Module EC 523
Système d'exploitation, scripting, sécurité
L3 PR CYBERDEVOPS

Vincent Vigneron,
vincent.vigneron@univ-evry.fr
Univ Evry, université Paris-Saclay, UFR ST

4 octobre 2023

Table des matières

Préparation des séances de TP Se connecter sous Unix :

- Se connecter sous Windows avec le login '00000005' et mot de passe '50000000'
- Lancer le programme XMing (icône Connexion ENS-Unix)

Préparation de la session sous Linux

- Lancer le programme `gedit &` ;
- Par défaut, le programme lance une invite de commande (appelée aussi console) ;
- Trouver le nom du répertoire courant : il s'agit de votre répertoire personnel ;
- Créer, dans votre répertoire, un sous-répertoire `EC521/TP1` pour la premiere séance de TP, `EC521/TP2` pour la deuxième séance de TP, etc.
- Se déplacer dans le répertoire que vous venez de créer.

Un mot sur les comptes-rendus de TP Les comptes-rendus de TP sont à rendre pour le 10/11/2022 sur la plateforme E-CAMPUS. Je souhaite voir dans vos TPs les commandes tapées pour répondre aux questions. Attention, ce n'est pas une thèse. Faites au plus court !

Activité 1

TD 1 : arborescence et premières commandes

L'objet de ces 2 premiers TD est l'étude des commandes externes du SHELL et de certains méta-caractères. La connaissance de la syntaxe des principales commandes externes du SHELL est un pré-requis.

1 Soit un répertoire contenant les fichiers suivants :
`f1.c, f2.c , f3.f , f4.c , f5.f , f10.a , f11.a , a.out , e.c , t.f`
Utiliser les méta-caractères de substitution pour lister les fichiers suivants : (on recherchera l'écriture minimale)

1. fichiers fortran (suffixe `f`),
2. les fichiers C et fortran,
3. les fichiers commençant par la lettre `f`,
4. les fichiers C commençant par la lettre `f`,
5. les fichiers dont le nom contient un chiffre avant '`.`',
6. les fichiers dont le 2nd caractère est un chiffre,
7. les fichiers dont le 2nd caractère est un '`.`' .

2 **Création d'arborescence**

Sans bouger du répertoire racine (celui qui est à la base de l'arborescence ; il s'agit ici de ∼), créez l'arborescence suivante (Fig. 1) :

3 **Navigation dans l'arborescence**

1. Dans votre répertoire personnel, créer un répertoire `test`. Créer sous-répertoires `pub` et `bin` du répertoire `test`.
2. Modifier les droits d'accès de façon que :
   (a) le répertoire `test` et son sous-répertoire `pub` soient accessibles en lecture et exécution pour tous, et en écriture pour le propriétaire seul.
   (b) le répertoire `bin` soit accessible en exécution pour tous, et en lecture et écriture pour le propriétaire et son groupe.
3. Aller dans le répertoire `test`. Créer un fichier `doc.txt`. Recopier le fichier dans `pub`.
4. Aller dans `bin`. Afficher à l'écran le contenu de `test/pub/doc.txt`.
5. Afficher la liste des fichiers contenus dans `test` et ses sous-répertoires.

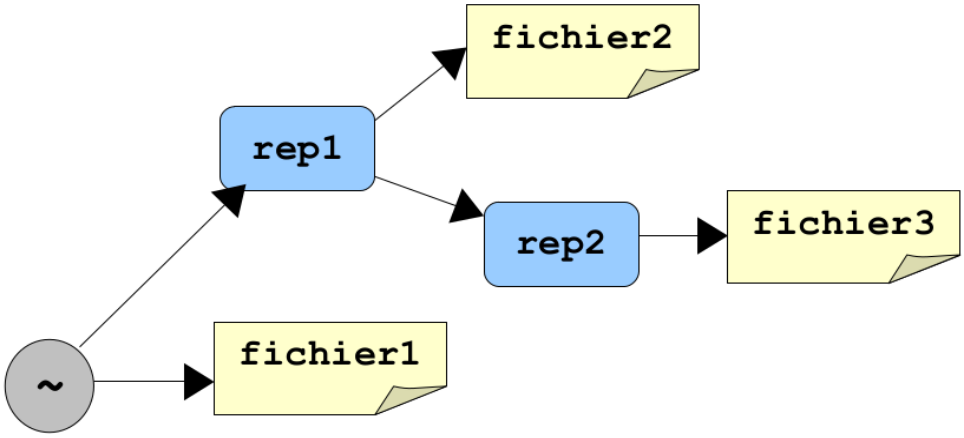

FIGURE 1.1 – Création d'arborescence.

6. Supprimer du répertoire `test` et de ses sous-répertoires tous les fichiers se terminant par le caractère ∼.
7. Afficher à l'écran le contenu des fichiers `toto` et `titi` qui se trouvent dans le répertoire d'accueil de l'utilisateur (fictif) `Makeba`. Supprimer ces deux fichiers

4

Commandes `grep` et `cut`

On suppose qu'un fichier carnet-adresse `liste.txt` contient des informations sur des commerciaux d'une entreprise (Table 1.1). Chaque ligne représente un employé, et contient les informations suivantes : nom, âge, téléphone et ville d'habitation. Les champs seront séparés par un '`:`'.

edouard :29 :0298333242 :Brest
valentin :13 :0466342233 :Gardagnes
ingrid :30 :0434214452 :Nimes
mathieu :92 :013344433 :Palaiseau
⋮

TABLE 1.1 – Contenu du fichier `liste.txt`.

La ligne `edouard:29:0298333242:Brest` correspond au commercial edouard, dans le département 29, de numéro de téléphone 0298333242 et habitant à Brest.

1. Renvoyer toutes les lignes du fichier `liste.txt` qui correspondent au commercial s'appelant '`Sami`'.
2. Renvoyer toutes les lignes correspondant aux commerciaux affectés à la ville de Créteil.
3. Renvoyer toutes les lignes des commerciaux âgé de '`22`' ans.
4. Renvoyer les lignes des étudiants n'ayant pas '`22`' ans.
5. Renvoyer toutes les lignes contenant la chaîne '`mi`' sans tenir compte de la casse.
6. Afficher le nom et le téléphone de chaque commercial, puis le nom et l'âge.
7. Afficher les trois premiers caractères de chaque ligne.

5

Visualisation de fichiers

Lister tous les fichiers :

1. se terminant par '5',
2. commençant par 'annee4',
3. commençant par 'annee4' et de 7 lettres maximum,
4. commençant par 'annee' avec aucun chiffre numérique,
5. commençant par 'a' ou 'A'
6. contenant la chaîne 'ana',

6 Donner la signification des commandes suivantes :

```
sort f.c | head
grep printf f.c | wc -l
grep printf f.c > sortie
cat /etc/passwd | grep user1
```

7 Déterminer les commandes qui permettent de :

1. savoir si l'utilisateur `user1` est connecté,
2. afficher le nombre d'utilisateur du système,
3. afficher la liste des utilisateurs par ordre alphabétique,
4. connaître le nombre de processus de `user1`,
5. connaître le nombre de processus de `root`,
6. enregistrer dans le fichier `fuser1` la date et l'ensemble des fichiers de `user1`.

8 Écrire une commande qui affiche l'ensemble des processus dont vous n'êtes pas propriétaire (votre nom d'utilisateur se trouvant dans la variable d'environnement `$USER`).

9 **Sleep**

Donner la syntaxe qui lance la commande `sleep` en arrière plan (*background*) pendant une durée de 5 minutes.

10 Créer un sous-répertoire de `/tmp` ayant pour nom votre nom de login. Positionnez vous dans `/tmp`. Créer dans ce sous-répertoire un fichier qui est la copie conforme de votre fichier `.profile` . Ce nouveau fichier doit avoir un nom ayant pour préfixe `.profile` et pour suffixe votre nom d'utilisateur. Protégez le contenu de ce sous-répertoire contre tout regard indiscret y compris le votre.

11 **Commande find**

1. Écrire une commande `find` qui va rechercher à partir de votre répertoire `HOME`, les fichiers nommés `core` ou `a.out` et les supprimer.
2. Chercher tous les fichiers dont le nom est 'passwd'.
3. Chercher tous les fichiers dont la date de la dernière modification remonte à plus de 10 minutes.
4. Trouver tous les fichiers du groupe 'root'.
5. Chercher tous les fichiers dont la taille est supérieure à 20 Mo.
6. lister tous les répertoires se trouvant sous `/etc`.
7. lister tous les fichiers de l'utilisateur 'antoine'

Activité 2

TD 2 : Scripting

L'objet de ce TD est l'étude des fonctions des scripts sh. On commence par une série simple.

12 **Premier script**

Écrire une procédure de commande `params` qui affiche le nom du script (`$0`), le nombre de paramètres (`$#`) et la liste des paramètres (`$*`)

13 **Liste de paramètres**

Écrire un script qui fait la même chose que `params` mais qui affiche la liste des paramètres à raison d'un par ligne, (quel que soit le nombre de paramètres même > 10)

14 **Somme des arguments**

Écrire un script `sh` qui réalise la somme de tous les arguments acquis à partir de la ligne de commande.

15 **Date en anglais**

Taper `date` dans votre terminal et observer le résultat de la commande. Écrire un script `sh` qui réalise l'affichage de la date en anglais.

16 **A rebours**

Écrire un script `sh` qui réalise l'affichage d'un décompteur : `10 , 9 , 8, etc.`

17 **Répertoire**

Écrire un script qui affiche le nom des fichiers du répertoire courant en utilisant une boucle `for`, mais sans utiliser la commande `ls`

18 **Fichiers exécutables**

Écrire un script qui affiche uniquement les noms des fichiers exécutables dans le répertoire courant.

19 **Commande à options (`case...`)**

Modifier ce script pour qu'il accepte une option :

- `-x` affichage des fichiers exécutables,
- `-d` affichage des répertoires
- `-r` affichage des fichiers lisibles
- `-w` affichage des fichiers modifiables.

20 Écrire un script qui copie tous les fichiers exécutables du répertoire courant dans un autre répertoire dont le nom est donné en argument.

21

Nouveau Del

Écrire une procédure de commande `del` qui a pour but de remplacer `rm`, en conservant les fichiers effacés dans un répertoire `$HOME/.del`. Dans le cas où un fichier de même nom existe déjà dans le répertoire, on suffixera le nom par `=n` ou `n` est le numéro de version.

22

Numérique

Écrire un programme `numerique` qui teste si une valeur passée en paramètre est numérique ou non (le code retour de la commande `expr "$variable" + 0` est `0`).
Exemple :

```
> numérique 13
ok numérique
> numérique a34fd
non numérique
```

23 Écrire un programme `affichen` qui prend un paramètre numérique `n` et génère en sortie la liste des n premiers entiers.
Exemple :

```
> affichen 15
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
>
```

24

C `umask`

Écrire un script permettant de positionner à `640` les autorisations d'accès de tous les fichiers normaux du répertoire courant. Si un nom de fichier est donné le traitement ne portera que sur ce fichier.

25 Modifier le script précédent pour que le premier paramètre soit considéré comme la permission à attribuer. Si le premier paramètre n'est pas un chiffre en base 8, il devra être interprété comme un nom de fichier et la valeur par défaut 640 lui sera appliqué.
Par exemple :

1. `ch 644` : Tous les fichiers normaux passent en 644.
2. `ch` : Tous les fichiers normaux passent en 640.
3. `ch toto titi` : Les fichiers `toto` et `titi` passent en 640.

26 Créez un script `question`. Vous lui donnerez en paramètre le texte d'une question à laquelle correspond une réponse de type Oui/Non. Le script doit afficher la question à l'écran et attendre la réponse. Si la réponse commence par `O` ou `o` il renvoie `0`, et si elle commence par `n` ou `N` il renvoie 1. Dans tous les autres cas, la question est reposée une nouvelle fois.

27 Créez un script appelé `Supprime`. Vous lui confierez des noms de fichiers et pour chaque fichier, il s'agira de vérifier que ce sont des fichiers normaux. Aucun autre type de fichier ne sera accepté. Pour les fichiers autorisés, la question devra être posée, s'il faut les supprimer ou non. Une réponse positive entraînera la suppression du fichier.
Réécrire ce script shell précédent en transformant `question` en une fonction.

28

Numbers...

Écrire un script shell qui lit 2 nombres entrés en arguments par l'utilisateur et qui affiche une phrase, bien évidemment correcte, du type Le nombre 3 est inférieur au nombre 5 (dans le cas où les deux nombres entrés sont 3 et 5). Attention, penser à gérer le cas de l'égalité [1].

29

Aspirateurs de site

On se propose d'écrire un petit script d'une ligne pour télécharger des webcomics, en l'occurrence `xkcd` (`http://xkcd.com`).
Une petite analyse du site permet de voir que la i ème image se trouve sur la page web `http://xkcd.com/i`. Par exemple, la page web de la 3 ème image est `http://xkcd.com/3`.
Écrire une boucle for permettant d'afficher toutes les urls correspondantes aux images de 1 à 15.
On utilisera pour ce faire les commandes `seq` et `echo`. On se propose de télécharger la page web

FIGURE 2.1 – Création d'arborescence.

et de l'afficher sur la sortie standard. Pour se faire, on peut utiliser par exemple "`wget -O -`" (attention, `O` majuscule ici) ou "`curl -o -`". Il reste alors à extraire la ligne qui nous intéresse à l'aide de la commande `grep`. Enfin, dernière étape, extraire de cette ligne l'URL. On utilise une fois de plus `grep`. La commande extrayant l'URL est donc au final :

```
wget http://xkcd.com/1 -O - | grep hotlink | grep -o 'http.*jpg'
```

Il est donc désormais possible d'itérer sur chaque page, d'extraire à chaque fois l'URL de l'image, et de télécharger l'image à l'aide de `wget`.

1. La commande `test XXX -le YYY` teste si `XXX` est plus petit ou égal à `YYY`. La commande test `XXX -eq YYY` teste l'égalité.

Activité 3

TP 1 : Début dans l'environnement Unix

30

Prise en main

1. Créez un fichier `premiertexte` contenant une ou deux phrases.
2. Quelle est la taille de `premiertexte` ?
3. Éditez `PREMIERTEXTE`. Que constatez-vous ?
4. Faites une copie de `premiertexte` appelée `double`.
5. Comparez les tailles de `premiertexte` et de `double`
6. Renommez `double` en `introduction`.
7. Quelle différence y a-t-il entre `mv double introduction` et `cp double introduction` ?
8. Créez un répertoire `essai`
9. Déplacez `introduction` dans `essai`.
10. Faites une copie de `premiertexte` appelée `copie`, et placez-la également dans `essai`.
11. Affichez une liste de ce que contient `essai`.
12. Essayez de détruire `essai`. Que se passe-t-il ? Que faut-il faire pour détruire un répertoire ?
13. Détruisez tout ce que contient `essai`.
14. Détruisez `essai`.

31

1. Où que vous soyez, quel est l'effet de la commande `cd` sans paramètre ?
2. Combien y a-t-il de noms de répertoire dans la racine '/' ?
3. Donnez un exemple de nom de fichier se trouvant dans votre répertoire personnel (a) par un chemin *relatif* (b) par un chemin *absolu*.

32

Commande `chmod`

1. modifiez les droits du fichier `blablabla` pour que tous ceux de votre les membres groupe puissent écrire dedans.
2. Quelle commande donne le droit d'exécution à tous les utilisateurs du fichier `blablabla` qui n'a jusqu'alors que des droits standards (`-rw-r- - r- - `) ?
3. Le fichier `blablabla` a les droits suivants : `-rwxr-xr-x`. Quelle commande donne seulement le droit de lecture aux utilisateurs ?

4. Quelle commande modifie les droits du fichier `blablabla` (`-rwxr- - r- -`) pour que le groupe et les autres aient les mêmes droits que le propriétaire ?
5. Quelle option de la commande `chmod` permet de modifier récursivement les droits d'un répertoire et des fichiers qu'il contient ?
6. Donner une commande utilisant `mkdir -m` pour créer un répertoire en spécifiant les droits sur ce répertoire ?
7. Affichez et interprétez les droits de `/usr/sbin`.

33

Commande `find`

1. En prenant soin de diriger les erreurs vers la *poubelle* `/dev/null`, cherchez dans '`/`' les fichiers dont le nom se termine par `.c`
2. commençant par `Z` ou `z`.
3. Dont les noms ne contiennent pas de chiffre.
4. Chercher dans `/usr` les fichiers dont la taille dépasse 1Mo (2000 blocs de 500Ko) et dont les droits sont fixés à 755 (`-rwxr-xr-x`).
5. Combien il y a de fichiers vous appartenant dont les droits sont fixés à 666 (`-rw-rw-rw-`).
6. Trouver le fichier `.bashrc` dans '`/`' et supprimez le (après confirmation).

34

Commande `sort`

1. Copiez le fichier `/etc/passwd` dans votre ' ∼/'. Afficher seulement les champs contenant le login et le "home directory".
2. Triez le fichier `passwd` à partir du nom
3. Editez les nom de login et UID, puis triez suivant les ordres croissants des UID en une seule commande ; vous dirigerez le résultat vers un fichier `uid_proof`
4. remplacer '`:`' dans le fichier de résultats précédent par un espace ' '.
5. Éditez les 8 dernières lignes du fichier `passwd`.
6. Éditez les 7 premiers caractères du fichier `passwd`.

35

Commande `ls`

Créer dans un répertoire les fichiers suivants

```
Napoli Pragua paris Paris  Londres jerusalem Le-Cap  bristol4 Nantes florence Rome veNISE
```

Éditez les fichiers (sortie de la commande `ls` redirigée vers `grep`) avec les critères sur leur nom suivant :

1. Le nom doit être '`paris`' ou '`Paris`'
2. '`es`' est en fin de nom
3. '`ri`' est présent dans le nom
4. Nom contenant un chiffre numérique ou '`r`'
5. Nom contenant la chaîne '`is`' ou '`IS`'

36

1. Copiez v dans votre home directory. Éditez la ligne de ce fichier commençant par votre nom de login

2. Dans le fichier `passwd` qui est dans votre home directory, affichez les lignes commençant par des noms de login ne contenant pas de chiffre.
3. Éditez les lignes du fichier `passwd` commençant par des noms de login de 3 ou 4 caractères.

37

1. Afficher le nombre de sessions ouvertes (c-à-d de connexions) sur la machine tout en générant le détail de ses connexions dans le fichier connectes.
2. Afficher les L lignes qui entourent la ligne numéro N d'un fichier `F`. Utiliser cette méthode pour afficher une partie du texte qui entoure la ligne où se trouve une chaîne `S` qu'on cherche dans un fichier `F` (utiliser la commande `grep`).

38

Commande `tr`

`tr` (*Translate*), est un filtre ne reconnaissant pas les expressions régulières. Proposer une commande permettant de :

1. convertir et afficher la ligne saisie au clavier en transformant le texte en minuscules
2. remplacer tous les caractères minuscules (de `a,b`, ...`z`) par un espace
3. supprimer tout caractère minuscules de la chaîne entrée
4. supprimer les espaces multiples entre les mots dans un texte

39

Commande `grep`

`grep` (*General Regular Expression Parser* pour analyseur général d'expression régulière) sélectionne toutes les lignes qui satisfont une expression régulière. Donner les commandes suivantes permettant de chercher dans un fichier `essai.txt` :

1. les lignes dont la 1ère lettre est quelconque et la 2ème doit être '`o`'
2. les lignes commençant par `t`
3. les lignes ne commençant pas par `t`
4. les lignes contenant le modèle `T.t.`
5. les lignes qui contiennent '`a`', '`b`' ou '`c`'
6. redirige la STD OUT du moniteur pour l'envoyer sur l'entrée de `wc` ?

40

1. Écrire une commande utilisant `at` pour exécuter le script `traitement` à 16h15 le 24 janvier 2023, sachant que nous sommes en septembre 2022.
2. Exécuter le script `traitement` se trouvant dans `bin` de votre répertoire de connexion :
   (a) Le Vendredi à 17h00
   (b) La semaine prochaine au même moment
   (c) Deux jours plus tard

41

Commande `sed`

`sed` (*Stream EDitor*) est un utilitaire qui sélectionne les lignes d'un fichier texte (ou d'un flot provenant d'un pipe) vérifiant une expression régulière et qui leur applique un traitement ou un remplacement. Écrire une commande

1. pour détruire toutes les lignes vides d'un fichier :

2. Écrire un script shell qui renomme, dans le répertoire courant, tous les fichiers comportant des blancs dans leur nom, en supprimant les blancs dans le nouveau nom. Si le temps le permet, faites en sorte que si un fichier avec le nouveau nom existe déjà, un numéro soit ajouté au nom pour le rendre unique [1]
3. Créer un script qui remplace le suffixe `.htm` d'un ensemble de fichiers en `.html`.

42

Commande `awk`

`awk` (dont le nom vient des initiales de ses 3 auteurs *Aho, Weinberger et Kernighan*) est l'implémentation GNU du langage `awk`. Écrire une commande qui

1. affiche toutes les lignes de plus de 20 caractères.
2. affiche le second champ de chaque ligne
3. affiche la liste des champs de chaque ligne dans l'ordre inverse
4. compte le nombre de lignes qui contiennent la chaîne de caractères `yop`

43

Application

On récupère le dictionnaire de la langue française qui se trouve sur la page E-MEDIA dans le fichier suivant `td01-dict.txt`.

1. compter le nombre de lignes, le trier alphabétiquement et supprimer les 13 premières lignes.
2. Combien de fois le mot `anti` y apparaît-il ? et le mot `pro` ?
3. Affichez le contenu du dictionnaire entre les lignes 200 et 215.
4. Créez un fichier contenant tous les mots de plus de 26 caractères.

44

recherche de mots

On cherche le mot `root` dans un fichier log des intrusions obtenu par la commande `lastcomm`. Écrire un script permettant de créer un fichier dans lequel figurera le nombre d'occurrences de ce mot avec le numéro des lignes.

1. *Indice :* Étant donné une chaîne $ `f` contenant des espaces, `echo "$f" | sed 's/ //g'` permet de supprimer ces espaces.

Activité 4

TP 2 : Scripting

45

Indice

Imaginez un script `indice.sh` qui vous affiche l'indice de son premier argument dans la liste de ses arguments, par exemple

`indice.sh un deux trois quatre un`

renverra l'indice de '`un`' dans la liste `un deux trois quatre un` soit 4. for ARG in $*$; $doindex$ =((index + 1)) if test ARG =motif ; then echo $motifestenposition$index fi done

46

Soit le fichier `essai.txt` contenant les lignes suivantes :

```
aaaa bbbb cccc dddd eeee ffff gggg hhhh iiii jjjj kkkk llll mmmm
nnnn oooo pppp qqqq rrrr ssss tttt uuuu vvvv wwww xxxx yyyy zzzz
```

Quelle enchaînement de commandes permet d'obtenir le résultat suivant :

```
aaaa bbbb cccc dddd eeee ffff gg | AAAA BBBB CCCC DDDD EEEE FFFxF GG
gg hhhh iiii jjjj kkkk llll mmmm | GG HHHH IIII JJJJ KKKK LLLL MMMM
nnnn oooo pppp qqqq rrrr ssss tt | NNNN OOOO PPPP QQQQ RRRR SSSS TT
tt uuuu vvvv wwww xxxx yyyy zzzz | TT UUUU VVVV WWWW XXXX YYYY ZZZZ
```

47

Peut on à l'aide des commandes `who, cut, uniq` et `wc` construire une commande qui compte le nombre d'utilisateur connectés au système.(éventuellement `man`...)

48

Manipulation de fichiers

Proposer un script qui, à partir des informations nom de connexion de l'utilisateur, nom complet (champ libre) et shell attribué à l'utilisateur provenant de `/etc/passwd`, reformate `/etc/passwd` sous la forme suivante :

```
root:*:0:0:Obelix:/:/bin/tcsh
daemon:*:1:1::/etc:
bin:*:2:2::/usr/bin:
sys:*:3:3::/usr/src:
adm:*:4:4: Administrateur du système:/usr/adm:/bin/sh
bobleponge:*:bobleponge:/home/staff/bobleponge:/bin/ksh
Joe:*:102:100:Asterix:/home/staff/Joe:/bin/ksh
```

Le résultat devrait être :

```
USER            WHO IS                          SHELL
root            Obelix                  /bin/tcsh
daemon
bin             BIN
sys             SYS
adm             Administrateur du système       /bin/sh
bobleponge      bobleponge                  /bin/ksh
Joe             Asterix                 /bin/ksh
```

49

Comptes clients

Un fichier `numbers` contient les comptes clients d'une entreprise avec une structure particulière qui est celle- ci :

5 chiffres pour le code banque,
5 chiffres pour le code guichet,
10 chiffres pour le numéro de compte,
1 lettre pour compléter le numéro de compte,
2 chiffres pour la clé RIB,
30 caractères pour la désignation du client.

Écrire uns script permettant de produire en sortie les informations suivantes (séparées par la barre verticale) : Désignation du client, son numéro à 10 chiffres et la lettre complétant ce numéro.

Soit `gestion` le fichier des gestionnaires des comptes. Il se compose de lignes ayants les champs (séparés par la barre verticale) suivants : Désignation du gestionnaire, lettre du numéro de compte des clients dont il est responsable.

Produire en sortie un listing avec les informations suivantes : Désignation du gestionnaire, le client dont il est responsable, le numéro de son compte.

50

Répertoire

Soient `tele` un fichier qui contient le répertoire téléphonique suivant :

```
Boileau         024867-6235
Derrick         024867-1842
Ernest          024867-1234
Grand           024867-2240
Herbrant        024867-0256
Jonathan        024867-7358
Louis           024867-3237
Tardif          024867-5341
Wagner          024867-1234
```

Et `noms` la liste suivante des noms et départements correspondants :

```
Ernest          Dept. 389
Frolo           Dept. 217
Grand           Dept. 311
Tardif          Dept. 454
Wagner          Dept. 520
```

Construire un fichier `NTD` (nom, téléphone, département) où chaque ligne est un nom suivi du numéro de téléphone puis du numéro de département. Comment faire pour avoir tous les noms dans le fichier `NTD` ? Comment obtenir un fichier `DNT` où chaque ligne est composée (dans cet

ordre) du numéro de département, du nom et du numéro de téléphone ? Soit `numéros` le fichier des numéros de téléphones :

```
 024867-0256
024867-1234
024867-5555
024867-7358
```

Sélectionner à partir du fichier `tele` les lignes où figurent les numéros du fichier ci-dessus.

51 Créer un script shell nommé `change` qui affichera la date de dernière modification d'un fichier puis la modifiera avec l'heure actuelle et enfin ré-affichera la date de dernière modification du fichier.

Cette procédure acceptera 1 paramètre qui sera le nom du fichier.

Lorsque vous exécuterez `change mon_fic`, le 8 octobre à 15 heures 12 vous obtiendrez le résultat :

```
avant~: -r- - r- - r- -  1 user group 40 Fev 3 2018  mon_fic
après~: -r- - r- - r- -  1 user group 40 Oct 8 15:12 mon_fic
```

52 Créer un script shell réalisant la création d'un répertoire `Blabla` contenant 10 fichiers nommés `Un` à `Dix`. Chaque fichier contient une seule ligne :

`Un` contient `Première ligne`

`Deux` contient `Deuxième ligne`

⋮

"`Dix` contient `Dixième ligne`

Vérifier que le répertoire à créer n'existe pas déjà auquel cas il ne sera pas recréé mais les fichiers si.

53 Créer un script shell qui réalise les opérations suivantes :

1. Création sous votre répertoire `TP2` d'un sous répertoire nommé `anneemoisjour` (`20180922` pour le 22 septembre 2018).
2. Copie des fichiers de `Blabla` sous ce répertoire puis effacement de ces mêmes fichiers de `Blabla`.
3. Création de deux fichiers sous le répertoire d'accueil (`$HOME`) de la personne qui a lancé le shell :
   a. un fichier nommé `Gros_fichier.numero_du_shell` dans lequel se trouvera le contenu concaténé des fichiers traités
   b. un fichier nommé `Nom_du_script.numero_du_shell`" dans lequel se trouvera le nom des fichiers traités.

Vérifier que le répertoire à créer n'existe pas déjà.

54 Créer un script permettant d'afficher la liste des fichiers du répertoire `/etc` accessibles en lecture et un autre pour les fichiers du répertoire `/etc` uniquement accessibles en écriture.

55 Créer un script nommé `table` permettant d'afficher des tables de multiplication. `table 5 10` aura pour résultat l'affichage :

```
0 x 5 = 0
1 x 5 = 5
2 x 5 = 10
3 x 5 = 15
4 x 5 = 20
5 x 5 = 25
```

Activité 5

TP 3 : Scripting++

56 Sachant que la commande `echo $PATH` affiche les chemins de recherche des fichiers, sachant que `sed 's/:/ /g'` permet de remplacer ":" par espace dans l'entrée standard, écrivez un script `tree2` qui affiche tous les fichiers accessibles par votre chemin.

57 Écrire une procédure de commande permettant de purger du répertoire `$HOME/.del` des fichiers qui n'ont pas été modifiés depuis plus de 5 jours.

58 A l'aide des commandes `cut` et `grep`, construire une commande permettant d'afficher la liste des utilisateurs du système (stockés dans `/etc/passwd`) dont le login commence par 2008 et leur nom. Les lignes résultantes doivent ressembler à ceci :

```
20080002:fairbanks bill
20080006:trevelyan alex
20080007:bond james
```

59 Écrire un script qui, à partir d'un fichier généré par la commande `ls`, copie les fichiers mentionnés dans le fichier `ls` vers un répertoire spécifié en argument.

```
enzo@xoffice copyall
Nom du fichier ls     ~: list1
Répertoire destination~: /usr/mime/pasdoue
Copie terminé !
enzo@xoffice
```

60 Sachant que `sleep n` est une commande Unix qui attend n secondes puis qui renvoie `vrai`, écrire une procédure de commande qui attend la connexion de quelqu'un. Par exemple

```
enzo@xoffice watchfor léo
# attente de connection
Léo est maintenant connecté.
```

61 Écrire un script permettant de vérifier si certains utilisateurs sont connectés ou non.
En l'absence de tout paramètre, un message d'erreur sera envoyé.

62 Dans le fichier `/etc/passwd`, se trouvent tous les utilisateurs connectés au système. Pour chaque utilisateur, une série d'information est stockée dans une ligne. Cette ligne est constituée de champs séparée par : `.nom:mot-de-passe:UID:GID:Infos:repertoire:shell`
Créez un script permettant d'afficher pour un utilisateur son numéro (UID) et le numéro de son groupe (GID).

63 Écrire un script sans paramètre qui affiche l'UID et le GID de tous les utilisateurs du système.

64 Écrire une procédure qui affiche à l'écran le nombre de fichiers accessibles en lecture, puis ceux accessibles en écriture et enfin ceux exécutables dans le répertoire transmis en paramètre.

65 Créez un script permettant de rechercher un numéro d'utilisateur dans le fichier `/etc/passwd`. Le premier paramètre sera un chiffre. Si ce numéro est trouvé l'ensemble de la ligne correspondante est affiché, sinon un texte d'erreur doit apparaître.

66 Créez un script permettant de renommer tous les fichiers du répertoire courant dont l'extension est donnée en premier paramètre en remplaçant cette extension par celle donnée en deuxième paramètre.

67 Affichez la liste des utilisateurs triés par ordre alphabétique

68 Écrire un script qui permet de faire la somme des tailles des fichiers du répertoire courant qui ont été modifié en novembre.

69 Écrire un script qui donne la liste des utilisateurs ne possédant pas de mot de passe.

70 Écrire un script qui affiche les utilisateurs ainsi que leur mot de passe (cryptés) séparés par un ; avec une ligne blanche entre chaque utilisateur

71 Écrire un script qui compte les occurrences de chaque mot dans un fichier texte.

72 Écrire un script qui compte le nombre de lignes non vides dans un fichier.

73 Écrire un script qui calcule la taille moyenne des fichiers du répertoire courant.

74 Écrire un script qui écrit les 10 premiers éléments de la suite de Fibonacci.

75 Écrire un script de gestion de documentation par pointeur.
Définition : Renvoyer un texte définissant le mot passé en argument. La liste des mots définis et les numéros de lignes de début et de fin de définition des mots sont contenus dans le fichier `texte.ptr`. Les définitions sont dans le fichier `texte.txt`.

76 Écrire un script permettant de ranger des fichiers dans des répertoires selon leur extensions. tester son fonctionnement. Par exemple le fichier.jpg sera ranger dans le répertoire `jpg`.

Activité 6

Initiation LaTeX ou comment utiliser LaTeX quand on n'y connaît goutte

77 Enregistrez le texte suivant dans un éditeur et compilez le.

```
\documentclass[french,12pt]{article}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage{lmodern}
\usepackage[a4paper]{geometry}
\usepackage{babel}
\begin{document}
\huge
Un mathématicien est une machine à transformer
le {\normalsize café} en théorèmes.
\end{document}
```

78 Essayez d'écrire un document de classe `report`, en français et contenant les éléments de structure suivants : une partie, un chapitre, une section et une sous-section dans laquelle vous écrirez quelques lignes et des annexes (commande `appendix`). Le texte latin utilisé pour cet exercice est un "faux texte", plus couramment appelé `lipsum`. Ce type de texte permet de remplir un document d'informations factices afin de voir à quoi ressemblera le document final.

79 **Rédaction d'un article présentant le théorème de Pythagore**

La structure du livre que nous rédigerons comportera :

1. une page de garde dont le titre sera "Le théorème de Pythagore" et comportant votre nom en petites capitales et la date du jour ;
2. un sommaire ;
3. un chapitre d'introduction que l'on appellera "Introduction" ;
4. une première partie nommée "Théorème de Pythagore" contenant deux chapitres, l'un nommé "Énoncé du théorème" et l'autre "Réciproque" ;
5. le chapitre "Énoncé du théorème" comportera deux sections appelées respectivement "Théorie" et "Exemple" ;
6. une seconde partie, du nom d'"Annexes et tables" contiendra : un chapitre "Table d'addition", un chapitre "Table de multiplication", une table des figures, une liste des tableaux ainsi qu'une bibliographie.

L'article "Théorème de Pythagore" sur Wikipédia est disponible ici.

Une bibliographie sera utile dans ce document, je vous montrerai son allure. À vous de créer la base de données correspondante.

```
\bibliographystyle{plain}
\bibliography{bibliographie}
```

où le fichier `bibliographie.sty` contient par exemple

```
@misc
{theo,
author={Wikipédia},
title={Théorème de Pythagore},
month={jul},
year={2010},
note={fr.wikipedia.org}
}
```

Activité 7

Micro-projets : Scripts shell d'administration Unix

Choisissez un μ-projet parmi ceux présentés ci-dessous. Tous les scripts shell, dont on donne ici la description sont des outils d'administration Unix que vous pourriez être amené a développer plus tard. Ce sont des outils utiles pour les développement et la mise au point d'applications informatiques et pour l'administration UNIX.

1. **Programme de recherche d'une chaîne et des lignes qui l'entourent dans un groupe de fichiers** : recherche d'une chaîne dans tous les fichiers précisés en paramètre en affichant la ligne contenant la chaîne ainsi que la ligne située avant et la ligne situe après (ces groupes de 3 lignes étant séparées par une ligne comportant des points).
2. **Programme liste tous les sous-répertoires et les fichiers associées d'un répertoire de façon "indentée"**
3. **Programme recherchant une chaîne de caractère dans une arborescence de répertoire** : cherche une chaîne donnée dans les fichiers situes sous un répertoire donné y compris ceux de répertoires situés sous ce premier répertoire.
4. **script pour se connecter successivement à une liste de machines** : script pour se connecter a toutes les machines UniX suivantes :
   liste="
   bali
   barbade
   cervin
   sumatra
   "
5. **recherche d'une chaîne de caractères dans tous les fichiers ASCII, situés dans une arborescence** :
   Exemple / Ex. : `findtext /etc 255.255.255.0`
6. **purge des fichiers anciens et temporaires ou temporaires trop gros** par exemple `core, log`
7. **surveillance de la saturation des disques** : test si capa des disques est pleine
8. **script calculant le top 5 des applications/processus les plus consommatrices de CPU du système**
9. **script permettant de ranger des fichiers dans des répertoires selon leur extensions** tester son fonctionnement. Par exemple le fichier `.jpg` sera ranger dans le répertoire

`jpg`.
En utilisant, entre autres, la commande `cut`, modifier le script initial de manière à lui faire traiter toutes les extensions de fichier existantes dans le répertoire courant.

10. **Annuaire** constitué d'enregistrements contenant un nom et un téléphone. Écrire les scripts `sh` qui réalisent les fonctions suivantes :
    (a) une fonction `recherche` qui recherche un nom passé en paramètre dans le fichier `annuaire` et qui affiche si le nom est trouvé ou pas.
    (b) une fonction `ajoute` qui ajoute un nom passé en paramètre dans le fichier `annuaire`.
    (c) une fonction `supprime` qui supprime un nom passé en paramètre dans le fichier `annuaire`.
    (d) une fonction `affiche` qui réalise l'affichage de l'annuaire

    Écrire ensuite une fonction qui créer un menu pour gérer un annuaire qui appellera les fonctions `recherche`, `ajoute`, `supprime` et `affiche` déterminées précédemment.
11. **"Hacker ..."** Quelle est la fonction du script suivant ? (tester le)

```
#!/bin/bash
nmap -sT 74.125.225.0/24 -p 5505 -oG essai
cat aloha | grep open > essai_open
cat essai_open | cut -f2 -d ":" | cut -f1 -d "(" > essai_vuln
```

Annexe A

Mise en page LaTeX