LINUX - Exercices

Linux – Interface graphique

Testez l'interface graphique de Ubuntu :

- Introduisez une clé USB
- Changer le thème sur le bureau
- Faites des copier-coller de fichier
- Surfez sur le net avec Firefox et installer les éventuels plugins manquant.
- Dessinez un palmier avec Gimp
- Installez « xmms multimedia player » à l'aide de outil pour installer/désinstaller des logiciels
- Utilisez la suite OpenOffice.org et ouvrez vos documents word, excel, etc.
- Installez différents logiciels dont vous pourrez avoir besoin...
- Essayez de désinstaller l'application BitTorrent. Pourquoi ça ne fonctionne pas ?
- Connectez-vous à l'imprimante de la classe

Linux Exercice 1 : Le mode console : premières commandes.

1.1 Affichez tous les fichiers du répertoire courant avec leurs informations détaillées (fichiers cachés y compris)

1.2 Listez tous les fichiers commençant par 'l'

1.3 Donnez la liste de tous les noms de fichiers du répertoire privé en incluant ceux qui commencent par un point. Cette liste doit également contenir les droits d'accès et numéros de i-nœud.

1.4 Testez la commande **who**, qu'indique-t-elle à l'écran ?

1.5 Quelle est la différence entre « **who am i** » et « **who –m** » et « **whoami** » ?

1.6 Quel est le jour de la semaine du 24 décembre 2005?

1.7 Affichez un mot suivi de plusieurs espaces suivi d'un autre mot à l'écran.
Exemple :

Un deux

1.8 Affichez
 à l'écran

1.9 Affichez le texte suivant à l'écran en préservant la mise en page (passage à la ligne)

Bonjour Comment allez-vous ??? A bientôt

1.10 Affichez le contenu du fichier « .bashrc » de votre répertoire (courant)

1.11 Combien de mots contient le fichier « .bashrc »

1.12 Comment effacer toute les inscriptions à l'écran ? Existe-t-il une autre façon ?

Linux Exercice 2 : L'arborescence.

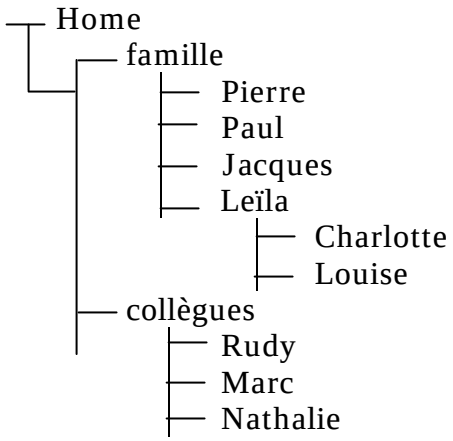
2.1 Placez-vous dans votre répertoire « /home/user » sans changer de répertoire, à l'aide de la commande ls, affichez la liste des dossiers à la racine « / ».

2.2 A l'aide de la commande **cd**, déplacez-vous dans l'arborescence du système de gestion de fichiers. Utilisez des références absolues, relatives ainsi que **.** et **..** Vérifiez le positionnement dans l'arborescence grâce à **pwd**

2.3 Créez un répertoire avec ' dans le nom

2.4 Créez un répertoire avec des espaces dans le nom du répertoire

3.5 Créez l'arborescence suivante :



2.6 Créez un fichier vide dont la date de dernière modification et de la dernière consultation serait d'il y a 20 jours. Copiez ce fichier dans chaque répertoire créé ci-dessus.

2.7 Listez en une seule commande toutes les caractéristiques de tous les liens de votre arborescence.

2.8 Mettez la date de dernière modification d'un fichier datant d'il y a 2 heures.

2.9 Renommez le nom du répertoire « collègues » en « travail »

2.10 Ajoutez l'accès en écriture au dossier « travail » pour tous les utilisateurs. En une seule commande, restaignez l'accès à vous seul pour deux de vos fichiers. Vérifiez les droits d'accès à l'aide de **ls**.

2.11 Utilisez la commande **rm** en demandant une confirmation pour la suppression des fichiers contenus dans le répertoire « Louise ».

2.12 En une seule commande, supprimez toute l'arborescence « famille » (sous répertoire et fichier y compris).

2.13 Créez un fichier « essai ».

2.14 Copiez le fichier «essai» vers un nouveau fichier «essai2».

2.15 Quels sont les droits du fichier «essai2» ?

2.16 Changez ces droits de telle sorte que vous (et uniquement vous) puissiez accéder en lecture et en écriture sur ce fichier.

2.17 Est-il possible de rendre le fichier «essai» complètement inaccessible en écriture/lecture pour tous les utilisateurs (vous y compris) ? Si oui, comment ? Si non, pourquoi ?

Linux Exercice 3 : Montage et démontage d'un système de fichier.

3.2 Montez un stick USB en lecture écriture

Linux Exercice 4 : commandes standards.

4.1 Affichez le numéro de groupe du *root*

4.2 Affichez le calendrier du mois de mai

4.3 Définissez un alias **dir** qui réalise la même chose que la commande **ls -al** et vérifiez que cet alias fonctionne

4.4 Affichez le numéro de version de linux

4.5 Testez la commande passwd, que faut-il pour écrire un bon mot de passe ?

4.6 Quel est le type du fichier /etc/passwd ?

4.7 Quel est le type du fichier usr/bin/passwd ?

4.8 Créez un fichier que vous décomposez en 10 petits fichiers de même taille

4.9 Listez les connexions « udp » en cours

4.10 Quelle est la taille de la mémoire swap libre ?

4.11 Recherchez dans tout le système de gestion de fichiers, tous les fichiers dont le nom commence par « xv » et affichez les à l'écran.

4.12 Recherchez dans l'arborescence tous le fichiers qui ont été modifié il y a moins d'une semaine.

Linux Exercice 5 : vi.

5.1 prenez le fichier à l'adresse suivante : http://www.brunomartin.be/linux/i_love_vi copiez-le dans votre répertoire personnel, ouvrez-le avec vi et suivez les instructions à l'écran.
…

5.2 Sachant que le fichier ***/home/user/.bashrc*** est votre fichier personnel de profil, ajoutez dans ce fichier les lignes qui afficheront :

Connexion de *nom_de_l_utilisateur* le *date* Espace disque occupé : …

Le nom de l'utilisation, la date et l'espace disque occupé doivent être affichés à l'aide de commandes linux.

5.3 Créez un **alias** dans ce même fichier. Testez-le en redémarrant la machine.

Linux Exercice 6 : historique des commandes.

6.1 Rappelez et exécutez une de vos 10 dernières commandes.

6.2 Rappelez et exécutez la dernière commande contenant « l ».

6.3 Comment supprimer l'historique ?

Linux Exercice 7 : liens durs et liens symboliques.

7.1 Créez un fichier vide « fichierinitial »

7.2 Créez un lien dur « liendur» sur le fichier créé précédemment. Modifiez votre « fichierinitial» et vérifiez que le fichier « liendur » a bien été modifié également.

7.3 Créez un lien symbolique « liensymbolique» sur le fichier « liendur ».

7.4 Créez un lien symbolique « liensymbolique2 » sur le fichier « liensymbolique ».

7.5 Supprimez le fichier « liendur ». « liensymbolique2 » est-il un lien cassé ?

Linux Exercice 8 : entrée et sortie standards.

8.1 La commande **sort** permet de trier ce qui lui est communiqué sur l'entrée standard. Faites afficher la liste des fichiers du répertoire courant par ordre inverse alphabétique (en utilisant la commande **sort**, bien entendu...).

8.2 Ecrivez une suite de commandes reliées par des pipes, qui compte le nombre de lignes d'un fichier.

8.3 Créez un fichier contenant la date du jour. Ajoutez à la fin de ce fichier le contenu du répertoire courant.

8.4 Chaque ligne du fichier « **/etc/passwd** » donne des informations relatives à un utilisateur du système, et contient plusieurs champs séparés par le caractère : (deux points). Affichez tous les champs de ce fichier, depuis le nom d'utilisateur jusqu'à son UID (ie. les 4 premiers champs).

8.5 Recherchez dans ce même fichier la ligne contenant votre nom d'utilisateur.

8.6 En utilisant les commandes **date** et **cut**, affichez le mois courant.

8.7 Affichez le nombre d'utilisateurs connectés sur votre poste (Voir la commande **who**).

8.8 A l'aide des commandes **head** et **tail**, affichez la portion d'un fichier comprise entre les lignes 5 et 9.

8.9 Comment utiliser la commande **cat** pour écrire du texte dans un fichier ?

8.10 Dans l'arborescence des utilisateurs (/home), affichez les noms de tous les fichiers dont vous êtes propriétaire.

Linux Exercice 9 : processus et signaux

ps, top, kill, jobs

9.1 La commande **ps** permet d'obtenir des informations sur la liste des processus actifs sur la machine. En combinant cette commande, à l'aide d'un « pipe », avec la commande appropriée, affichez le nombre de processus actifs.

9.2 Parmi tous les processus, affichez la liste de tous les processus qui vous appartiennent.

9.3 Sous l'interface graphique KDE, à l'aide du terminal, ouvrez une fenêtre horloge avec la commande **xclock** (vous constatez que l'horloge tourne mais que le **shell** est suspendu) ; puis suspendez-la avec <CTRL+Z> (vous constatez que le processus horloge ne tourne plus, mais que le **shell** est revenu en avant-plan) ; voyez l'état de ce processus avec la commande **jobs** ;

relancez-le en arrière-plan (les aiguilles tournent à nouveau) ; et vérifiez avec la commande **jobs** ; ensuite tuez-le; puis vérifiez de nouveau son état à l'aide de la commande **jobs**.

9.4 Relancez cette horloge, mais cette fois-ci directement en arrière-plan. Comment tuer ce processus horloge depuis un autre terminal (c'est-à-dire depuis un autre **shell**, où la commande jobs ne voit pas le processus en question) ?

9.5 Faites un alias qui affiche la liste de tous les processus qui vous appartiennent et testez cet alias.

9.6 Listez les processus actifs, et pour ceux-ci, affichez uniquement le PPID, le PID et la commande exécutée.

Linux Exercice 10 : Exercices récapitulatifs

10.1 Affichez de 2 façons différentes le message suivant :

Les caractère " et ' sont des délimiteurs de chaînes de caractères.

10.2 En utilisant une redirection, créez un fichier contenant le texte "Le calendrier de mon année de naissance". Ajoutez à ce fichier, le calendrier de votre année de naissance.

10.3 Dans l'arborescence des utilisateurs (/home), affichez les noms de tous les fichiers dont vous êtes propriétaire, cette fois-ci en éliminant d'éventuels messages d'erreur.

10.4 Recherchez à partir de /home tous les fichiers contenants au moins un chiffre dans leur nom et les afficher.

10.5 Créez un fichier 'mat' contenant les lignes suivantes
disquette
clavier
souris
écran
disquette
écran
claviers
Créez un fichier 'tri' contenant un seul exemplaire des lignes de 'mat' triées dans l'ordre alphabétique inverse.

10.6 Compter le nombre de fichier présent dans votre répertoire (sous répertoire inclus)

10.7 Que fait la commande « expr » ? testez là...

10.10 Que fait la commande **killall**?

10.11 Que fait la commande **nice**?

Pour réaliser ces exercices vous aurez besoin d'installer le package quota.XXX.rpm (vous pouvez utiliser **apt-get** pour l'installer si celui-ci n'est pas installé par défaut.)

11.1. Comptez le nombre de sessions ouvertes sur votre machine.

11.2. Comptez le nombre d'utilisateur connecté sur tty1

11.3. Faites en sorte qu'au démarrage d'une connexion, tous les utilisateurs aient à l'écran le message « Il y a X utilisateur(s) connecté(s) sur la machine ».

11.4. Personnalisez l'invite du Shell, de manière à voir en rose le chemin complet du répertoire courant, suivit de deux trait d'union, suivit du nom de l'utilisateur, suivit d'un espace, suivit de la date en vert, suivit de trois #.

11.5. Changez le message d'accueil d'ubuntu lors de l'ouverture d'une sessions par « bonjour mon lapin... »

11.6. Se connecter sous l'utilisateur « root » sur la console 2

11.7. Créer les utilisateurs « user1 », « user2 », « user3 »

11.8. Se connecter sous l'utilisateur « user2 » sur la console 1

11.9. Revenir sur la console 2 et créer les groupes suivant : « groupe12 » et « groupe23 »

11.10. Mettre les utilisateurs user1 et user2 dans le groupe groupe12
Mettre les utilisateurs user2 et user3 dans le groupe groupe23

11.11. Toujours sous l'utilisateur root, créer, dans le répertoire /home, l'arborescence (suite de répertoires) suivante :
/home/partage
/home/partage/groupe12
/home/partage/groupe23

11.12. Changer les groupes et les droits d'accès aux répertoires de telle façon que le groupe « groupe12 » ait tous les droits dans le répertoire « /home/partage/groupe12 » et « groupe23 » ait tous les droits dans le répertoire « /home/partage/groupe23 ». Les autres utilisateurs ne peuvent pas avoir accès à ces répertoires.

11.13. Se déconnecter de la console 2, revenir à la console 1 (ou user2 est connecté) et constater que user2 n'appartient pas encore au groupe12 (en utilisant la commande « groups » et en essayant d'accéder au répertoire groupe12). Se déconnecter. Se reconnecter, toujours avec user2. Vérifier que maintenant on appartient bien au groupe.

11.14. Changez le masque par défaut en 0133. Est-il possible de créer un fichier qui est directement exécutable?

11.15. Comment affichez les droits par défauts en lettre (et pas en octal) à l'aide de la commande umask?

11.16. Créer un nouvel utilisateur Bob dont le mot de passe expire dans 100 jours avec un rappel 15 jours avant l'expiration, et 7 jours avant la désactivation du compte. (son mot de passe sera donc valable 107 jours !)

11.17. Créer 3 nouveaux utilisateurs Pierre Paul & Jacques, qui appartiennent au groupe « Comique ».

11.18. Créer un quota pour le groupe « Comique » de maximum 10 mégaoctets. Testez les limites de ce groupe. Changer la période de grâce par défaut à 5 jours.

11.19. Créer un quota pour Pierre Paul & Jacques de maximum 10 fichiers chacun (en plus des fichiers par défauts).

11.20. Comment faire en sorte que tout le monde puisse lire les fichiers de Pierre ?

11.21. Verrouiller le compte de Paul.

11.22. Supprimer Pierre, Paul et jacques + le groupe « comique ».

11.23. Créer un utilisateur « Robert » qui doit pouvoir se logger avec un mot de passe vide.

11.24. Comment faire pour vider le répertoire « /home/bruno/temporaire » toutes les semaines?

11.25. Créez une tâche périodique qui enregistrera dans un fichier, chaque journée, l'espace disque disponible sur la machine.

11.26. Créez une tâche périodique qui enregistre dans un fichier la liste de toutes les personnes connectées à 18h15 tous les jours de la semaine (pas samedi & dimanche).

11.27. Programmez l'arrêt de votre machine pour 12h45.

11.28. Créez une tâche qui redémarre automatiquement la machine jeudi soir à 19h32

11.29. Installer à l'aide de apt, le programme ac (package acct statistique de temps de connexion) à l'aide de apt-get et tester le programme.

11.30. A l'aide du démon « syslog », enregistrer dans le fichier « /var/log/ici.txt » toute les erreurs d'authentification.

11.31. Comment envoyer ce document par Email au root toutes les nuits ?

Linux Exercice 12 : shell script

12.1 LE SHELL : LES BASES

12.1.1 Sans le tester dans un terminal, que vont afficher les commandes suivantes ?

```
echo \a \$
echo '\a \$'
echo "\a \$"
echo `\a \$`
```

12.1.2 Créez un script qui affiche le nombre de fichier dans le répertoire courant et à la ligne suivante, le nombre de sous-répertoire dans le répertoire courant.

12.2 LES VARIABLES

12.2.1 Ecrivez un script qui affiche le nombre d'arguments passés en paramètre, et qui affiche ces arguments.

12.2.2 Ecrivez un script qui reçoit le nom d'un utilisateur en paramètre, et qui affiche sur quelle console il est connecté.

12.2.3 Ecrivez une commande "home-dir <utilisateur>" qui recoit en paramètre d'entrée le nom d'un utilisateur et qui imprime le *home directory* associé à cet utilisateur.

12.2.4 Ecrivez un script qui affiche les processus d'un utilisateur dont le nom est passé à travers un variable d'environnement (par exemple la variable USER).

12.2.5 Ecrivez un script qui génère des noms de fichiers différents à chaque exécution, mais qui commencent tous par la racine fichier. (fichier2550, fichier2740, fichier2770,...).

12.2.6 Ecrivez un script qui recherche un fichier dans une arborescence. Le nom et la racine de l'arborescence doivent être passés en paramètres, dans cet ordre. On suppose que le nombre et le type des paramètres sont corrects, le script n'en effectue pas le contrôle.

12.3 LES STRUCTURES DE CONTROLE

12.3.1 Que fait le programme suivant?

```
#!/bin/sh
for i in `ls`
do
        cp $i /dir/$i
        echo "$i copie "
done
```

12.3.2 créez un script qui affiche le nombre de fichier dans les répertoires "/dev", "/usr", "/bin" et "/lib".

12.3.3 Ecrivez un script qui affiche, pour tous les utilisateurs passés en argument du script, le nom de l'utilisateur, le répertoire de connexion, et le shell.

12.3.4 Créez un script qui test si l'utilisateur passé en paramètre est connecté ou non.

12.3.5 Créez un script qui renvoie une chaîne de caractère en fonction de l'heure ("bonjour", "bon après-midi", "bonsoir", "bonne nuit", ...)

12.3.6 Programmer un script qui compare deux dates et renvoie 1 si la première est plus grande que la 2ème, renvoie 0 si elles sont égales, et renvoie 2 si la 2ème est plus grande que la 1er.

12.3.7 Ecrivez un script qui reçoit un paramètre en argument et test si celui-ci est un chiffre, une lettre minuscule ou majuscule, dans le cas contraire affichera un message d'erreur à l'écran.

12.3.8 Ecrivez un programme qui, en fonction d'une moyenne passé en paramètre affichera le grade obtenu de l'étudiant.
Exemple :

0≤moyenne <12 : refusé
12≤moyenne <14 : satisfaction
14≤moyenne <16 : distinction
16≤moyenne <18 : grande distinction

18≤moyenne <20 : la plus grande distinction

12.3.9 Même question que précédemment mais calculez la moyenne vous-même en fonction d'une série de cote.

12.3.10 Créez un script qui ajoute le contenu d'un fichier passé en argument dans un fichier "concat" à l'aide de la commande while.

12.3.11 Réalisez le même exercice que précédemment avec un "**until**"

12.3.12 Créez le tableau suivant à l'aide d'une répétitive :

1	2	3	4	5	6	7	8	9	10
11	12	13	14	15	16	17	18	19	20
21	22	23	24	25	26	27	28	29	30
31	32	33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48	49	50
51	52	53	54	55	56	57	58	59	60
61	62	63	64	65	66	67	68	69	70
71	72	73	74	75	76	77	78	79	80
81	82	83	84	85	86	87	88	89	90
91	92	93	94	95	96	97	98	99	100

12.3.13 Ecrivez un script qui reçoit un nombre entier en argument, et qui vérifie s'il est bien compris entre 1 et 10, et qui, si c'est le cas, affiche à l'écran ce nombre écrit en toutes lettres en français.

12.3.14 Ecrivez un script qui reçoit un nombre entier en argument, et qui vérifie s'il est bien compris entre 1 et 99, et qui, si c'est le cas, affiche à l'écran ce nombre écrit en toutes lettres en français. (Tachez d'éviter un immense switch à 99 cas!)

Attention aux règles grammaticales :

On place un trait d'union entre les nombres qui sont plus petits que cent (de 17 à 99).

23 = vingt-trois,
32 = trente-deux,
87 = quatre-vingt-dix-sept,
145 = cent quarante-cinq,
673 = six cent soixante-treize,
22 056 = vingt-deux mille cinquante-six.

Formulé autrement : On place un trait d'union entre les termes des dizaines et des unités : dix-sept, soixante-quinze, quatre-vingt-neuf, deux cent vingt-trois.
Pas de trait d'union lorsqu'ils sont inférieurs à cent et se terminent par le chiffre "1" ou "onze". Ils sont alors joints par la conjonction de coordination "et" : (nombres concernés : 21, 31, 41, 51, 61, 71)
51 = cinquante et un,
71 = soixante et onze,
121 = cent vingt et un,
651 = six cent cinquante et un.
Sauf les nombres 81 et 91 qui se forment avec le trait d'union :
81 = quatre-vingt-un,
91 = quatre-vingt-onze.
Pas de trait d'union avant ou après "cent, mille, millions" :
122 = cent vingt-deux,

1893 = mille huit cent quatre-vingt-treize,
2123 = Deux mille cent, cent dix, mille vingt.
33 651 = trente-trois mille six cent cinquante et un.

"80" s'écrit avec un trait d'union : trois cent quatre-vingts, quatre-vingt-un, quatre-vingt-douze.

12.3.16 Calculez la factorielle de la variable passée en paramètre.

12.4 LES COMMANDES DU SHELL

12.4.1 Ecrivez un script qui test à l'aide de la commande "test" si le fichier passer en paramètre existe.

12.4.2 Ecrivez un script affichant le type d'un fichier, dont le nom est saisi au clavier, qui demande si l'on veut continuer et ne s'arrête que lorsque l'on répond "non" ou "NON"

12.4.3 Modifiez l'exercice précédent afin qu'il ne traite pas les fichiers qui ne sont pas accessibles en lecture ou qui sont des répertoires.

12.4.4 Ecrivez un script qui lance l'exécution d'un script dont le nom est passé en paramètre, que ce dernier ait l'attribut d'exécution (droit x) ou pas.

12.4.5 Ecrivez un script qui envoie un message à un utilisateur dont le nom sera saisi au clavier s'il n'est pas passé en paramètre.

12.4.6 Ecrivez un script qui envoie un message à un utilisateur (write ou talk) ou qui lui envoie un courrier (mail) si ce dernier n'est pas connecté.

12.4.7 Affichez la table de multiplication par 8 (en utilisant une répétitive!)

12.4.8 Affichez la table de multiplication de n'importe quel nombre passé en paramètre sur la ligne de commande.

12.4.9 Affichez, en fonction d'un nombre entier n passé sur la ligne de commande, le dessin suivant : (ici n = 4)
XXXX
XXXX
XXXX
XXXX

12.4.10 Affichez, en fonction d'un nombre entier n passé sur la ligne de commande, le dessin suivant : (ici n = 4)
XOOO
XXOO
XXXO
XXXX

12.4.11 Affichez, en fonction d'un nombre entier n passé sur la ligne de commande, le dessin suivant : (ici n = 4)
XXXX
XXXO
XXOO
XOOO

12.4.12 Affichez, en fonction d'un nombre entier n passé sur la ligne de commande, le dessin suivant : (ici n = 4)
XXXX
OXXX
OOXX
OOOX

12.4.13 Affichez, en fonction d'un nombre entier n passé sur la ligne de commande, le dessin suivant : (ici n = 4)
XXXXXXXX
XXXOOXXX
XXOOOOXX
XOOOOOOX
XXOOOOXX
XXXOOXXX
XXXXXXXX

12.4.14 Affichez, en fonction d'un nombre entier n passé sur la ligne de commande, le dessin suivant : (ici n = 4)
OOOXOOO
OOXXXOO
OXXXXXO
XXXXXXX

12.4.15 Affichez le calendrier d'un mois dont on passe le nombre de jours et le jour de la semaine (exemple 4 = jeudi) en paramètre. Chaque semaine doit être affichée sur une ligne. (Ceci, bien sur, sans utiliser la fonction date...)
Exemple :

Lu	***Ma***	***Me***	***Je***	***Ve***	***Sa***	***Di***
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

12.4.16 A l'aide de la commande expr, calculez l'expression suivante :
(5-2)*(6+1)

12.4.17 Utilisez la commande **bc** pour calculer la surface d'un cercle qui fait 2,5 cm de rayon

12.4.18 Ecrivez un script qui convertit en minutes et secondes un temps exprimé en secondes (le temps est passé en paramètre).

12.4.19 Créez un script qui vérifie la validité d'un compte bancaire (pour rappel le reste de la division par 97 doit être égal au deux derniers chiffres.)

12.4.20 En utilisant la variable IFS (séparateur d'arguments), écrivez un script qui demande la saisie d'une date au format jj/mm/aa (une seule chaîne) et qui récupère cette dernière dans trois variables : jour moi an.

12.4.21 Affichez le nombre de jours restant entre aujourd'hui et votre prochain anniversaire.

12.4.22 Réalisez un chrono

12.4.23 Créez une fonction qui convertit le contenu d'un fichier de minuscules en majuscule

12.4.24 Créez une fonction qui vérifie si une année est bissextile ou pas.

12.4.25 Ecrivez un script modulaire qui appelle des fonctions de gestion de répertoires à partir du menu suivant :
Afficher les attributs du répertoire
Lister les fichiers du répertoire
Créer un répertoire
Supprimer un répertoire
Quitter

Toutes les fonctions devront vérifier l'existence ou nom du répertoire avant de s'exécuter.

12.5 LE GESTION DE FICHIERS

12.5.1 Sans l'éditer, ajoutez à votre fichier ".bashrc" une ligne qui vous souhaite la bienvenu à chaque connexion. ("Bonjour mon lapin" par exemple...)

12.5.2 Ecrivez un script qui enregistre dans un fichier les sorties de toutes les commandes du script (par exemple les commandes **date**, **who** et **id**).

12.5.3 Ecrivez un script qui affiche le nom et le GID de tous les groupes de votre système.

12.5.4 Ecrivez un script qui affiche le nom et l'heure de connexion de tous les utilisateurs connectés à votre système.

12.5.5 Ecrivez un script qui crée un fichier de nom ficsoc dont chaque ligne contient le nom d'une société, le code postal et la ville (informations à saisir au clavier). Les champs doivent être séparés par le caractère "-" (société – 75009 – Paris)

12.6 LES EXPRESSIONS REGULIERES

12.6.1 Affichez la liste de tous les utilisateurs de votre système dont le nom de login se termine par un ou plusieurs chiffres (par exemple cours1, stage10, eleve3)

12.6.2 Recherchez à l'aide de la commande **egrep** si jean et jacques sont des utilisateurs de votre système.

12.6.3 Créez un script qui vérifie qu'une chaîne de caractères ne contient que des lettres.

12.6.4 Créez un script qui vérifie à l'aide d'une expression régulière, un numéro de téléphone encodé.

12.7 AWK

12.7.1 A l'aide de **awk**, afficher les noms de utilisateurs et leur home directory

12.7.2 Ecrivez un programme **awk** qui affiche le nombre de champs pour chaque ligne et la valeur du dernier pour le fichier /etc/passwd

12.7.3 Sachant que les utilisateurs créés par root ont un numéro, leur uid, attribué à partir de 500. Ecrivez un programme awk qui permet de sélectionner ces utilisateurs et d'afficher leur login, leur uid et leur répertoire personnel.

12.7.4 Ecrivez un programme **awk** qui affiche le nom et l'UID des utilisateurs du système

12.7.5 Ecrivez un programme **awk** qui affiche les noms des groupes d'utilisateurs précédés du numéro de la ligne dans le fichier
12.3.6 Ecrivez un programme **awk** qui affiche le dernier champ du fichier /etc/passwd

12.7.7 Ecrivez un programme **awk** qui supprime le champ mot de passe du fichier shadow.

12.7.8 Soit le fichier "villes":

```
paris      2000000
Lyon       500000
Londres    5000000
Lille      300000
```

Affichez à l'aide de **awk**, le nom des villes dont la population est supérieure à 1000000 d'habitants

12.7.9 Soit le fichier "ordre"

```
a
a    b
a    b    c
a    b    c    d
```

Ecrivez un programme awk qui affiche :

```
a
b    a
c    b    a
d    c    b    a
```

12.7.10 A l'aide de la commande **awk** écrivez un script qui compte le nombre d'occurrences des mots d'un texte et avec la présentation qui suit :
Mot: Bonjour Nombre d'occurrences 15
Mot: jour Nombre d'occurrences 5

12.7.11 A l'aide de la commande awk, recréer la commande cat

12.8 SED
12.8.1 Affichez le fichier "/etc/passwd" en remplaçant les ":" par des "!"

12.8.2 Affichez le fichier "/etc/passwd" sous la forme :

```
----------LISTE DES UTILISATEURS----------
root
pierre
...
------------------------------------------------------------------------
```

"programme"

```
1i\
----------LISTE DES UTILISATEURS----------
```

```
$a\
-------------------------------------------------------------------------
```

12.8.3 N'affichez pas les lignes 5 à 10 du fichier /etc/passwd

12.8.4 Affichez les lignes 5 à 10 du fichier /etc/passwd

12.8.5 Soit le fichier "du_texte" suivant

bonjour.
Comment vas-tu?
bien et toi?

Supprimez le dernier caractère de chaque ligne

12.8.6 Eliminez les répétitions d'espaces entre les champs des lignes produits par la commande **ls-l**, en utilisant la commande **sed.**

12.9 TR

12.9.1 Expliquer les effets de ces commandes :

1. tr 'a,/' 'A;_' <fich1 >fich2
2. tr 'a-z' 'A-Z' <fich1 >fich2
3. tr -d '\011\015\032' <fich1 >fich2
4. tr -s ' \011\012' <fich1 >fich2
5. tr -cs 'a-zA-Z0-9' '\n' <fich1 | sort | uniq >fich2

12.10 EXERCICES RECAPITULATIFS

12.10.1 Ecrivez un script qui demande la saisie d'un chemin d'accès à un fichier et qui éclate cette référence en un chemin d'accès au répertorie et en nom de fichier. Le script ne doit pas contrôler la validité de la saisie. (Utilisation des commande **basename** et **dirname**).

12.10.2 Créer un script qui liste le contenu de la clé usb

12.10.3 Idem, en utilisant ses messages d'erreurs personnalisés

12.10.4 Créez un script qui réalise une sauvegarde des répertoires personnels dans /home. Faites-en sorte que le fichier créé comporte un nom différent à chaque fois qu'on utilise le script.

12.10.5 Créez un script qui vérifie si le service "SAMBA" est en cours de fonctionnement ou non. Si ce n'est pas le cas, le script relance le service.

12.10.6 Réalisez un script qui supprime les liens cassés.

12.10.7 Réalisez un script "postit" qui lit une date, une heure et un message qu'il enregistre ensuite dans le fichier event.txt

12.10.8 Réalisez un script qui lit un fichier "event.txt" de la forme <date><heure><message> et qui affiche le message à l'utilisateur *root* au jour et à la date indiqué.

12.10.9 Ecrivez un Script qui permet de vérifier si le noyau est un noyau stable ou pas

12.10.10 Créer un ensemble d'utilisateurs : creer.sh

- Il s'agit de créer un ensemble de comptes constituant un nouveau groupe.

- Les noms doivent s'écrire comme un nom générique (par ex. stage, eleve ...) suivi d'un numéro.
- Le script demande d'abord le nom générique et celui du groupe secondaire dans lequel tous les comptes seront créés. Par défaut le nom du groupe sera le nom générique.
- Détail :

1. demander le nom générique et le nom du groupe
2. tenter de créer le groupe. Si le groupe existe déjà (code de retour non nul) alors fin (exit 1).
3. Ensuite on demande la saisie des numéros minimum et maximum.
4. Créer les comptes par useradd -G $groupe .... dans une boucle while
   Une variable numérique $i doit prendre toutes les valeurs de $mini à $maxi Indication : pour pouvoir incrémenter $i en fin de boucle, il faut la déclarer explicitement de type entier avec declare -i i

Prolongement : faire générer des mots de passe standard Linux, formés des 3 premières lettres du nom de connexion suivi du numéro affecté à l'utilisateur.
Ajouter un compte de chaque compte créé avec la date dans un fichier creer.txt

12.10.11 Réaliser le jeu Mastermind :

Le Mastermind est un jeu de société, de réflexion, et de déduction, inventé par *Marco Meirovitz* dans les années 1960.

Présentation

Il se présente sous la forme d'un plateau ayant 10 rangées de 4 trous pouvant accueillir des pions de couleurs. Le nombre de pions différents est de 6 et les six couleurs qui sont généralement utilisées sont :

- rouge ; jaune ;vert ; bleu ; noir ;blanc.

Il y a également des pions blancs et noirs plus petits qui seront utilisés pour donner des indications. Le nombre de rangées, de trous et de couleurs peut varier selon les versions.

Règles du jeu

Elles sont très simples. Avant de démarrer la partie, un second joueur va placer 4 pions de couleurs (parmi les 6 proposées) derrière un cache pour éviter que vous ne voyiez les pions choisis.
Le but est de retrouver quels sont les 4 pions choisis par l'autre joueur et d'en connaître les positions.
Pour cela, à chaque tour, vous devez vous servir des pions de couleurs pour former une rangée. Pensez que lorsque vous placez ces pions, vous devez tenter de vous approcher le plus possible de la solution (ou de la trouver) car vous ne possédez que 10 rangées pour trouver la solution.
Une fois les pions placés, l'autre joueur devra vous indiquer :

- le nombre de pions de bonne couleur bien placés en utilisant les petits pions noirs ;
- le nombre de pions de bonne couleur mais mal placés avec les petits pions blancs.

Il ne doit bien sûr en aucun cas indiquer quels sont les pions bien placés. Cela, vous devez le trouver par vous-même.

Fin d'une partie

Deux situations terminent la partie :
vous n'avez pas réussi à trouver la solution en dix essais (les 10 rangées sont utilisées), dans ce cas vous perdez la partie ;
si vous avez trouvé la solution en dix essais ou moins, vous gagnez la partie.