Travaux Pratiques d'algorithmique n°5

Cours d'Informatique de Deuxième Année

—DEUG Sciences S3-MI—

Manipulation d'ensembles

Au cours de cette séance, nous allons manipuler des fiches concernant des produits et représentées par des structures. Après avoir implementé les fonctions de base sur ces objets, nous les utiliserons pour gérer un ensemble. Ensuite nous trierons les éléments de l'ensemble. Enfin nous étudirons les entrées/sorties de données sur disque.

Le type `Produit` utilisé durant ce TP est défini de la façon suivante (où `MAXSTR` est la taille maximale d'une chaîne de caractères) :

```
struct produit
{
  char          nom[MAXSTR];
  char          fournisseur[MAXSTR];
  double        prix;
};
typedef struct produit Produit;
```

▸ **Exercice 1.** ***(manipulation de fiches)*** *Écrire les fonctions suivantes :*

- `SaisieProduit` *qui met à jour une fiche passée en argument par les valeurs saisies par l'utilisateur ;*
- `AfficheProduit` *qui affiche le contenu d'une fiche passée en argument ;*
- `CompareProduit` *qui compare les deux fiches passées en argument selon le nom puis le fournisseur (le prix n'est pas pris en compte ici). Elle renverra un nombre négatif, nul ou positif selon que la première fiche est "inférieure" "égale" ou "supérieure" à la seconde ;*
- `CopieProduit` *qui copie dans la première fiche la seconde fiche passée en argument ;*
- `EchangeProduit` *qui échange les deux fiches passées en argument.*

▸ **Exercice 2.** ***(création d'un ensemble)***

Comment créer un tableau de `MAXENS` *produits (où* `MAXENS` *est un constante) ?*

Définir un type `Ensemble` *contenant un tel tableau et sa taille utile (nombre de produits).*

Écrire les fonctions suivantes :

- `InitEnsemble` *qui initialise un ensemble donné en argument à l'ensemble vide ;*
- `int AjouteProduit(???)` *qui ajoute un produit à un ensemble. Si le produit est déjà présent on veillera à ne pas l'insérer deux fois mais à mettre à jour son prix ;*
- `int SaisieEnsemble(???)` *qui met à jour un ensemble par les produits saisis au clavier ;*
- `AfficheEnsemble` *qui affiche l'ensemble passé en argument.*

▸ **Exercice 3.** ***(manipulation d'ensembles)*** *Écrire les fonctions suivantes :*

- `RechercheProduit` *qui recherche dans un ensemble un produit donné. Cette fonction retournera l'indice du produit dans le tableau, ou* -1 *s'il n'est pas présent ;*
- `int SupprimeProduit(???)` *qui supprime un produit donné d'un ensemble ;*
- `AjouteProduitBis` *qui effectue le même travail que la fonction d'ajout de l'exercice 2 mais qui utilise* `RechercheProduit`*.*

▸ **Exercice 4.** ***(entrées/sorties)***

Dans cet exercice, nous allons utiliser les fonctions `fopen`*,* `fprintf`*,* `fscanf` *et* `fclose` *permettant d'écrire et lire des données dans un fichier.*

La fonction `FILE *fopen(char *nom, char *mode);` *permet d'ouvrir un fichier selon différents modes. Les principaux modes sont :* `"r"` *pour la lecture,* `"w"` *pour l'initialisation/écriture et* `"a"` *pour l'ajout/lecture. La fonction* `fopen` *retourne un pointeur sur une structure de type* `FILE` *appellée flux.*

Les fonctions `int fprintf(FILE *f,char *format,...);` *et* `int fscanf(FILE *f, char *format,...);` *ont le même comportement que les fonctions* `printf` *et* `scanf` *excepté le fait qu'elles lisent/écrivent dans le flux passé en premier argument.*

La fonction `int fclose(FILE *f);` *ferme le flux* `f`*.*

Écrire une fonction `int EcritEnsemble(Ensemble e, char *nomFich)` *qui écrit l'ensemble* `e` *dans le fichier dont le nom est founi. Que se passe-t-il si l'on utilise plusieurs fois cette fonction sur le même fichier ?*

Écrire une fonction `int LitEnsemble(Ensemble *e, char *nomFich)` *qui affecte à l'ensemble* `e` *l'ensemble contenu dans le fichier dont le nom est founi. Effectuer les tests nécessaires pour s'assurer que le fichier lu est du bon format. Tester la fonction sur un fichier correct dans son début mais erroné sur sa fin.*

Les fonctions retourneront 0 *en cas de problème,* 1 *sinon.*

▸ **Exercice 5.** ***(tri à bulles de l'ensemble)***

Le tri à bulle consiste à parcourir le tableau en permutant deux à deux les éléments qui ne sont pas dans l'ordre souhaité. Ainsi, lors du premier passage, on fera remonter (comme *une bulle) l'élément le plus grand en haut du tableau. Le second parcours se fait donc sur un tableau de taille $n-1$ et ainsi de suite.*

Écrire la fonction `TriABulle` *triant un ensemble. Évaluer sa complexité.*