**Langage de programmation : bases**

**HAUTE ÉCOLE DE NAMUR-LIÈGE-LUXEMBOURG**

**DA-IA – bloc 1**

Exercices 2 – Fichiers

Objectifs

* Continuez à utiliser les éléments vus dans le cadre du cours de « Langage de programmation : bases »
* Écrire et utiliser des structures
* Apprendre à gérer des fichiers

# Introduction

Dans ce document, plusieurs conventions sont utilisées :

* les mots gras désignent des termes de vocabulaire liés à l’**informatique en général**.
* les mots soulignés et gras désignent des termes de vocabulaire directement liés aux cours de **programmation**.
* le logo signifie que vous avez quelque chose à réaliser.
* le logo est associé aux cadres présentant certaines conventions.
* le logo est associé aux cadres présentant les éléments liés à la propreté/lisibilité du code (*clean code*).

Pour ne pas perdre de temps, créez une seule solution avec plusieurs projets correspondant à chaque exercice.

Vous devez penser à ouvrir et à fermer le fichier de données dans chaque fonction employée.

# Exercice 1 : jeux à évaluer

Commencez par créer un nouveau projet (et solution) en lui donnant comme nom ListeJeux, par exemple. N’oubliez pas de faire en sorte que le fichier principale, celui qui contient la fonction main, ait l’extension .c.

L’objectif de cet exercice est de générer un fichier FiJeux.dat qui contient une liste de jeux à évaluer.

Chaque enregistrement concerne un jeu et contient les champs suivants :

son titre (maximum 100 caractères),

un entier représentant le nombre d’avis donnés et

un réel représentant la moyenne des notes attribuées sur 5.

Écrivez un programme qui génère ce fichier.

Pour ce faire, vous devez commencer par définir une structure qui correspond à un jeu/enregistrement comme décrit ci-dessus.

Dans la fonction principale, déclarez un enregistrement et initialisez les deux champs numériques de celui-ci à 0. Inutile de le faire pour chaque jeu, seul le nom du jeu change.

Ensuite, ouvrez le fichier en écriture afin d’y écrire les enregistrements.

Pour chaque jeu, effectuez les étapes suivantes :

demandez le titre du jeu à l’utilisateur (jusqu’à ce qu’il entre un nom vide) et le mémoriser dans l’enregistrement,

écrivez cet enregistrement dans le fichier.

Le titre est stocké en majuscules dans le fichier ; à vous de prévoir une fonction, qui transforme le titre entré par l’utilisateur sous la forme voulue.

Afin de tester si ce programme fonctionne, écrivez une fonction qui permettent, après avoir écrit les enregistrements dans le fichier, de les lire et d’affiche le contenu du fichier, enregistrement par enregistrement. Attention, l’ouverture du fichier doit se faire en lecture, cette fois-ci.

**Note :** aucun tableau **autre** que la chaine de caractères du titre n’est nécessaire dans cet exercice. Vous devez écrire les enregistrements directement dans le fichier.

# Exercice 2 : évaluation des jeux

Commencez par créer un nouveau projet (et solution) en lui donnant comme nom EvaluationJeux, par exemple. N’oubliez pas de faire en sorte que le fichier principale, celui qui contient la fonction main, ait l’extension .c.

Le but de cet exercice est de mettre à jour le fichier créé dans l’exercice précédent sur base des évaluations données par les joueurs.

Copiez le fichier FiJeux.dat, déjà créé et garni dans l’exercice précédent, dans le répertoire du nouveau projet afin de pourvoir le réutiliser.

Commencez par définir une structure qui correspond à un jeu/enregistrement comme décrit dans l’exercice précédent.

Dans la fonction principale, vous devez déclarer un enregistrement afin d’y mémoriser ce que vous allez lire dans le fichier lors de la recherche du jeu.

Ouvrez le fichier en lecture et écriture, mais en positionnant le pointeur en début de fichier. Le but est en effet de lire les enregistrements jusqu’à trouver le jeu en question, de modifier les données concernant son évaluation et ensuite de réécrire cet enregistrement au bon endroit.

Pour chaque jeu évalué, effectuez les étapes suivantes :

demandez le titre du jeu au joueur (jusqu’à ce qu’il entre un nom vide) et une note entière sur 5,

mettez le titre en majuscules,

cherchez le titre du jeu dans le fichier (il peut ne pas être dans le fichier),

* + si vous trouvez le titre du jeu dans le fichier, mettez les données liées à son évaluation à jour,
  + si non, affichez un message d’erreur.

Attention, le fichier contient le nombre d’avis et la note moyenne. Le calcul de la nouvelle moyenne n’est donc pas juste une somme… Par exemple, si la moyenne était de 3,6 pour un nombre d’avis égal à 14 et que la nouvelle note obtenue est de 3. Que vaut la nouvelle moyenne ? Réfléchissez !

Afin de vérifier que votre programme fonctionne correctement, vous devez afficher le contenu du fichier après chaque mise-à-jour. Vous pouvez copier/coller la fonction définie dans l’exercice 1 et y faire appel judicieusement.

**Note :** aucun tableau **autre** que la chaine de caractères du titre n’est nécessaire dans cet exercice. Vous devez lire et écrire les enregistrements directement dans le fichier.

# Exercice 3 : bibliothèque

Commencez par créer un nouveau projet (et solution) en lui donnant comme nom Bibliotheque, par exemple. N’oubliez pas de faire en sorte que le fichier principale, celui qui contient la fonction main, ait l’extension .c.

Recopiez le fichier source de l’exercice (sur la bibliothèque) réalisé dans la série d’exercices précédente, dans le répertoire du nouveau projet (et solution).

Modifiez-le en ajoutant les fonctions suivantes :

1. Définissez une fonction qui permet de mémoriser l’ensembles des livres dans un fichier en fin de programme. Il vous suffit de parcourir la liste chainée et d’écrire chaque livre dans un fichier.
2. Définissez une fonction qui permet de charger l’ensemble des livres mémorisés dans le fichier en début de programme. Il faut lire chaque livre dans le fichier et l’ajouter à la liste chainée. N’oubliez pas que vous avez déjà une fonction d’ajout ! Faites appel à ces fonctions de façon judicieuse…

N’hésitez pas à diviser le code en sous-fonctions et/ou à réutiliser des fonctions existantes pour réaliser ces deux nouvelles tâches.

Soyez attentifs à l’algorithmique, au point de modification unique et à l’utilisation des types énumérés lorsque ça se justifie.

# Exercice 4 [facultatif] : évaluation de jeux au complet !

Commencez par créer un nouveau projet (et solution) en lui donnant comme nom JeuxComplet, par exemple. N’oubliez pas de faire en sorte que le fichier principale, celui qui contient la fonction main, ait l’extension .c.

Créez un programme qui permet de répondre aux souhaits suivants en passant par un menu :

1. Ajouter un jeu en fin de fichier
2. Supprimer un jeu, c’est-à-dire mettre « \*\*\* » dans le champ titre

Suppression logique vs suppression physique

Au lieu d’effectuer une suppression physique à chaque demande de l’utilisateur, et donc devoir modifier tout le fichier à chaque suppression, il est parfois préférable de faire ce qu’on appelle la suppression logique.

En effet, dans le cas d’une suppression physique, on ne peut pas faire un « simple » décalage comme dans les tableaux. Il faut lire chaque enregistrement et le recopier, bien sûr en passant celui qui doit être supprimé, dans un nouveau fichier. Cette procédure est donc lourde en sachant que seul un enregistrement est supprimé… Un fichier peut en effet contenir des milliers d’enregistrements !

En procédant comme demandé ci-dessus, l’enregistrement est encore physiquement présent dans le fichier, mais un des champs est modifié pour signifier qu’il ne doit plus être considéré comme présent lors des autres traitements.

Ainsi, pour effectuer une suppression physique, on lance une procédure à part, toutes les x périodes, de façon à effacer réellement les enregistrements ainsi modifiés. Cela permet de supprimer physiquement tous les enregistrements effacés logiquement préalablement. Il ne faut donc parcourir le fichier et le recopier (sans les enregistrements effacés) qu’une fois toutes les x périodes et non à chaque suppression.

1. Modifier le titre d’un jeu
2. Ajouter une note à un jeu
3. Afficher tous les jeux (titres, nombres d’avis, moyenne)
4. Afficher un jeu sur base d’un titre obtenu de l’utilisateur
5. Afficher les jeux ayant une moyenne supérieure ou égale à celle obtenue de l’utilisateur
6. Quitter

N’hésitez pas à diviser le code en sous-fonctions et/ou à réutiliser des fonctions existantes pour réaliser ces deux nouvelles tâches. Pensez au **point de modification unique** !

Soyez attentifs à l’algorithmique, au point de modification unique et à l’utilisation des types énumérés lorsque ça se justifie.