Travaux Pratiques d'algorithmique n°8

Cours d'Informatique de Deuxième Année

—Deug SSM M3-MI—

Listes chaînées

Le but de ce TP est de gérer un petit répertoire d'étudiants grâce aux listes chaînées. Les opérations à réaliser seront les suivantes :

- la saisie des coordonnées d'un étudiant ;
- l'ajout d'un étudiant dans le répertoire ;
- la suppression d'un étudiant du répertoire ;
- la visualisation et le tri du répertoire.

▸ **Exercice 1. (saisie des coordonnées d'un étudiant)** *Un étudiant est caractérisé par les informations suivantes :*

- *son nom ;*
- *son prénom ;*
- *son numéro de TP ;*
- *son numéro de TD ;*
- *son numéro de téléphone personnel.*

Créer le type de données `Fiche` *approprié, puis écrire les deux fonctions* `void SaisirFiche(Fiche *f)` *et* `void AfficherFiche(Fiche f)` *pour respectivement, saisir et afficher les coordonnées d'un étudiant.*

▸ **Exercice 2. (Gestion du répertoire)** *Créer le type de données* `Repertoire` *approprié (une liste chaînée de "fiches" d'étudiants), puis écrire les fonctions* `AllouerFiche` *qui alloue une fiche en tête de liste du répertoire, et* `LibererFiche` *qui libère l'espace mémoire consommé par la fiche située en tête de liste du répertoire.*

▸ **Exercice 3.** *Ecrire la fonction* `int FichesIdentiques(Fiche f1, Fiche f2)` *qui teste l'égalité des deux fiches* `f1` *et* `f2`*.*

▸ **Exercice 4. (Gestion d'un étudiant dans le répertoire)** *Ecrire les fonctions suivantes :*

- `Repertoire * AjouterFiche(Repertoire *rep, Fiche f)`;
- `Repertoire * SupprimerFiche(Repertoire *rep, Fiche f)`;
- `void AfficherRepertoire(Repertoire *rep)`;
- `Repertoire * DetruireRepertoire(Repertoire *rep)`.

▸ **Exercice 5. (Tri du répertoire)** *En vous servant des algorithmes de tri que vous avez appris, écrivez une fonction* `TrierRepertoire` *de tri performante du répertoire.*

▸ **Exercice 6. (Rechercher un étudiant dans le répertoire)** *Ecrire la fonction* `RechercheRepertoire` *qui permet de rechercher un (ou plusieurs) étudiant(s) en fonction de ses caractéristiques (son nom, son prénom, son numéro de TP, son numéro de TD). Une manière de faire est de réutiliser le module de saisie des coordonnées d'un étudiant. Les informations saisies sont alors les critères de recherche. Si un champ de saisie n'est pas un critère de recherche, alors il suffira de taper sur retour chariot. On devra pouvoir visualiser simplement les réponses multiples.*

▸ **Exercice 7. (Gestions des notes)** *On veut maintenant gérer les notes des étudiants. Reprendre l'exercice précédent avec les champs de données suivants :*

- *son nom ;*
- *son prénom ;*
- *son numéro de TP ;*
- *son numéro de TD ;*
- *la liste de ses notes de TP ;*
- *la liste de ses notes de TD ;*
- *son numéro de téléphone personnel.*

Quelle structure de données est nécessaire pour modéliser les notes ? Pourquoi ? On veut pouvoir saisir la note d'un TP d'un étudiant donné à n'importe quel moment. Ecrire deux fonctions permettant d'ajouter `AjouterNote` et de modifier `ModifierNote` des notes dynamiquement. Faire un menu général permettant de saisir les informations relatives à un étudiant, ses notes obtenues et de calculer la moyenne de toutes les notes disponibles au moment du calcul. Comment faire pour gérer les absences aux TD ou TP et obtenir des moyennes correctes ?