

**Notes :**

- Tous les documents manuscrits (cours, notes de TP) et listings de TP sont autorisés (les livres sont interdits).
- Les exercices sont indépendants.

Exercice 1 – Un peu de Shell (5 pts)

On souhaite stocker dans un fichier *dispo.txt* l'espace disque restant dans le répertoire */home*. La commande permettant de récupérer l'espace disque est la commande *df* dont un extrait de la page de manuel est donné ci-dessous.

DF(1)	User Commands	DF(1)
NAME		
	df - report file system disk space usage	
SYNOPSIS		
	df [OPTION]... [FILE]...	
DESCRIPTION		
	This manual page documents the GNU version of df. df displays the amount of disk space available on the file system containing each file name argument. If no file name is given, the space available on all currently mounted file systems is shown.	
OPTIONS		
	Show information about the file system on which each FILE resides, or all file systems by default.	
	Mandatory arguments to long options are mandatory for short options too.	
	-B, --block-size=SIZE use SIZE-byte blocks	
	-h, --human-readable print sizes in human readable format (e.g., 1K 234M 2G)	
	-i, --inodes list inode information instead of block usage	
AUTHOR		
	Written by Torbjorn Granlund, David MacKenzie, and Paul Eggert.	
SEE ALSO		
	The full documentation for df is maintained as a Texinfo manual.	
GNU coreutils 7.4	October 2009	DF(1)

Nous allons construire pas à pas la commande permettant de créer le fichier *dispo.txt*. Dans toutes les questions suivantes, vous devrez donner une commande sur une ligne sans concaténation (pas de « ; », par contre les tubes et les redirections sont autorisés... voire recommandés).

1. Donnez la commande permettant d'afficher de manière lisible un descriptif de l'utilisation de l'espace disque du système.
2. La commande donnée en 1, affiche les informations de la manière suivante :



```
Sys. de fich.      Tail. Occ. Disp. %Occ. Monté sur
/dev/sda2         28G  10G  17G  38% /
udev              1,6G 264K 1,6G  1% /dev
none              1,6G  0  1,6G  0% /lib/init/rw
/dev/sda4         403G 42G  341G  11% /home
```

Donnez la commande (sur une ligne) permettant de n'afficher que les informations relatives au répertoire `/home` (vous devez compléter la commande donnée en 1).

3. Donnez la commande permettant de n'afficher que la taille de disque disponible (sans le G) pour le répertoire `/home` (vous devez compléter la commande donnée en 2).
4. Enfin, donnez la commande complète permettant de sauvegarder l'information obtenue en 3 dans le fichier `dispo.txt` (vous devez compléter la commande donnée en 3).

Exercice 2 – Chemin absolu (5 pts)

Réalisez le programme Python permettant d'obtenir la sortie suivante (demander à l'utilisateur de saisir un chemin absolu et renvoyer la liste des répertoires qui le composent) :

```
Donnez le chemin absolu : home
Erreur : vous n'avez pas donné un chemin absolu
Donnez le chemin absolu : /home/usr/bin
Liste des répertoires contenus dans ce chemin :
home, usr, bin
```

Votre code devra obligatoirement être structuré à l'aide d'une (ou plusieurs) fonction(s).

Exercice 3 – Conversion de dates (5 pts)

On considère des dates au format « long » `<jour> <numéro_jour> <mois> <année>` (exemple : mardi 5 janvier 2010). On veut écrire un programme Python demandant à l'utilisateur de saisir une date à ce format et qui renvoie la date au format « court » `jj/mm/aaaa` (pour l'exemple précédent on attend 05/01/2010).

1. Écrire une fonction `conversionMois` qui prend en paramètre un nom de mois et renvoie le numéro qui lui est associé.
2. Écrire les fonctions `donneJour`, `donneMois` et `donneAnnee` qui renvoient respectivement le numéro du jour, le mois et l'année d'une date donnée au format long.
3. En appelant les fonctions précédentes, donnez le code du programme final demandant une date au format long à l'utilisateur et affichant la date au format court.

Exercice 4 – Palindromes (5 pts)

Un palindrome est une phrase qui peut être lue à l'envers en conservant le même sens (on fait abstraction de la position des espaces). Par exemple, la phrase « esope reste ici et se repose » est un palindrome. Vous allez écrire un détecteur de palindromes.

1. Écrire une fonction qui étant donnée une phrase passée en paramètre renvoie cette phrase en ayant supprimé tous les espaces.
2. Écrire une fonction qui renvoie l'inverse de la phrase qui lui est passée en paramètre
3. Écrire une fonction `palindrome` qui renvoie la valeur booléenne « vrai » si la phrase passée en paramètre est un palindrome.
4. Donnez le code du programme final demandant à un utilisateur de saisir une phrase et indiquant s'il s'agit ou non d'un palindrome.