Année 2018-2019	SNIR 1
Algorithmique	**Pointeurs, structures et énumération**
~ Travaux pratiques sur les pointeurs et structures	

Gestionnaire de club omnisport

1 Mise en situation

Un club omnisports est un club sportif qui gère, sous une même organisation, plusieurs disciplines sportives distinctes. Chaque discipline correspond à une section sportive .

Le gestionnaire en charge de cette organisation souhaite pouvoir gérer les fiches des adhérents licenciés.

Il pourra ajouter, supprimer, modifier ou éditer la fiche d'un adhérant licencié

Il pourra également supprimer ou voir l'ensemble des fiches des adhérents.

2 Fonctionnement de l'application

Les différentes fonctionnalités seront proposées sous forme de menu.
Si aucune fiche n'est présente, seuls les choix « Ajouter » et « Quitter » devront apparaître.

3 Mise en oeuvre

3.1 Énumération des activités possibles

Le club omnisport propose une liste exhaustive de discipline sportive qui sont:

- natation
- basketball
- aviron
- athlétisme
- tennis

1.Proposez un type énumération nommé sports, pour lister l'ensemble des disciplines sportives.

3.2 Structure d'un membre adhérent

Un membre adhérent est caractérisé par:

- son nom
- son prénom
- sa date de naissance
- le nombre des disciplines pratiquées
- un tableau des disciplines pratiquée (5 au maximum dans notre cas) de type enum sport
- un numéro de licence

La date de naissance sera de type typeDate.
Le type typeDate (à définir) est composé de trois champs de type entier:

- jour
- mois
- annee

2.Définissez les types typeDate et typeAdherent.

3.3 Menu

Le menu des choix se présente de deux façons :

```
A : Ajouter une fiche adhérent
Q : Quitter
```

ou, s'il y a déjà des adhérents:

```
A : Ajouter une fiche adhérent
S : Supprimer une fiche
M : Modifier une fiche
V : Voir le contenu d'une fiche
L : Voir le contenu de l'ensemble des fiches
E : Supprimer toutes les fiches
Q : Quitter
```

3.Codez une fonction **afficheMenu** qui réponde au besoin

3.4 Fonctionnalités

4.Ecrivez une fonction **afficherDate** qui prend en paramètres une date (de type typeDate) et affiche celle-ci sous la forme JJ/MM/AAAA

5.Ecrivez une fonction **afficherActivite** qui prend en paramètre un enum sport et affiche le sport correspondant.

6.Codez la fonction ayant pour prototype ***void afficherUnAdherent(typeAdherent *ad)*** qui affiche les informations relatives à l'adhérent passé en paramètre.

7.Codez la fonction ayant pour prototype ***typeAdherent *creerUnAdherent()*** qui va demander les informations relatives à un adhérent et mettre à jour la structure (allouée dynamiquement) correspondante avant de retourner l'adresse de cette dernière.

Le club doit pouvoir gérer jusqu'à 5000 adhérents.

Afin de ne pas surcharger la mémoire, nous utiliserons un tableau de pointeurs de type adherent plutôt qu'un tableau de type adherent.

8.Pour justifier ce choix, donnez la taille en mémoire d'un tableau de pointeur de type adherent et celle d'un tableau de type adherent, chacun ayant 5000 entrées.

9.Déclarez le tableau de pointeurs.

Pour savoir combien d'adhérents sont actuellement dans le tableau, nous utiliserons une variable compteurAdherent de type entier.

10.Déclarez et initialisez cette variable.

11.Codez les fonctions suivantes :

`void afficherAdherents(typeAdherent *ad[],int nb)` : Affiche l'ensemble des fiches des adhérents (de la fiche 0 à la fiche nb-1)

- tab est le tableau de pointeur de type adherent.
- nb est le nombre d'adhérents actuellement dans le tableau.

`int ajouterUnAdherent(typeAdherent *tab[], int nb)` : Ajoute un adhérent à l'indice nb et retourne l'indice du prochain adhérent (→ nb+1)

- tab est le tableau de pointeur de type adherent.
- nb est le nombre d'adhérents actuellement dans le tableau.

12.Codez la fonction ***int supprimerUnAdherent(typeAdherent *tab[],int nro,int nb)*** telle que:

- tab est le tableau des adhérents.
- nro est le numéro de l'adhérent dont la fiche doit être supprimée.
- nb est le nombre d'adhérents actuellement dans le tableau.

La fonction ***supprimerUnAdherent*** doit:

```
rechercher l'indice correspondant au numéro de l'adhérent à supprimer

si le numéro d'adhérent a été trouvé
alors
      supprimer la fiche
      décaler tous les adhérents à partir de l'indice de l'adhérent supprimé
finsi

retourner le nouveau nombre d'adhérents ( → nb-1 )
```

13.Codez une fonction permettant de supprimer toutes les fiches.

14.Codez une fonction permettant de modifier le contenu d'une fiche

15.Proposez une solution pour que le nombre d'adhérents ne soit limité que par les capacités de la machine.