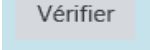



## I2181A : Langage C - Modularisation (TP2)

Avant de résoudre les exercices de cette fiche, faites le petit test CodeRunner « **S2 Types utilisateur - Quiz** » sur Moodle.

Pour chaque question du quizz, il faut d'abord cliquer sur le bouton  avant de cliquer sur « *Page suivante* » afin de vérifier que votre solution compile et passe les différents tests.

Conseil : afin d'avoir une meilleure visibilité des questions, cachez le volet gauche de Moodle grâce à l'icône (en haut à gauche de la fenêtre) : 

Il est également possible d'agrandir la cadre contenant le code à compléter en tirant sur le coin inférieur droit :



### 1. Les types utilisateurs

L'objectif de cet exercice est de concevoir un module `biblio` pour gérer une bibliothèque.

Pour ce faire, définissez une **structure Livre** qui contient :

1. une chaîne de 128 caractères pour le titre
2. une chaîne de 80 caractères pour le(s) auteur(s)
3. un entier long pour l'ISBN (*International Standard Book Number*)
4. une chaîne de 50 caractères pour l'éditeur
5. un entier pour l'année d'édition
6. un type énuméré pour le *genre*, sachant que les genres à prendre en considération pour cette application sont : « Bande dessinée », « Poésie », « Théâtre », « Roman », « Roman historique », « Littérature française », « Littérature étrangère », « Sciences », « Informatique », « Science-fiction », « Santé » et « Histoire ».

Le **module biblio** fournit plusieurs fonctions :

- `lireLivre` qui renvoie vrai si un livre a pu être lu à l'entrée standard (à raison d'une information par ligne, une ligne vide terminant un livre) et faux sinon ; cette fonction permet de charger un nouveau livre à partir de l'entrée standard. Nous vous conseillons d'utiliser la

fonction `readLimitedLine()` du module `utils` pour lire les chaînes de caractères et la fonction `scanf()` pour lire les données numériques.

- `str2genre` qui renvoie le genre littéraire correspondant à une chaîne de caractères
- `genre2str` qui renvoie la chaîne de caractères correspondant à un genre littéraire
- `livre2str` qui convertit un livre en chaîne de caractères (utilisez la fonction `sprintf()`)
- `afficherBib` qui affiche le contenu d'une table de livres
- `ajouterLivre` qui ajoute un livre à une table de `Livre` (en gérant dynamiquement la taille physique de cette table)

À l'aide des fonctions du module `biblio`, le **programme principal** doit :

- lire à l'entrée standard les informations de plusieurs livres (à raison d'une information par ligne; une ligne vide sépare deux livres)
- placer les informations dans une table de structures `Livre`
- afficher le contenu de la bibliothèque

Nous vous demandons de :

- concevoir le fichier entête `biblio.h` contenant vos définitions de types et les prototypes de fonctions (il ne vous est pas demandé d'écrire les spécifications de vos fonctions) ; vous pouvez vérifier la syntaxe de votre header en le compilant : `cc biblio.h`
- écrire le fichier source `biblio.c` contenant l'implémentation des fonctions ; pour gagner du temps, utilisez le module `utils.h`
- écrire un *makefile* pour compiler votre application.

Vous pouvez tester votre programme en redirigeant les fichiers `bib.txt` ou `tintin.txt` vers l'entrée standard comme suit :

```
./testBilbio < tintin.txt.
```

## 2. Exercice Bonus 1

Reprenez le programme du TP1 (manipulations de tableaux dynamiques à l'aide de fonctions) en définissant un nouveau type de données : une structure contenant un tableau d'entiers, accompagné de ses tailles logique et physique.

### 3. Exercice Bonus 2

Reprenez votre solution de l'exercice 1.

Ajouter la fonction suivante à votre module `biblio` :

- `comparerLivre` qui compare l'année d'édition de deux livres (elle envoie 0 si les livres a et b ont la même année d'édition; une valeur positive si l'année d'édition du premier livre est postérieure à celle du second livre; une valeur négative si c'est le contraire).

Après avoir chargé et affiché une bibliothèque de livres, votre programme principal doit :

- trier la bibliothèque par année d'édition (utilisez les fonctions `qsort` de la bibliothèque standard C et `comparerLivre`)
- afficher la bibliothèque triée.

### 4. Exercice Bonus 3

Effectuez un refactoring de l'exercice 6 du TP6 (de l'UE I2011 Langage C bases au Q1) en introduisant un type `Image`. Cette structure contient une matrice de pixels, ses dimensions ainsi que sa profondeur de bits.