E "ette"` contiennent toutes la suite de lettres "ette".

3.

Les lignes affichées par `cat unGrosBordel.txt | grep -E "T"` contiennent toutes la lettre "T".

4.

Les lignes affichées par `cat unGrosBordel.txt | grep -E "^T"` contiennent toutes la lettre "T" en début de ligne.

5.

L'expression `^` signifie donc "début".

6.

Les lignes affichées par `cat unGrosBordel.txt | grep -E "te$"` contiennent toutes la suite de lettres "te" en fin de ligne.

7.

Les lignes affichées par `cat unGrosBordel.txt | grep -E "c.r"` contiennent toutes la suite de lettres "c", un caractère quelconque, "r".

8.

Les lignes affichées par `cat unGrosBordel.txt | grep -E "(oui|non)"` contiennent toutes soit "oui", soit "non".

9.

- `$` représente la fin d'une ligne.
- `|` représente une condition "ou".
- `.` représente un caractère quelconque.

10.

L'option `-o` dans la commande `cat unGrosBordel.txt | grep -o -E "c.r"` permet de n'afficher que la partie de la ligne correspondant à l'expression "c.r" appelé "motif".

11.

Les motifs affichés par `cat unGrosBordel.txt | grep -o -E "[A-Z]"` contiennent une suite de 4 lettres majuscule.

12.

Les motifs affichés par `cat unGrosBordel.txt | grep -o -E "[A-Z][a-z]+"` contiennent une suite d'au moins une majuscule et une minuscule.

13.

Les motifs affichés par `cat unGrosBordel.txt | grep -o -E "[A-Z][a-z]*"` contiennent une suite de 0, 1 ou plus de une majuscule et une minuscule.

14.

- + représente 1 ou plusieurs.
- * représente 0, 1 ou plusieurs.

15.

La commande

```
cat unGrosBordel.txt | grep -o -E "[A-Za-z0-9\.\_]+@[A-Za-z0-9\-\]+\.[a-zA-Z]{2,4}"
```

permet de récupérer les adresse e-mail.

16.

La commande

```
cat unGrosBordel.txt | grep -o -E "(\+33|0)(\.| |)?[0-9]((\.| |)?[0-9]{2}){4}"
```

permet de récupérer les numéro de téléphone.

17.

La commande `cat unGrosBordel.txt | grep -o -E "\(\([a-z]+\)\)"` permet de trouver la phrase secrète : "bien joue tu as trouve la reponse a la derniere question"

2. TP2 : Scripts Shell

2.1 Exercice 5 : Un premier script

```
#!/bin/bash
# Axel LE BOT - 2017-10-01
clear
USER=$(whoami)
echo "Hello $USER"
echo -e "\n\e[9mhello world\e[0m\n" \ "hello Kitty\n"
echo "Do you like fishsticks(y/n)?"
read answer

if [ "$answer" = "y" ]
then
    echo "then you are a gayfish!"
else
    echo "OK"
fi
```

2.2 Exercice 6 : Comptage des paramètres

```
#!/bin/bash
# Axel LE BOT - 2017-10-02
echo "Ex6 - paramètres"
echo -n "-Liste des parametres entres : "
COUNT=0

for i in $*
```

```
do
    echo -n "$i "
    (( COUNT++ ))
done

echo -e "\n-Nombre de parametres : $COUNT"

#!/bin/bash
# Axel LE BOT - 2017-10-02
echo "Ex6 - paramètres"
echo -n "-Liste des parametres entres : "

A=$#
for ((i=0 ; i<A ; i++)) do
    echo -n "$1 "
    shift
done

echo -e "\n-Nombre de parametres : $A"
```

2.3 Exercice 7 : Portée des variables

1. Portée des variables locales

La variable créé dans le terminal n'est pas accessible depuis un script.

2. Portée limitée au shell

Les variables sont local au terminal

3. Étendre la portée de la valeur d'une variable locale

3	TP1+TP2 : Tableaux, matrices et Fonctions recursives	21
3.1	Exercice sur des tableaux	
4	TP3+TP4 : Manipulation des arbres	23
4.1	Algorithmes sur arborescences binaires de recherche (ABR) non équilibrées	
4.2	Modification et parcours d'ABR non équilibrées	
4.3	Parcours d'arborescences binaires	
5	TP5+TP6 : Hiérarchie de processus, signaux	25
5.1	Gestion des signaux : envoi et reception	
5.2	Gestion des processus	
5.3	Gestion des processus - Suite	
6	TP7+TP8 : Communication socket	27
6.1	Communication distante en utilisant l'outil net-cat	
6.2	Développement d'un client et d'un serveur en C	
6.3	Exercices bonus	
7	TP9+TP10 : Héritage multiple et modélisation	29
7.1	Exercice 1 : Organisation d'un jeu de combat au tour par tour	



3. TP1+TP2 : Tableaux, matrices et Fonctions

3.1 Exercice sur des tableaux

3.1.1 Fonction sur les tableaux non triés

Exercice 1 : Algorithmes de parcours classiques sur tableau non triés

Quelques copier coller ont suffi. Les tests ont bien été effectués.

Exercice 2 : Ajout et suppression d'éléments tableaux non triés

3.1.2 Algorithmes de tri de tableaux

Exercice 3 : Trier des tableaux aléatoires

3.1.3 Fonctions sur les tableaux triés

Exercice 4 : Algorithmes de parcours classiques sur tableau non triés

Exercice 5 : Ajout et suppression d'éléments sur tableaux triés

3.1.4 Exercice sur les Fonctions récursives

Exercice 6 : Définition de fonction récursive

Exercice 7 : Algorithme récursif sur matrice



4. TP3+TP4 : Manipulation des arbres

- 4.1 Algorithmes sur arborescences binaires de recherche (ABR) non équilibrées**
 - 4.1.1 Exercice 1 : Mise en place d'ABR et premiers algorithmes**
 - 4.1.2 Exercice 2 : Algorithmes récurifs sur arborescences**
- 4.2 Modification et parcours d'ABR non équilibrées**
 - 4.2.1 Exercice 3 : Insertion et suppression de valeurs dans une arborescence**
- 4.3 Parcours d'arborescences binaires**
 - 4.3.1 Exercice 4 : Parcours sur arbres**



5. TP5+TP6 : Hiérarchie de processus, signaux

5.1 Gestion des signaux : envoi et reception

- 5.1.1 Exercice 1 : Droits et signaux
- 5.1.2 Exercice 2 : capture de signal et traduction en langage C
- 5.1.3 Exercice 3 : Capture de signaux et redirections (exercice difficile)
- 5.1.4 Exercice 4 : envoi multiples et capture de signal en C

5.2 Gestion des processus

- 5.2.1 Exercice 5 : Processus en premier-plan / Arriere-plan
- 5.2.2 Exercice 6 : Duplication et recouvrement de processus

5.3 Gestion des processus - Suite

- 5.3.1 Exercice 7 : Duplication de processus
- 5.3.2 Exercice 8 : Creation et destruction de processus
- 5.3.3 Exercice 9 : Evaluation du nombre de processus
- 5.3.4 Exercice 10 : Conjonctions, Disjonctions, et Duplication
- 5.3.5 Exercice 11 : Terminaison normale de processus



6. TP7+TP8 : Communication socket

6.1 Communication distante en utilisant l'outil netcat

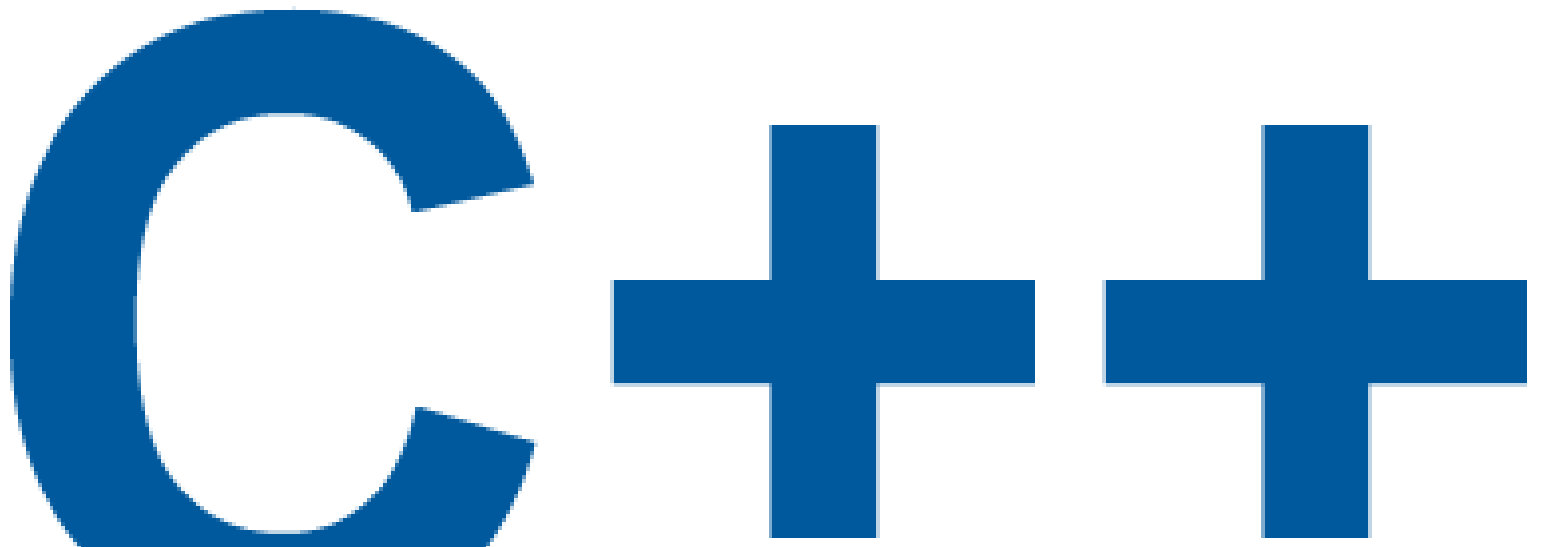
- 6.1.1 Exercice 1 : Découverte de la commande nc : netcat
- 6.1.2 Exercice 2 : Utilisation de la commande nc : netcat pour le transfert de fichier et l'évaluation de la bande passante
- 6.1.3 Exercice 3 : Une histoire de serveurs concurrents ...
- 6.1.4 Exercice 4 : Comprendre une requête HTTP

6.2 Développement d'un client et d'un serveur en C

- 6.2.1 Exercice 5 : Mise en place d'une communication en mode non connecte
- 6.2.2 Exercice 6 : Création d'une architecture (client UDP) - (relai UDP-TCP)- (serveur TCP)

6.3 Exercices bonus

- 6.3.1 Exercice 7 : Résolution de noms
- 6.3.2 Exercice 8 : Serveur multi-client en mode connecte



7. TP9+TP10 : Héritage multiple et modélisation

7.1 Exercice 1 : Organisation d'un jeu de combat au tour par tour