Série 4 :
Programmation *C* - Entrées / Sorties

Buts

En attendant d'en savoir plus sur la compilation séparée, le but de cette série d'exercices est de vous permettre de pratiquer les entrées/sorties vues la semaine passée.

Rappel

Avez-vous pris connaissance des **conseils relatifs à ces séries d'exercices** ?

Exercice 1 : écriture dans un fichier (niveau 1)

Écrivez le programme `ecriture.c` qui :

- lit depuis le clavier les **noms** et **âges** de différents individus;
- sauvegarde ces données dans un fichier nommé `data.dat`.

Votre programme devra en outre :

- lire des valeurs tant que l'utilisateur n'indique pas qu'il a terminé la saisie, en appuyant sur les touches `CTRL+D` (ce qui correspond au caractère signalant la fin de fichier);
- vérifier que l'ouverture du fichier s'est correctement réalisée, et afficher un message d'erreur dans le cas contraire.

Indications:

- pensez à vérifier que vos opérations d'extraction (entrée) se déroulent bien ;
- n'oubliez pas de fermer le fichier à la fin des opérations d'écriture;
- et finalement, regardez le contenu du fichier produit.

Testez votre programme avec les entrées suivantes :

```
Jo      24
Marc    35
Ted     74
Andy     3
Werner  48
OldBob 103
```

Exemple d'execution :

```
Entrez un nom (CTRL+D pour terminer) : Jo
âge : 24
Entrez un nom (CTRL+D pour terminer) : Marc
âge : Ted
Je vous demande un age (nombre entier positif) pas n'importe quoi !
Cet enregistrement est annulé.
Entrez un nom (CTRL+D pour terminer) : Marc
âge : 35
Entrez un nom (CTRL+D pour terminer) : ^D
```

Exercice 2 : lecture depuis un fichier (niveau 1)

Dans le fichier `lecture.c`:

1. Affichez à l'écran le contenu du fichier créé lors de l'exercice précédant, et affichez de plus le nombre de personnes contenues dans ce fichier, ainsi que la moyenne et les extrêmes des âges.
   Vérifiez que l'ouverture du fichier s'est bien réalisée, et affichez un message d'erreur dans le cas contraire.
2. Modifiez ensuite votre programme de sorte qu'il réalise l'affichage en s'approchant le plus possible du format suivant:
   - affichage du nom sur 15 caractères, alignement à gauche;
   - affichage de l'âge sur 3 caractères, alignement à droite ;
   - affichage du total des entrées sur 2 caractères ;

- affichage de la moyenne sur 3 caractères au maximum ;
- affichage des ages minimum et maximum comme les autres âges.

Exemple :

```
+-----------------+-----+
| Jo              |  24 |
| Marc            |  35 |
| Ted             |  74 |
| Andy            |   3 |
| Werner          |  48 |
| Bob             | 103 |
+-----------------+-----+
  âge minimum     :   3
  âge maximum     : 103
 6 personnes, âge moyen : 47.8 ans
```

[optionnel] Exercice 3 : fichiers binaires (niveau 1)

Dans le **zip à télécharger ici** [ou sinon **lien direct ici**] se trouve un fichier binaire `a_lire.bin` contenant des entiers (nombre non précisé) .

1. Écrivez un programme qui lit le contenu de ce fichier et l'affiche en clair à l'écran.
2. Modifier ce programme pour qu'il «décode» le précédent contenu en affichant le **caractère** correspondant à la racine carrée du nombre lu.

   On fera ici particulièrement attention aux conversions de type, la fonction «racine carrée» (`sqrt` de `math.h`) prenant comme argument des `double`s et retournant un `double` (vous devriez obtenir un message intelligible en français).
3. Écrivez ensuite un programme permettant de constituer de tels fichiers, i.e. un programme lisant une phrase au clavier et créant un fichier binaire contenant le tableau d'entiers dont les valeurs sont les carrés des caractères de la phrase (sans le 0 final).
4. Testez votre programme avec l'exemple précédemment obtenu.
   Faites en particulier attention à l'aspect signé/non signé (e.g. avec les accents).

Exercice 4 : statistiques sur un fichier (niveau 2)

On cherche ici à écrire un programme `stat.c` qui calcule les statistiques sur les lettres contenues dans un fichier.

4.1 Ouverture du fichier

Remarque préliminaire : Les moins à l'aise d'entre vous peuvent laisser tomber cette première partie (et la fonction `demander_fichier`) et ouvrir le fichier comme ils le préfèrent, par exemple comme dans les exercices précedents. Revenez alors plus tard à cette partie quand vous vous sentirez plus à l'aide.

Écrire une fonction `demander_fichier`, qui retourne `FILE*` correspondant au fichier ouvert (ou `NULL` sino) :

```
FILE* demander_fichier();
```

Cette fonction, après avoir demandé à l'utilisateur d'entrer le nom du fichier à lire, ouvrira le fichier correspondant.

En cas d'erreur à l'ouverture, la fonction redemandera le nom du fichier, au maximum 3 fois. Au bout de 3 échecs le programme abandonera et retournera `NULL`.

Exemple d'exécution :

```
Nom du fichier à lire : stupid.name
 -> ERREUR, je ne peux pas lire le fichier "stupid.name"
Nom du fichier à lire : stupid2.name
 -> ERREUR, je ne peux pas lire le fichier "stupid2.name"
Nom du fichier à lire : evenmorestupid.name
 -> ERREUR, je ne peux pas lire le fichier "evenmorestupid.name"
=> j'abandonne !
```

Exemple positif :

```
Nom du fichier à lire : stupid.name
 -> ERREUR, je ne peux pas lire le fichier "stupid.name"
Nom du fichier à lire : data.dat
 -> OK, fichier "data.dat" ouvert pour lecture.
```

4.2 Récolte des statistiques

Définir le type `Statistique` comme un tableau d'entiers longs non-signés (choisissez de préférence une allocation dynamique).

Écrire une fonction `initialise_statistique`, prenant comme argument une variable `Statistique` (et peut être d'autres arguments si nécessaire) et initialisant à 0 tous ses éléments.

Écrire une fonction `collecte_statistique`, de prototype

```
unsigned long int collecte_statistique(Statistique a_remplir, FILE* fichier_a_lire);
```

qui collectera le nombre de fois que chaque caractère compris entre l'espace (`' '` ou `(char) 32`) et `'ý'` (`(char) 253`) apparait dans le fichier `fichier_a_lire`.

Ainsi `a_remplir[0]` contiendra le nombre d'espaces contenus dans le fichier `fichier_a_lire`, et `a_remplir[221]` (253-32) le nombre de `'ý'` que contient `fichier_a_lire`.

Pour lire un fichier caractère par caractère, utilisez la fonction `get(char)`. `man getc` pour plus de détails.

La fonction `collecte_statistique` retournera le nombre total de caractères enregistrés dans `a_remplir`, c'est-à-dire la somme de ses éléments.

4.3 Affichage des statistiques

Écrire pour finir une fonction `affiche` qui prend une `Statistique` en argument (plus d'autres arguments si nécessaire) et affiche les statistique récoltées en valeurs absolues (les vrais nombres) et relatives (pourcentages).

Les valeurs absolues seront affichés alignées à droite sur 11 caractères et les pourcentages alignés à droite sur 5 caractères.

Attention ! Il ne faut pas afficher les caractères qui ne sont pas apparus dans le fichier (c.-à-d. ayant un compte de 0).

Exemple d'affichage :

```
STATISTIQUES :
  :           6 -  12.2%
0 :           1 -  2.04%
1 :           1 -  2.04%
2 :           1 -  2.04%
3 :           3 -  6.12%
4 :           3 -  6.12%
...
```

Dernière mise à jour le 10 mars 2016
Last modified: Thu Mar 10, 2016