TD 9 de langage C

Les fichiers

Exercice 1 :

A l'aide d'un éditeur de textes, créer un fichier NOMBRES.TXT qui contient une liste de nombres entiers. Dans le fichier, chaque nombre doit être suivi par un retour à la ligne. Ecrire un programme qui affiche les nombres du fichier, leur somme et leur moyenne.

Exercice 2 :

Ecrire un programme qui crée un fichier texte nommé "tableX.txt" contenant la table de multiplication pour le nombre X (le fichier "table7.txt" contiendra la table de 7), présentée sous la forme suivante :

1 x 7 = 7
2 x 7 = 14
etc...
9 x 7 = 63

Exercice 3 :

Ecrire un programme qui crée le fichier MOTS.TXT contenant une série de 50 mots au maximum (longueur maximale d'un mot : 50 caractères). La saisie des mots se terminera à l'introduction du symbole '*****' qui ne sera pas écrit dans le fichier.

Exercice 4 :

Ecrire un programme qui affiche le nombre de mots, le nombre de palindromes ainsi que la longueur moyenne des mots contenus dans le fichier MOTS.TXT. Définir une fonction dont le prototype est ***int palin(char *)*** qui fournit le résultat 1 si la chaîne transmise comme paramètre est un palindrome, sinon la valeur zéro.

Exercice 5 :

Ecrire un programme qui charge les mots du fichier MOTS.TXT dans la mémoire centrale, les trie (tri à bulle par exemple) et les écrit dans un deuxième fichier MOTS_TRI.TXT. Les mots seront mémorisés à l'aide d'un tableau de pointeurs sur caractère et la mémoire nécessaire sera réservée de façon dynamique.

Exercice 6 :

Créer puis afficher à l'écran le fichier INFORM.TXT dont les informations sont structurées de la manière suivante :

Numéro de matricule (entier)
Nom (chaîne de caractères)
Prénom (chaîne de caractères)

Le nombre d'enregistrements à créer est à entrer au clavier par l'utilisateur.

Exercice 7 :

Ecrire un programme qui crée un fichier INFBIS.TXT qui est la copie exacte (enregistrement par enregistrement) du fichier INFORM.TXT. Il faut gérer l'erreur d'ouverture du fichier lu et faire en sorte qu'on ne puisse pas écrire dans un fichier existant.

Exercice 8 :

Ajouter un nouvel enregistrement (entré au clavier) à la fin de INFORM.TXT et sauver le nouveau fichier sous le nom INFBIS.TXT.

Exercice 9 :

Insérer un nouvel enregistrement dans INFORM.TXT en supposant que le fichier est trié relativement à la rubrique NOM et sauver le nouveau fichier sous le nom INFBIS.TXT.

Exercice 10 :

Supprimer dans INFORM.TXT tous les enregistrements suivants (sauver le nouveau fichier sous le nom INFBIS.TXT) :

a) dont le numéro de matricule se termine par 8
b) dont le prénom est "Paul" (utiliser strcmp)
c) dont le nom est un palindrome. Utiliser la fonction ***palin()*** définie précédemment.

Exercice 11 :

Ecrire un programme qui détermine dans un fichier texte dont le nom est entré au clavier :

- le nombre de caractères qu'il contient,
- le nombre de chacune des lettres de l'alphabet (sans distinguer les majuscules et les minuscules),
- le nombre de mots,
- le nombre de paragraphes (c.-à-d. de retours à la ligne),

Les retours à la ligne ne devront pas être comptabilisés dans les caractères.

On admettra que deux mots sont toujours séparés par un ou plusieurs des caractères suivants :

- fin de ligne
- espace
- ponctuation : **. : , ; ? !**
- parenthèses : **( )**
- guillemets : **"**
- apostrophe : **'**

Utiliser une fonction d'aide dont le prototype est ***int sepa(char)*** qui décide si un caractère transmis comme paramètre est l'un des séparateurs mentionnés ci-dessus. ***sepa()*** restituera la valeur 1 si le caractère est un séparateur et 0 dans le cas contraire. ***sepa()*** utilise un tableau qui contient les séparateurs à détecter.

Exemple :

```
Nom du fichier texte : LITTERA.TXT
Votre fichier contient:
  12 paragraphes
  571 mots
  4186 caractères
dont
  279 fois la lettre a
  56 fois la lettre b
  . . .
  3 fois la lettre z
et 470 autres caractères
```

Exercice 12 :

Ecrire un programme qui vérifie la validité d'une série de numéros de CCP mémorisés dans un fichier. Un numéro de CCP est composé de trois parties : un numéro de compte, un séparateur '-' et un numéro de contrôle. Un numéro de CCP est correct :

- si le reste de la division entière de la valeur devant le séparateur '-' par 97 est différent de zéro et égal à la valeur de contrôle.
- si le reste de la division par 97 est zéro et la valeur de contrôle est 97.

Exemple :

Numéro de CCP 15742-28 :
15742 modulo 97 = 28
correct
Numéro de CCP 72270-5 :
72270 modulo 97 = 5
correct
Numéro de CCP 22610-10 :
22610 modulo 97 = 9
incorrect
Numéro de CCP 50537-0 :
50537 modulo 97 = 0
nombre incorrect, car la valeur de contrôle devrait être 97.
Numéro de CCP 50537-97 :
50537 modulo 97 = 0
correct

Utiliser une fonction CCP_TEST qui a comme paramètres les deux parties numériques d'un nombre de CCP et qui affiche alors un message indiquant si le numéro de CCP est valide ou non.
Pour tester le programme, créer à l'aide d'un éditeur de texte un fichier CCP.TXT qui contient les numéros ci-dessus, suivis par des retours à la ligne.

Exercice 13 :

Ecrire un programme qui remplace, dans un fichier contenant un texte, les retours à la ligne par des espaces. Si plusieurs retours à la ligne se suivent, seul le premier sera remplacé. Les noms des fichiers source et destination sont entrés au clavier.

Exercice 14 :

Ecrire un programme qui affiche le contenu d'un fichier texte sur un écran de 25 lignes et 80 colonnes en attendant la confirmation de l'utilisateur (par 'Enter') après chaque page d'écran. Utiliser la fonction ***getchar()***.