

Un club de loisirs offre un ensemble d'*activités* (lecture, jeux, tennis, ...).
Une personne peut s'inscrire

à une ou plusieurs activités. Pour chaque activité on dispose d'un
cahier d'inscriptions contenant le nom de l'activité et l'ensemble des personnes qui
y sont inscrites.

L'ensemble des cahiers d'inscriptions,

ayant au moins une personne inscrite forme le *dossier des inscriptions courantes*.
Ces notions sont représentées selon les déclarations suivantes :

//Cellule d'une liste simplement chaînée de noms de personnes

struct Cellule

```
{ char NomP[20]//nom de personne (les noms de personnes sont supposés distincts)
  Cellule * PP;    //pointeur sur la personne suivante inscrite
};
```

//Cahier des inscriptions

struct Cahier

```
{ char NomA[30]//nom d'activité (les activités sont supposées distinctes)
  Cellule * LP ;    //donne accès à l'adresse de tête de la liste chaînée des personnes inscrites
};                  à cette activité
```

//Dossier des inscriptions du club : un ensemble de **NbC** cahiers ayant **au moins**
une personne inscrite,

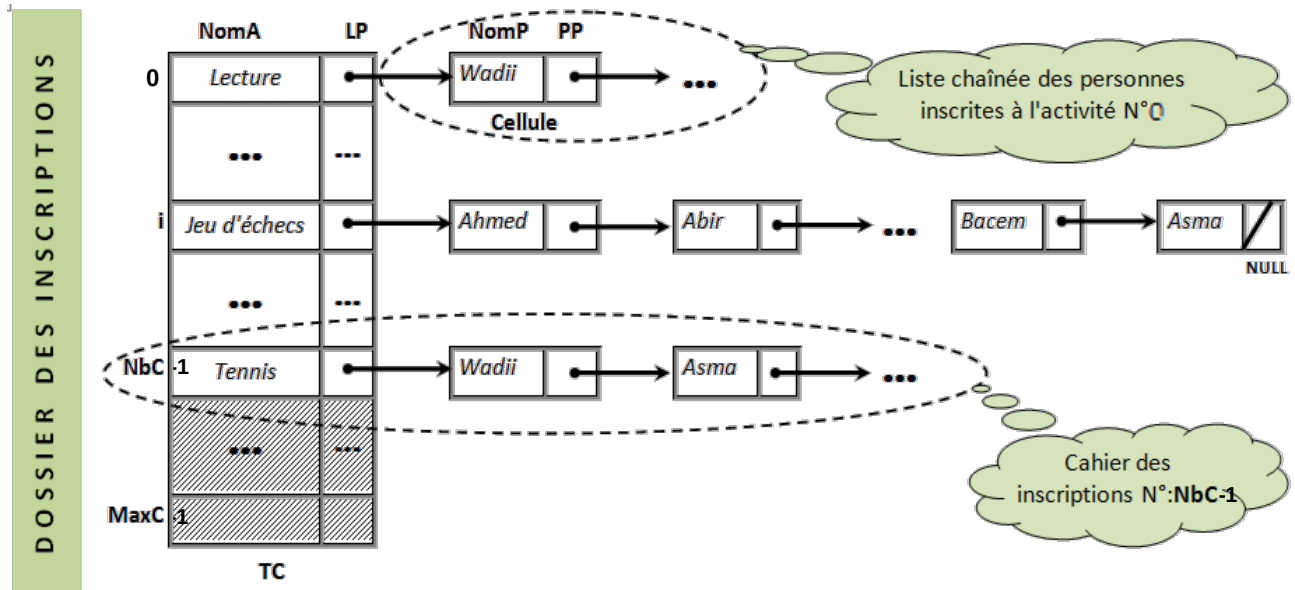
sous forme contiguë dans un tableau nommé **TC**

struct Dossier

```
{ Cahier TC[MaxC]//MaxC: nombre maximum de cahiers (constante globale)
  int NbC ;    //nombre de cahiers des inscriptions courantes. Nbc est compris entre 0 et MaxC-1
};
```

La figure suivante schématise les structures de données nécessaires pour la
réalisation de ce problème :

Exemple de dossier des inscriptions (**D**) aux différentes activités du club



TRAVAIL DEMANDE :

Développer les modules suivants :

1. VERIFICATION DE L'INSCRIPTION D'UNE PERSONNE A UNE ACTIVITE

Cellule * Appartient (char NP[20], Cellule * T)

🔗 Renvoie l'adresse de la cellule qui contient le nom **NP** s'il appartient à la liste des personnes de tête **T**. Sinon, elle renvoie **NULL**.

2. RECHERCHE D'UNE ACTIVITE

int RechercheActivite (Dossier D, char NA[20])

🔗 Retourne l'indice de l'activité nommée **NA** dans le dossier **D** si elle est y présente, sinon elle retourne -1.

3. AJOUT D'UNE PERSONNE

void AjouterPersonne (char NP[20], Cellule * T)

🔗 Permet d'ajouter une personne de nom **NP** à la fin de la liste de tête **T** (la liste est supposée non vide).

4. AJOUT D'UN NOUVEAU CAHIER D'INSCRIPTION

void AjouterCahier (Dossier & D, char NP[20], char NA[20])


🔗 Permet d'ajouter, dans le dossier **D**, un nouveau cahier d'inscriptions à l'activité nommée **NA** avec la personne de nom **NP** comme premier inscrit (c.à.d. l'activité **NA** est ajoutée à la fin du :

tableau **TC** et la personne **NP** est ajoutée en tête de la liste des inscrits).

NB : NbC < MaxC (sinon, ajout impossible !)

5. INSCRIPTION D'UNE PERSONNE A UNE ACTIVITE

void AjouterInscription (Dossier & D, char NP[20], char NA[20])

 Modifie le dossier **D** des **NbC** inscriptions courantes de telle sorte que la personne de nom **NP** apparaisse dans la liste des personnes inscrites à l'activité de nom **NA**.

Deux cas se présentent :

Cas 1: L'activité **NA** est présente dans le dossier **D** :

Cas 1.1: Si la personne **NP** est déjà inscrite à l'activité **NA**, le dossier **D** est **inchangé**

Cas 1.2: Si la personne **NP** n'est pas inscrite à l'activité **NA**, **NP** est ajoutée à la fin de la liste des personnes inscrites à l'activité **NP**

Cas 2: L'activité **NA** n'est pas présente dans le dossier **D** : le cahier correspondant y est ajouté avec **NP** comme premier inscrit.

NB : NbC < MaxC (sinon, ajout impossible !)

6. AFFICHAGE DES INSCRIPTIONS

void AfficherDossier (Dossier D, char NA[20])

 Affiche toutes les personnes inscrites à l'activité **NA** dans le dossier **D**.

7. SUPPRESSION D'UNE PERSONNE

Void SupprimerPersonne (Dossier & D, char NP[20], char NA[20])

 Supprime la personne nommée **NP** (si elle existe) de l'activité **NA** dans le dossier **D**.

8. FONCTION MAIN

Ecrire la fonction main qui permet de :

- Déclarer un dossier D.
- Afficher le menu suivant
 1. Remplissage automatique
 2. Ajouter une inscription
 3. Afficher toutes les activités du club
 4. Afficher les personnes inscrites à une activité
 5. Supprimer une personne d'une activité
 6. Quitter

NB :

- Selon le choix de l'utilisateur (1..5), le programme effectue le traitement correspondant. Le programme prend fin si l'utilisateur saisie 5.
- Pour pouvoir tester facilement les différents modules, on vous donne la procédure qui fait le remplissage automatique (choix 1) :

void RemplissageAutomatique(Dossier & D)

{

//Tableau des activités à insérer dans le dossier

char tabAct[3][20]={"Lecture", "Jeu d'echecs", "Tennis"};

// Tableau des personnes inscrites à l'activité 1 "Lecture"

char PerA1[3][10]={"Wadii", "Youssef", "Asma"};

```

// Tableau des personnes inscrites à l'activité 2      "Jeu d'echecs"
char PerA2[3][10]={"Nour", "Mohamed", "Emir"};

// Tableau des personnes inscrites à l'activité 3      "Tennis"
char PerA3[3][10]={"Wadii", "Aziz", "Sarra"};

int i;    Cellule * cel;
D.NbC=0;

//Ajout de l'activité 1
strcpy(D.TC[0].NomA, tabAct[0]);
D.NbC++;
D.TC[0].LP=NULL;
//Insertion des personnes à l'activité 1. La nouvelle personne est insérée au début de liste.
for(i=2;i>=0;i--)
{ cel = (Cellule*)malloc(sizeof(Cellule));
  if (cel == NULL)
    printf("Allocation impossible !");
  else
  {  strcpy(cel->NomP, PerA1[i]);
    cel->PP = D.TC[0].LP;
    D.TC[0].LP = cel;
  }
}

//Ajout de l'activité 2
strcpy(D.TC[1].NomA, tabAct[1]);
D.NbC++;
D.TC[1].LP=NULL;
//Insertion des personnes à l'activité 2. La nouvelle personne est insérée au début de liste.
for(i=2;i>=0;i--)
{ cel = (Cellule*)malloc(sizeof(Cellule));
  if (cel == NULL)
    printf("Allocation impossible !");
  else
  {  strcpy(cel->NomP, PerA2[i]);
    cel->PP = D.TC[1].LP;
    D.TC[1].LP = cel;
  }
}

//Ajout de l'activité 3
strcpy(D.TC[2].NomA, tabAct[2]);
D.NbC++;
D.TC[2].LP=NULL;

```

//Insertion des personnes à l'activité 3. La nouvelle personne est insérée au début de liste.

```
for(i=2;i>=0;i--)  
{ cel = (Cellule*)malloc(sizeof(Cellule));  
  if (cel == NULL)  
    printf("Allocation impossible !");  
  else  
  { strcpy(cel->NomP,PerA3[i]);  
    cel->PP = D.TC[2].LP;  
    D.TC[2].LP = cel;  
  }  
}
```