

TP2: Entrées/sorties

Lire, écrire dans un fichier : fopen, fclose,...

a. Écrire un programme `lire` qui prend comme arguments le nom d'un fichier texte et l'affiche ligne par ligne. On fera attention à la gestion des erreurs (le fichier n'existe pas, n'est pas lisible,...).

b. Modifier le programme pour qu'il indique à la fin de la lecture combien de mots, de caractères et de lignes a-t-il lu.

On se propose d'appliquer ça à la gestion d'un annuaire :

c. Écrire un programme `ajouter` qui prend comme arguments le nom d'un fichier "annuaire" ainsi que le nom, que le nom, prénom, et l'e-mail d'une personne et qui génère un autre fichier annuaire.

d. Modifier le programme `lire` qui devra prendre deux arguments : le nom de fichier "annuaire" et le nom d'une personne et qui affiche son adresse email si elle est dans l'annuaire et une erreur si elle n'y est pas.

Exemple :

```
touch annuaire.txt
./lire annuaire.txt
./ajouter annuaire.txt Dupont Martin dupont@imag.fr
./ajouter annuaire.txt Dupond Jean jean@rien.fr
cat annuaire.txt
Dupont Martin dupont@imag.fr
Dupond Jean jean@rien.fr
./lire annuaire.txt Dupond
jean@rien.fr
```

Interpréteur de commande

Le but des TDs suivant est d'arriver à programmer un mini-shell. L'idée est d'implémenter l'algorithme suivant :

```
tantque ( pas_fini )
    afficher une invite
    lire la commande tapée
    exécuter la commande
```

On supposera dans un premier temps que les commandes ne comportent pas de `&` ; ou autres `>`. On se limitera à des commandes de moins de 255 caractères. On fera attention aux différentes erreurs qui peuvent se produire.

a. Écrire un programme qui affiche une invite de commande (ex : `$`), attend que l'utilisateur ai tapé sa commande puis écrit "vous voulez executer la commande x" jusqu'à ce que l'utilisateur entre la commande "quitter".

Certaines commandes peuvent avoir des arguments, séparés par un ou plusieurs espaces. Par exemple `cp -r dossier1 dossier2` se décompose en {cp, -r, dossier1, dossier2, NULL}.

b. Écrire une fonction `char** arguments_commande(char * ma_commande)` qui rend un tableau contenant les arguments correspondants. Attention aux espaces multiples.

Généralement, exécuter une commande revient à lancer un programme, néanmoins dans certains cas particuliers, on exécute une commande directement dans le shell.

c. On verra plus tard le lancement d'un programme. Implémenter les fonctions `cd` et `echo`.

d. (bonus) Modifier l'invite pour faire apparaître le dossier courant.