TP6
Programmation C L2.1
Comptage des mots dans un fichier

Dans cette séance de travaux pratiques, nous allons coder un programme permettant de déterminer le nombre de mots, de caractères et de lignes d'un ou plusieurs fichiers textes. Concrètement, nous allons réaliser une implémentation personnalisée du programme `wc`, qui, malgré les apparences, n'est pas un programme sanitaire.

---

1. **Traitement des arguments d'un programme**
   Écrivez un programme qui :
   - Affiche le nombre d'arguments qui lui a été fourni.
   - Affiche si le premier argument commence par le caractère `-`.
   - Pour chaque argument ne commençant pas par le caractère `-` affiche si le contenu d'un fichier portant ce nom peut-être lu.

2. **Compter le nombre de caractères**
   Écrivez une fonction `int CompteCaracteres(FILE* fichier)` qui renvoie le nombre de caractères présents dans `fichier`. Modifier le programme précédent pour tester votre fonction.

3. **Compter le nombre de lignes**
   Écrivez une fonction `int CompteLignes(FILE* fichier)` qui renvoie le nombre de lignes présentes dans `fichier`. Modifier le programme précédent pour tester votre fonction. On rappelle qu'un saut de ligne est codé par le caractère `'\n'`.

4. **Compter le nombre de mots**
   Écrivez une fonction `int CompteMots(FILE* fichier)` qui renvoie le nombre de mots présents dans `fichier`. Modifier le programme précédent pour tester votre fonction. On rappelle qu'un mot est une suite de caractères ne contenant pas de séparateur (espace `' '`,tabulation, `'\t'`, passage à la ligne `'\n'`. Faites attention : deux mots peuvent être séparés par plus d'un séparateur.

5. **Tout compter d'un coup**
   Le défaut des fonctions précédentes est qu'elles parcourent toutes les trois entièrement le fichier. On souhaite créer un programme donnant ces trois valeurs en ne parcourant qu'une seule fois le fichier. En vous inspirant des fonctions précédentes, écrivez une fonction `CompteTout(FILE* fichier,int* caracteres, int* lignes, int* mots)` qui place dans `*caracteres` le nombre de caractères présents dans `fichier`, dans `*lignes` le nombre de lignes et dans `mots` le nombre de mots.

6. **Réaliser un programme complet**
   Vous allez utiliser les fonctions précédemment codées pour réaliser une implémentation personnalisée de la fonction `wc` (word count). Vous créerez donc un programme `my_wc` dont le comportement est le suivant :

   SYNOPSIS : `wc [-mlvw] [fichier1] [fichier2] ···`

   DESCRIPTION : Affiche le nombre de lignes, mots, et caractères présents dans fichier1, fichier2, etc... Dans le cas où aucun fichier n'est fourni, la fonction effectuera son opération sur l'entrée standard (rappel : sur l'entrée standard, la fin de fichier est indiquée par la combinaison `CTRL+d`).

OPTIONS :

- `-m` : affiche uniquement le nombre de caractères.
- `-l` : affiche uniquement le nombre de lignes.
- `-w` : affiche uniquement le nombre de mots.
- `-v` : affiche un texte annonçant que ce programme est votre version de la fonction `wc`.

Bien entendu, il est possible d'utiliser plusieurs options en même temps (`-mlv`). Dans le cas où aucune des options `m,l,w` n'est fournie, on considère qu'elles sont présentes toutes les trois. Si deux d'entre elles sont présentes, on affiche seulement les deux informations correspondantes.

Faites bien attention : les options sont fournies comme le premier argument du programme, et si elles sont présentes ailleurs, votre programme les traitera comme un fichier de ce nom. Si un fichier fourni en argument n'existe pas, la fonction affichera un message d'erreur.

Si une option invalide est fournie, vous afficherez un message d'erreur ainsi qu'une explication de l'utilisation de la fonction et des options disponibles. Pour cela, écrire une fonction `void usage(char *nom)` où `nom` est le nom de votre programme et qui affiche le synopsis.