INFORMATIQUE

GESTION D'UN CLUB SPORTIF

On rappelle que les opérateurs de comparaison (<, >, <=, >=, =, <>) permettent en Pascal de comparer non seulement des valeurs numériques, mais aussi des chaînes de caractères entre elles. L'ordre ainsi établi pour les chaînes composées de lettres est l'ordre alphabétique. Par exemple, `'DUPONT'<'DURAND'<'DUVAL'` et `'DUCHAMPS'>'DUCHAMP'`. Pour simplifier, on supposera que, dans toutes les chaînes traitées, les lettres sont toujours en majuscules.

On désire réaliser la gestion informatique d'un club sportif. Sur chaque joueur, on voudra stocker un ensemble de renseignements (nom, prénom, date de naissance) d'une part, et savoir s'il a payé sa cotisation d'autre part.

On classifie les joueurs suivant leur année de naissance :

- les joueurs nés en 1972 ou avant 1972 sont seniors et payent 200 F de cotisation.
- les joueurs nés en 1973 ou 1974 sont juniors et payent 150 F.
- les joueurs nés en 1975 ou 1976 sont cadets et payent 120 F.
- les joueurs nés en 1977 ou 1978 sont minimes et payent 100 F.
- les joueurs nés en 1979 ou 1980 sont benjamins et payent 80 F.
- les joueurs nés en 1981 ou après 1981 sont poussins et payent 70 F.

Pour simplifier, nous associerons, à chaque catégorie de joueurs, un numéro de 1 à 6 (les seniors ont la catégorie numéro 1, les juniors la catégorie numéro 2, les cadets la catégorie numéro 3, etc).

Pour traiter ce problème, on définit :

```
const NbMax = 200; {aucun club n'aura plus de MbMax joueurs}
type Mot = string[20];
     Date = record
              Jour, Mois, Annee : integer
            end;
     Joueur = record
                Nom, Prenom     : Mot;
                DateDeNaissance : Date;
                Cotisation      : boolean
              end;
     Club = record
              NbJoueurs : integer;
              Liste     : array [1..NbMax] of Joueur
            end;
```

Précisons que le champ **Cotisation** d'une variable de type **Joueur** a la valeur **true** lorsque le joueur représenté par cette variable a payé sa cotisation.

D'autre part, dans une variable de type **Club**, le champ **NbJoueurs** indique le nombre effectif de joueurs et les joueurs sont rangés dans les éléments de rang **1** à **NbJoueurs** du tableau **Liste**.

Tournez la page S.V.P.

Dans ce tableau, les joueurs apparaissent dans l'ordre alphabétique de leurs noms. Pour simplifier, on supposera que les joueurs ont tous des noms différents.

On rappelle enfin les notations pour l'utilisation des types composés en Pascal. Avec la définition de :

```
var ClubHandBall : Club;
```

la notation

```
ClubHandBall.NbJoueurs
```

désignera le nombre de joueurs du club **ClubHandBall**. De même, la notation

```
ClubHandBall.Liste[3].Cotisation
```

désignera le champ **Cotisation** pour le troisième joueur enregistré dans le club **ClubHandBall**. Enfin, la notation

```
ClubHandBall.Liste[1].DateDeNaissance.Annee
```

désignera l'année de naissance du premier joueur du même club.

1 Opérations de saisie

a) Ecrire la procédure **InitialiserClub** qui prend comme paramètre-résultat une variable **C** de type **Club** et modifie la variable **C.NbJoueurs** pour indiquer que le club n'a encore enregistré aucun joueur.

b) On considère la procédure **SaisirDate** qui permet de saisir une date dans le paramètre **D** :

```
procedure SaisirDate (var D : Date);
   begin
     write ('Année ? '); readln (D.Annee);
     write ('Mois ? '); readln (D.Mois);
     write ('Jour ? '); readln (D.Jour);
   end;
```

On remarquera qu'une date peut être incorrecte pour plusieurs raisons (mois supérieur à 12, numéro du jour impossible pour le mois considéré, etc). On suppose que l'on dispose, pour valider une date saisie, d'une fonction booléenne **EstValide** qui prend comme paramètre un paramètre de type **Date** et renvoie **true** si la date est valide, **false** sinon.
Par exemple, on pourra utiliser cette fonction dans la séquence suivante :

```
var DateDuJour : Date;
     { ... suite des définitions non représentée ici }
SaisirDate(DateDuJour);
if not EstValide(DateDuJour)
then writeln ('Date incorrecte').
```

Ecrire la procédure **SaisirDateValide** qui réalise la saisie d'une date dans son paramètre-résultat **D** de type **Date**. Tant que la date saisie n'est pas valide, la procédure devra le signaler et redemander une autre date. On pourra, pour cela, utiliser la fonction **EstValide** supposée donnée (il n'est pas demandé de donner le contenu de cette fonction).

c) Ecrire une procédure **SaisirJoueur** qui prend comme paramètre-résultat une variable **J**, de type **Joueur**, et permet de saisir les renseignements concernant le joueur décrit par la variable **J**. On vérifiera la validité de la date de naissance, en utilisant la procédure

précédemment écrite. La procédure demandée n'effectuant pas l'enregistrement de la cotisation, elle devra donc initialiser le champ `Cotisation` à la valeur `false`.

2 Catégorie d'un joueur

Ecrire une fonction `Categorie` qui prend comme paramètre une variable `J`, de type `Joueur` et renvoie un entier représentant le numéro de la catégorie à laquelle appartient le joueur décrit par la variable `J` (on remarquera que le numéro de la catégorie d'un joueur né l'année `An` est dans la plupart des cas égal à `(An-1969) div 2)`.

3 Recherche d'un Joueur

Nous rappelons que dans le tableau `Liste` d'une variable de type `Club`, les joueurs apparaissent dans l'ordre alphabétique de leurs noms et qu'il n'y a pas, dans ce tableau, deux joueurs de même nom.

On veut écrire une fonction qui cherche si un joueur appartient à un club. Cette fonction se nommera `ChercherJoueur`. Elle prendra comme paramètres-données les paramètres suivants :

- `C`, de type `Club` ;
- `N`, le nom du joueur recherché, de type `Mot`.

La fonction aura pour paramètre-résultat un entier `P` qui contiendra l'indice du premier joueur dans le tableau `Liste` de `C` qui a un nom supérieur ou égal à `N` dans l'ordre alphabétique. Si un tel joueur n'existe pas, la valeur de `P` devra être `C.NbJoueurs+1`.

Par exemple, si `N` vaut `'DUPONT'` :

- si les noms des joueurs dans le tableau sont (`'DUMONT'`, `'DUPONT'`, `'DUVAL'`, `'VIDAL'`), alors `P` vaudra 2,
- si les noms des joueurs dans le tableau sont (`'DUDUCHE'`, `'DUMONT'`, `'LEGRAND'`, `'PREVOST'`), alors `P` vaudra 3,
- si les noms des joueurs dans le tableau sont (`'DELILLE'`, `DESCHAMPS'`, `'DUMONT'`, `'DUPONT'`), alors `P` vaudra 4.

4 Insertion d'un nouveau joueur

On veut créer une procédure qui permet d'insérer un nouveau joueur `J` dans le tableau `Liste` d'un club `C`. Par exemple, si `J.Nom` vaut `'DUPONT'` :

- si les noms des joueurs dans le tableau sont (`'DUMONT'`, `'DUVAL'`, `'VIDAL'`) alors, au retour de la procédure, les noms dans le tableau devront être (`'DUMONT'`, `'DUPONT'`, `'DUVAL'`, `'VIDAL'`),
- si les noms des joueurs dans le tableau sont (`'DUDUCHE'`, `'DUMONT'`) alors les noms dans le tableau devront être (`'DUDUCHE'`, `'DUMONT'`, `'DUPONT'`),
- si les noms des joueurs dans le tableau sont (`'DUDUCHE'`, `'DUMONT'`, `'DUPONT'`, `'DUVAL'`, `'VIDAL'`) alors le tableau restera inchangé.

En utilisant la procédure construite à la question 3, écrire une procédure `InsererJoueur` qui ajoute un joueur `J` dans le tableau `Liste` correspondant au club `C`. Bien sûr, si `J` est déjà dans le tableau, la procédure ne change rien au tableau `C.Liste`.

Tournez la page S.V.P.

page 408 7/7 - 1992 -
I. N. A. Options : Informatique

5 Total des cotisations

Ecrire une fonction qui calcule, pour un club donné, la somme d'argent rapportée par les cotisations qui ont été payées.

6 Gestion d'un menu

On suppose que, pour un club donné, l'inscription et l'encaissement des cotisations s'effectue en une seule journée[1]. Ecrivez le programme qui :

- initialise une variable de type `Club` sans aucun joueur,
- entre dans une itération qui gère un menu permettant au responsable du club :
  - d'ajouter un joueur au club,
  - de mettre à jour les données lorsqu'un joueur paye sa cotisation,
  - d'afficher la somme totale en caisse (total des cotisations payées),
  - de provoquer la sortie de l'itération pour la fin de l'exécution,
- appelle la procédure `ImprimerClub(C:Club)` avant de terminer l'exécution, pour imprimer les inscriptions enregistrées pour le club `C`.

Dans ce programme, on ne décrira pas la procédure `ImprimerClub` supposée fournie. On utilisera les procédures et fonctions déjà écrites, sans recopier leurs définitions.

7 Mise à jour

Ecrire une procédure `MettreAJour`, dont on déterminera les paramètres, qui supprime d'un tableau de type `Club` tous les joueurs qui n'ont pas payé leur cotisation.

8 Fusion de deux clubs

On suppose maintenant que l'on veut écrire un programme permettant de gérer plusieurs clubs. Sans écrire ce programme, construisez une procédure qui permet de fusionner deux clubs en un troisième club, en conservant l'ordre alphabétique des joueurs.

1 Dans la réalité, l'application devrait utiliser un fichier pour charger les informations nécessaires en début de traitement et les sauvegarder en fin de traitement. La gestion des fichiers n'étant pas au programme du concours, aucune opération de cette sorte n'est demandée ici.

IMPRIMERIE NATIONALE D'après documents fournis partiellement.