TP4 – MANIPULATION DE FICHIERS

Le but des questions suivantes est de réaliser le programme **compte** qui prend une liste de fichiers en paramètre, et affiche le nombre de caractères, de lignes et de mots de chaque fichier, ainsi que le total pour tous les fichiers. La syntaxe d'appel est donc : **compte [nom_fic]+**

Soit la fonction **int compte(char fic[], int *ligne)** qui prend un nom de fichier en entrée et qui retourne dans la variable ligne le nombre de ligne que contient le fichier. En cas d'erreur cette fonction retourne 0, sinon 1.

Q1. Écrire la fonction **main()** qui prend en entrée une liste de noms de fichiers (retourne une erreur sur la sortie standard d'erreur s'il n'y a aucun fichier). La fonction **compte()** est appelée pour chaque fichier passé en argument et, s'il n'y a pas eu d'erreur lors de l'appel, affiche sur la sortie standard le nom du fichier suivi de son nombre de lignes, sinon affiche sur la sortie erreur le nom du fichier en indiquant que c'est une erreur. Avant de terminer la fonction main affiche le nombre total de ligne pour tous les fichiers.

Par exemple :

```
% compte f1 f2 f3
LIGNES      FICHIER
 32         f1
120         f2
 44         f3
196         total
```

Q2. En vous aidant **exclusivement** des fonctions en annexe, écrivez la fonction **int compte(char fic[], int *ligne).**

Q3. Réécrivez la fonction **compte** pour qu'elle compte également le nombre de caractères et le nombre de mots :

int compte(char fic[], int *ligne, int *car, int *mot).

Q4. Réécrivez la fonction **main()** pour que ce programme accepte maintenant des arguments avant la liste des fichiers. Un argument est toujours précédé du signe moins "-". Les arguments sont : "l" pour compter les lignes uniquement, "c" pour compter les caractères uniquement, "m" pour compter les mots uniquement, n'importe quel combinaison de l c et m (dans n'importe quel ordre) ou aucun argument (dans ce cas le programme agit comme si l'on avait précisé l c et m). La syntaxe devient donc :

compte [-arguments]* [nom_fic]+

Par exemple :

```
% compte -lm f1 f2 f3
LIGNES      MOTS        FICHIER
 32          246        f1
120         1024        f2
 44          856        f3
196         2126        total
```

Q5. Serait-il intéressant d'optimiser la fonction **compte()** pour qu'elle ne recherche pas systématiquement les lignes, les caractères et les mots, lorsque l'on demande qu'une ou deux informations (par exemple que les mots et les lignes, ou bien que les caractères). Justifiez votre réponse.

ANNEXE

- Séparateur de ligne: `'\n'`
- Séparateur de mot : `' '` (espace) ou `'\t'` (tabulation).

- Ouverture d'un fichier :

```
FILE *fopen(char *fic, char *type)
```

`fic` est le nom de fichier à ouvrir,

`type` est une combinaison des caractères **r** (read : lecture), **w** (write : écriture), et **a** (append : écriture en fin de fichier).

Retourne un pointeur sur la structure du fichier ouvert, ou un pointeur nul (0) en cas d'erreur.

- Fermeture d'un fichier :

```
int fclose(FILE *f)
```

ferme le fichier ouvert pointé par f.
Retourne 0 en cas d'erreur, 1 sinon.

- Lecture d'un fichier :

```
int fread(char *ptr, int taille, int nitem, FILE *f)
```

lit le fichier ouvert pointé par f dans l'espace mémoire (pré-alloué) pointé par `ptr` un nombre d'octet égal à `nitem * taille`.
Retourne 0 en cas d'erreur ou de fin de fichier, le nombre d'item lu sinon.