Travaux Dirigés – TD 8

Exercice 50. Fichiers

1. Écrire une procédure/fonction permettant de stocker M vecteurs réels de taille N (M et N seront saisis par l'utilisateur) dans un fichier avec un nom donné par l'utilisateur.
2. Écrire une procédure/fonction qui permet de lire le contenu d'un fichier avec un nom donné par l'utilisateur. Afficher le contenu du fichier s'il existe.
3. Écrire une troisième procédure/fonction permettant de calculer la somme de deux vecteurs quelconques dans le fichier précédent et ajouter ce résultat de calcul dans ce fichier.

Exercice 51. Structure pointeurs et fichiers

Le responsable d'une petite bibliothèque souhaite informatiser la gestion du prêt des livres. Chaque livre est identifié par une côte de 8 caractères, un titre, le nom de l'auteur principal, le nom de l'éditeur, l'année de publication. A ces informations il faut ajouter l'indication du prêt (livre en prêt ou en rayon) et la date de prêt s'il y a lieu.

Voici une proposition pour la structure BibLivres :

```
typedef enum {prete, disponible} etatLivre;
typedef struct {
                char auteur[20] ;
                char titre[100] ;
                char cote[9] ;
                int annee ;
                char editeur[10] ;
                etatLivre EnPret ;
                date DateDePret ;
} BibLivres ;
```

1. Ecrire une procédure/fonction pour saisir les informations d'un livre.
2. Ecrire une procédure/fonction permettant de saisir N livres (N sera donné par l'utilisateur) et de les enregistrer dans un fichier dont le nom du fichier est saisi par l'utilisateur. La première ligne du fichier contient le nombre de livres stockés. Chaque type d'information d'un livre est stocké dans une ligne.

3. Écrire une procédure qui permet de lire le contenu dans un fichier de bibliothèque en utilisant un pointeur type BibLivres et l'allocation dynamique de la mémoire
4. Écrire une fonction qui permet de chercher les informations d'un livre. Le nom du livre cherché sera donné par l'utilisateur. Si le livre demandé est trouvé, afficher ces informations.
5. Écrire une procédure permettant de modifier certaines informations d'un livre demandé (par exemple si un livre est rendu à la bibliothèque, il faut modifier EnPret, DateDePret,...). Enregistrer ces informations sur le même fichier de source.

Exercice 52. Concaténation de fichiers

Écrire une fonction `int concat(char* sResultat, char* sFichier1, char* sFichier2)` qui permet de concaténer les fichiers dont les noms sont sFichier1 et sFichier2 et mettre le résultat dans le fichier dont le nom est sResultat. La fonction doit retourner 1 si aucune erreur n'est détectée, 0 sinon.

Exercice 53. Copie de fichiers

Écrire une fonction `int coder(int d, char* source, char* dest)` qui permet de copier le fichier de nom `source` dans le fichier de noms `dest` en remplaçant chaque caractère `c` du fichier `source` par le caractère de valeur `(c + d) modulo 256`. Tester cette fonction.

Exercice 54. Statistiques sur les fichiers

Écrire une procédure `void stat_freq(char* nomFichier)` qui affiche tous les caractères figurant dans le fichier de nom `NomFichier` dans un ordre décroissant selon fréquence d'apparition dans le fichier. Tester cette procédure.