En tant qu'utilisateurs d'ordinateurs vous êtes habitués à manipuler des interfaces. Une interface permet à un utilisateur de donner des ordres à une machine. La plupart des interfaces que vous avez l'habitude d'utiliser sont des interfaces graphiques (où il faut cliquer sur des boutons). De telles interfaces ont l'avantage d'être simple d'utilisations mais l'inconvénient de n'être pas très expressive. Aussi si l'interface vous propose un bouton faisant exactement pour ce que vous recherchez c'est parfait sinon, vous êtes bloqués.

Pour cette raison, nous nous intéressons dans ce premier TP au terminal et au langage de programmation bash que nous utiliserons pour interfacer avec la machine tout au long de l'année.

Exercice 1 (Le terminal) :

1. Ouvrir un terminal à l'aide du raccourci `CTRL+ALT+T` ou via le menu graphique (l'émulateur de terminal installé sur les clés USB s'appelle **Konsole**).

Dans ce terminal, vous voyez apparaître une invite de commande ressemblant à cela :

```
janson@mp2i-janson:~$ _
```

La première partie (`janson`) indique votre nom d'utilisateur. La seconde partie (`mp2i-janson`) indique le nom de la machine à laquelle vous êtes connectés. La troisième partie (∼) indique le répertoire dans lequel vous vous trouvez actuellement dans l'arborescence de fichiers de votre machine. Un chemin dans l'arborescence de fichier de votre machine est dénoté par la liste des dossiers à suivre depuis la racine pour arriver là où vous voulez aller, séparé par des `/`. Par exemple dans ma machine le chemin vers le dossier contenant vos TPs est le suivant : `/home/anthony/git/cpge/mp2i/TP/`. Cette remarque devrait vous posez problème puisque nous avons dit que ∼ correspondait aux positionnement du terminal dans la machine, ∼ est en fait une abréviation pour `/home/<nom_utilisateur>/`, en effet ce répertoire appelé *home de l'utilisateur* est le répertoire par défaut qui sera ouvert lorsque vous ouvrirez une nouvelle console.

2. Demander à votre machine de **lister** le contenu du dossier courant à l'aide de la commande `ls` (pour *list*).
3. Demander à votre machine de vous montrer l'adresse du dossier courant à l'aide de la commande `pwd` (pour *path to working directory*).

La fonction `ls` peut être utilisée sans argument (comme à la question 2), mais elle peut aussi être utilisée avec des arguments. Pour donner des arguments à une fonction en bash, il suffit d'écrire :

```
janson@mp2i-janson:~$ fonction argument1 argument2
```

C'est à dire de séparer la fonction et les arguments par des espaces.

La fonction `tree` permet de voir l'arborescence de fichiers de votre machine. Elle n'est pas installée par défaut, mais nous pouvons l'installer si nous disposons des droits administrateurs.

Pour exécuter une commande avec les droits administrateurs, nous pouvons utiliser l'instruction `sudo` (pour *super user do*). Dans ce cas, la console vous demandera d'entrer votre mot de passe (rien ne s'affiche quand vous tapez votre mot de passe, c'est normal : c'est pour éviter que votre voisin puisse voir le nombre de caractères de votre mot de passe).

4. Pour installer la fonction `tree`, entrer la commande suivante, puis votre mot de passe :

```
janson@mp2i-janson:~$ sudo apt install tree
```

5. Une fois installée, tester la commande `tree`.

La fonction `mkdir` (pour *make directory*) permet de créer un nouveau dossier. Elle prend en argument la **position relative** du nouveau dossier.

6. Créer un nouveau dossier `MP2I` à l'aide de la commande "`mkdir MP2I`".
7. Observer la nouvelle arborescence de fichiers à l'aide de `tree`.

Il est possible de changer de dossier à l'aide de la commande `cd` (pour *change directory*), qui prend en argument le chemin relatif auquel on veut se déplacer.

8. Déplacez-vous dans le dossier `MP2I`.
9. Créer un dossier `TP_info` dans le dossier `MP2I`, puis un dossier `TP0` dans le dossier `TP_info` ; déplacez-vous dans le dossier `TP0`.

Quelques astuces pour utiliser la console

Voici quelques actions bien pratiques quand on utilise la console :

- on peut récupérer la dernière commande utilisée en appuyant sur la touche "flèche vers le haut" ;
- ensuite, on peut naviguer dans l'historique de nos commandes avec les touches "flèche vers le haut" et "flèche vers le bas" ;
- une fois le début d'une commande tapée, on peut l'autocompléter à l'aide de la touche `TAB` (à gauche de la touche `A` sur votre clavier) ; dans ce cas :
  - s'il n'y a qu'une seule commande qui commence par ce que vous avez tapé, le reste de la commande est rajouté ;
  - s'il y a plusieurs commandes qui commencent par ce que vous avez tapé, rappuyer sur `TAB` affiche toutes les commandes possibles ;
- on peut interrompre une commande avec `CTRL+C`.

Exercice 2 (Lancement de programmes) :
La console est également capable de lancer des programmes graphiques :

1. Entrer la commande `firefox`.
2. Tout en gardant la fenêtre de Firefox ouverte, retourner dans la console. Que constatez-vous ?

La console est bloquée, car elle attend que le programme (`firefox`) se termine.

3. Interrompre le programme dans la console avec `CTRL+C`.
4. Entrer la commande `firefox & disown`. Que constatez-vous ?

Exercice 3 (Chemins relatifs et absolus) :
Vous devriez vous trouver dans le dossier `~/MP2I/TP_info/TP0/`.

1. Vérifier cela à l'aide de la commande `pwd`.
2. Entrer la commande "`cd ..`". Où vous trouvez-vous ?

Il est donc possible de remonter dans l'arborescence de fichiers au moyen de `..` qui représente le répertoire parent.

Nous avons vu qu'un fichier `file` est représenté par un chemin de la racine de votre système de fichiers jusqu'à ce fichier. Cela peut devenir très gros à mesure que vous rajoutez des dossiers dans des dossiers. Heureusement plutôt que d'utiliser une telle description **absolue** du fichier il est possible de donner une description **relative** de sa position. Une description relative d'une position A dans l'arborescence de fichiers par rapport à une autre position B est un chemin de B vers A.

Dans le reste de cette exercice nous considérons l'exemple de l'arborescence de fichier de la figure 1.

3. Pour chacun des fichiers et répertoires de la figure 1, donner le chemin absolu vers ce fichier (ou répertoire) ainsi que le chemin relatif vers ce fichier depuis la position `/foo/b/`.

Exercice 4 (Édition de fichiers) : Nous allons dans la suite utiliser l'éditeur de texte VSCode, dont l'exécutable s'appelle `code`. Par curiosité, vérifions où se situe le programme `code` :

```
janson@mp2i-janson:~$ which code
```

Cette commande affiche le chemin vers le programme exécutable `code`. En principe, nous devrions taper tout ce chemin pour lancer `code`, mais le répertoire dans lequel est contenu `code`[1] est en fait automatiquement parcouru par le shell chaque fois qu'une commande est entrée par l'utilisateur. Ainsi, il n'est pas utile d'entrer systématiquement l'intégralité de ce chemin chaque fois qu'on veut lancer VSCode : tapez simplement la commande `code` pour lancer l'éditeur de texte VSCode.

1. Créer et sauvegarder un fichier dans ce même répertoire comprenant plusieurs lignes avec, par exemple, la liste de vos musiques préférées.

1. ainsi qu'un certain nombre d'autres répertoires

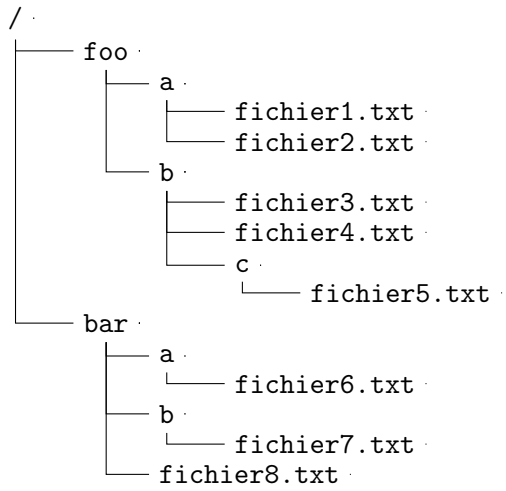

FIGURE 1 – Exemple d'arborescence de fichiers

À partir de maintenant, toutes les commandes ne seront plus décrites en détail à chaque fois. Afin de vous aider, vous disposez d'un outil très utile pour (re)trouver une commande et/ou la détail de ses options : la commande `man` (pour *manual*), qui permet de donner la documentation d'une commande. Par exemple si vous tapez `man cd` vous aurez des informations sur `cd`. Si vous ne savez pas le nom de la commande que vous cherchez, l'option `-k` vous permet de chercher toutes les pages du manuel qui contiennent une certaine chaîne de caractères (par exemple `man -k <mot>` vous permet de chercher toute les pages qui contiennent `<mot>`) [2]. Dans une page de manuel, vous pouvez faire une recherche en tapant le caractère "`/`" suivi de votre recherche, puis en appuyant sur la touche `ENTER`. Vous vous déplacez ensuite de résultat en résultat vers l'avant avec la touche `N` (pour *next*), et vers l'arrière avec `SHIFT-N`. On quitte le manuel en appuyant sur la touche `q` (pour *quit*).

La commande `man` bénéficie elle-même d'une page de manuel, accessible avec la commande `man man`.

2. Chercher et utiliser la commande adéquate pour afficher le nombre de caractères présents dans le fichier que vous venez de créer.
3. Trouver dans votre fichier les lignes qui contiennent la lettre "`e`" à l'aide de la commande `grep`.

Exercice 5 (Manipulation de fichiers) :

1. Faites une copie de votre fichier à l'aide de la commande `cp` (pour *copy*).
2. Créer un dossier `Test`, et déplacer la copie de votre fichier dans le dossier `Test` à l'aide de la commande `mv` (pour *move*).
3. Déplacez-vous dans le dossier `Test`, et renommer le fichier en `toto.txt`.
4. Supprimer le fichier `toto.txt` à l'aide de la commande `rm` (pour *remove*).
5. Revenir au dossier `TP0`, et supprimer le dossier `Test` (on vérifiera à l'aide de `ls` que le dossier a bien été supprimé).

`commande > fichier` copie la sortie de la commande `commande` dans le fichier `fichier`. Par exemple `ls -l > files.txt` va enregistrer dans `files.txt` la liste des fichiers présents dans le répertoire courant. `commande1 | commande2` donne en paramètre de `commande2` la sortie de la commande `commande1`. Par exemple, `echo ‘‘bonjour tout le monde’’ | wc -w` compte le nombre de mots contenus dans la chaîne "bonjour tout le monde".

2. À défaut, un moteur de recherche et internet constituent évidemment une solution très puissante

Soit le fichier *real.csv* contenant le texte suivant :

real.csv

```
Nom ; Prenom ; Naissance
Hitchcock ; Alfred ; 1899
Welles ; Orson ; 1915
Kubrick ; Stanley ; 1928
Fellini ; Federico ; 1920
Godard ; Jean-Luc ; 1930
Coppola ; Francis ; 1939
Renoir ; Jean ; 1899
Bergman ; Ingmar ; 1918
Ford ; John ; 1894
```

6. Donner la commande qui permet d'obtenir le fichier *liste.csv* qui est une copie de *real.csv* sans la première ligne (utiliser `tail` et `>`).
7. Donner la commande pour obtenir un affichage de la liste des réalisateurs par noms croissants (utiliser `cat`, `sort`, et `|`).
8. Sauvegarder le résultat de la dernière question dans un fichier `tri.csv`.

Exercice 6 (Compilation et scripts bash) :

1. Créer un fichier `hello.c`, et écrire un programme qui affiche "`Hello World!`" à l'écran.
2. Compiler votre programme avec la commande :

```
janson@mp2i-janson:~$ gcc -Wall hello.c -o executable
```

3. Afficher les fichiers à l'aide de `ls`.
4. Exécuter votre programme.