PARIS DESCARTES

PROG2 - Programmation impérative

TP 05 – Les types complexes

Julien Sopena

Février 2008

Exercice 1 : Les types énumérés

Question 1

Déclarer un type énuméré nommé **t_ufr** comportant 5 termes qui correspondent aux 5 années de la scolarité $(L1, L2, L3, M1, M2, D)$. Puis, déclarer le type **t_tabUfr** qui permet d'enregistrer le nombre d'inscrits dans chacune des années du LMD.

```
type
   t_ufr = (L1,L2,L3,M1,M2,D) ;
   t_tabUfr = array [t_ufr] of word ;
```

Question 2

Les valeurs d'un type énuméré **n'étant pas des chaînes de caractères**, il faut faire une fonction d'affichage pour le type **t_ufr**. Ce sous programme **afficherLMD** affichera par exemple la chaîne "L1" si on lui passe la valeur $L1$ en argument.

```
procedure afficherLMD (x : t_ufr);
begin
   case x of
      L1 : Write('L1') ;
      L2 : Write('L2') ;
      L3 : Write('L3') ;
      M1 : Write('M1') ;
      M2 : Write('M2') ;
      D : Write('D') ;
   end ;
```

Question 3

Écrire un sous-programme **saisirEffectif** qui permette de saisir au clavier les différentes valeurs d'un tableau tel que celui déclaré dans la question précédente. Il contient l'effectif de **toutes** les années d'enseignement du LMD.

```
procedure saisirEffectif(var tab : t_tabUfr) ;
var
   i : t_ufr ;
begin
   for i := L1 to D do
   begin
      write ('Entrez le nombre d"étudiants de ');
      afficherLMD(i);
      write (' :');
      readln(tab[i]);
   end
end;
```

Question 4

Écrire un sous-programme **effectifTotalUfr** qui calcule la somme des éléments d'un tableau contenant les effectifs de l'UFR.

```
function effectifTotalUfr (tab : t_tabUfr) : word;
var
   i : t_ufr ;
   s : word ;
begin
   s := 0;
   for i := L1 to M2 do
   begin
      s := s + tab[i];
   end ;
   effectifTotalUfr := s;
end;
```

Question 5

Écrire un sous-programme **anneeEffectifMax** qui retourne à partir d'un tableau d'effectif l'année de LMD comportant le plus d'inscrits.

```
function anneeEffectifMax(tab : t_tabUfr) : t_ufr ;
var
   i,max : t_ufr;
begin
   max := L1;
   for i := L2 to D do
      if tab[max] < tab[i] then
         max := i ;
   anneeEffectifMax := max;
end;
```


Question 6

Ecrire un programme utilisant les sous-programmes et les déclarations précédents, qui permette :
— de saisir les effectifs d'une UFR.
— d'afficher le nombre total d'inscrit.
— d'afficher l'année ayant le plus d'étudiant.
— d'afficher les variations (hausse-régression) d'effectif.

```
var
   ufr  : t_tabUfr;
   i  : t_ufr;
 begin
   saisirEffectif(ufr);
   writeln('Effectif total de l UFR :',effectifTotalUfr (ufr));
   write('L"année de plus fort effectif global est : ');
   afficherLMD (anneeEffectifMax(ufr));
   writeln;
   for i  := L2 to D do
   begin
      write('Entre '); afficherLMD ( pred(i) );
      write(' et '); afficherLMD (i);
      if ufr[pred(i)] < ufr[i] then
         writeln(' il y a eu une hausse d"effectif')
      else
         writeln(' il y a eu une baisse d"effectif');
   end;
   readln;
end.
```

Exercice 2 : Les intervalles

Question 1

Donnez les types permetant de décrire une date : seconde, minute, heure, jour, années.

```
type
   t_seconde = 0..59 ;
   t_minute = 0..59 ;
   t_heure = 0..23 ;
   t_mois = 0..11 ;
   t_jour = 0..364 ;
   t_annee = longint ;
```

Question 2

Écrire un sous-programme **afficherDate** qui permet à l'utilisateur d'afficher un nombre de seconde sous la forme d'une date : AAAA année(s) JJJ jour(s) et HH :MM :JJ

```
procedure afficherDate (t  : longint);
var
   sec  : t_seconde ;
   min  : t_minute ;
   hr  : t_heure ;
   j  : t_jour ;
   a  : t_annee ;
begin
   sec  := t mod 60 ;
   t  := t div 60 ;
   min  := t mod 60 ;
   t  := t div 60 ;
   hr  := t mod 24 ;
   t  := t div 24 ;
   j  := t mod 365 ;
   a  := t div 365 ;
   writeln(a,' annee(s) ',j,' jour(s) et ',hr,':',min,':',sec);
end ;
```

Question 3

Écrire un programme qui demande de saisir un nombre de secondes et qui affiche la date correspondante tant que l'utilisateur n'entre pas de valeur nulle.

```
var
   nbSecondes  : longint ;  begin
   repeat
      write ('Entrez un nombre de seconde (0 pour quitter) : ');
      readln (nbSecondes);
      afficherDate(nbSecondes);
   until nbSecondes = 0 ;
end.
```

Exercice 3 : Les ensembles

On dispose d'un tableau contenant les noms des étudiants inscrits (Il y sont au plus 30 étudiants inscrits). On dispose aussi de deux tableaux contenant leurs notes d'examen respectivement en mathématiques et en informatique. Pour étudier leurs résultats on souhaite réaliser des ensembles d'étudiants suivant plusieurs critères.

Question 1

Quels sont les types nécessaires à la réalisation d'un tel programme ?

```
const
   NB_MAX = 30 ;
 type
   t_note = 1..20 ;
```

```
t_indiceEtudiant = 1..NB_MAX ;
t_nomEtudiant = string ;
t_tabNomEtudiant = array [t_indiceEtudiant] of t_nomEtudiant ;
t_tabNotes = array [t_indiceEtudiant] of t_note ;
t_ensEtudiants = set of t_indiceEtudiant ;
```

Question 2

Écrire un sous-programme **sairsirNoms** qui permet à l'utilisateur d'entrer les noms des étudiants et qui retourne le nombre d'étudiants. La saisie s'arrête lorsque l'utilisateur entre une chaîne vide, mais l'on doit s'assurer qu'il entre au moins un nom. (La longueur d'une chaîne est accessible depuis la fonction *length*).

```
function saisirNoms(var tab : t_tabNomEtudiant) : integer ;
var
   i : t_indiceEtudiant;
begin
   repeat
      write ('entrez le nom de l"Ã©tudiant 1 : ');
      readln (tab[1]);
   until length(tab[1]) <> 0 ;
   i := 1 ;
   repeat
      i := i + 1 ;
      write ('entrez le nom de l"Ã©tudiant ',i,' : ');
      readln (tab[i]);
   until (length(tab[i]) = 0) or (i = NB_MAX) ;
   if (length(tab[i]) = 0) then
      i := i-1;
   saisirNoms := i;
end ;
```

Question 3

Écrire un sous-programme **sairsirNotes** qui permet à l'utilisateur d'entrer l'ensemble notes d'une UE. On suppose que tous les étudiants inscrits ont une note.

```
procedure saisirNotes (var notes : t_tabNotes;
                       nom : t_tabNomEtudiant;
                       nb : t_indiceEtudiant) ;
var
   i : t_indiceEtudiant;
begin
   for i := 1 to nb do
   begin
      write ('- entrez la note de ',nom[i],' : ');
      readln (notes[i]);
   end ;
end ;
```

Question 4

Écrire un sous-programme **afficherUnEnsemble** qui permet d'afficher les noms d'un ensemble d'étudiants.

```
procedure afficherUnEnsemble (ens : t_ensEtudiants; nom : t_tabNomEtudiant) ;
var
   i : t_indiceEtudiant;
begin
for i := 1 to NB_MAX do
   if i in ens then
   writeln ('- ',nom[i]) ;
end;
```

Question 5

Écrire un sous-programme **admis** qui permet de construire les ensembles de ceux qui ont eu la moyenne en mathématiques et de ceux qui ont eu la moyenne en informatique .

```
function admis (t : t_tabNotes; n : t_indiceEtudiant) : t_ensEtudiants;
var
   i : t_indiceEtudiant ;
   ensAdmis : t_ensEtudiants ;
begin
   ensAdmis := [];
   for i := 1 to n do
      if (t[i] >= 10) then
         ensAdmis := ensAdmis + [i];
   admis := ensAdmis ;
end ;
```

Question 6

A l'aide de ces fonctions faire un programme qui peremette de saisir les noms et les notes des étudiants, puis affiche successivement les ensembles :

- de tous les étudiants
- des étudiants admis en mathématiques
- des étudiants admis en informatique
- des étudiants admis en mathématiques et en informatique
- des étudiants qui ne sont pas admis mathématiques
- des étudiants admis ni en mathématiques, ni en informatique
- des étudiants qui n'ont été admis qu'à une seule des 2 UE

```
var
   nomDesEtudiants : t_tabNomEtudiant;
   notesEnMath, notesEnInfo : t_tabNotes;
   admisMath, admisInfo, tousEtudiants : t_ensEtudiants;
   nbEtudiants : t_indiceEtudiant;
 begin
   writeln('Saisir les noms des Ã©tudiants (chaÃ®ne vide pour quitter) : ');
   nbEtudiants := saisirNoms(nomDesEtudiants);
   writeln('Saisir les notes de mathÃ©matiques des Ã©tudiants : ');
   saisirNotes(notesEnMath,nomDesEtudiants,nbEtudiants);
   writeln('Saisir les notes d"informatiques des Ã©tudiants : ');
```



```
    saisirNotes(notesEnInfo,nomDesEtudiants,nbEtudiants);
    tousEtudiants := [1..nbEtudiants];
    admisMath := admis(notesEnMath,nbEtudiants);
    admisInfo := admis(notesEnInfo,nbEtudiants);
    writeln ('Tous les etudiants inscrits');
    afficherUnEnsemble(tousEtudiants,nomDesEtudiants);
    writeln ('Les etudiants admis en math');
    afficherUnEnsemble(admisMath,nomDesEtudiants);
    writeln ('Les etudiants admis en info');
    afficherUnEnsemble(admisInfo,nomDesEtudiants);
    writeln ('Les etudiants admis en math et en info');
    afficherUnEnsemble(admisMath*admisInfo,nomDesEtudiants);
    writeln ('Les etudiants admis en info mais pas en math');
    afficherUnEnsemble(admisInfo*(tousEtudiants-admisMath),nomDesEtudiants);
    writeln ('Les etudiants admis en math mais pas en info');
    afficherUnEnsemble(admisMath*(tousEtudiants-admisInfo),nomDesEtudiants);
    writeln ('Les etudiants admis en math ou en info');
    afficherUnEnsemble(admisMath+admisInfo,nomDesEtudiants);
    writeln ('Les etudiants admis ni math et ni info');
    afficherUnEnsemble(tousEtudiants-(admisMath+admisInfo),nomDesEtudiants);
    writeln ('Les etudiants admis dans une seule ue');
    afficherUnEnsemble((admisMath+admisInfo)-(admisMath*admisInfo),nomDesEtudiants);
    readln ;

end.
```

