TP 2 :

Objectifs du TP Ce TP a pour but de

1. mettre en œuvre les fonctions et procédures sur les types ordinaux ;
2. utiliser l'instruction **case** ;
3. réfléchir sur les modes de passage de paramètres.

1 Calendrier

1.1 Préliminaire

Dans ce travail vous allez utiliser les types déclarés ci-dessous.

```
type
  ANNEE       = 1532..9999;
    {1532 début du calendrier grégorien...}
    {préparons le bug de l'an 10000}
  NUMERO_MOIS = 1..12;
  MOIS        = (janvier,fevrier,mars,
                 avril,mai,juin,
                 juillet,aout,septembre,
                 octobre,novembre,decembre);
```

Q 1 . Récupérez le fichier `calendrier.pas` et sauvegardez le dans votre répertoire de travail. Ce fichier contient le squelette d'un programme que vous compléterez.
Q 2 . Complétez l'en-tête.

1.2 Affichage des valeurs du type MOIS

Le type MOIS est un type *énuméré*. Il ne possède donc pas de procédures de sortie. La première chose que vous allez réaliser c'est une procédure d'affichage.
Q 3 . Completez la procédure nommée afficher_mois dont la spécification est la suivante :

```
//-------------------------
// PROCEDURE : afficher_mois
// OBJET :
//   provoque l'affichage du mois
//   passé en paramètre. on ne souhaite
//   pas qu'il y ait de retour à la ligne
//   après cet affichage.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//   { X = janvier}
//   afficher_mois(X); {affiche : janvier)
// CONTRAINTE D'UTILISATION :
//   aucune.
```

Lorsqu'on réalise une procédure ou une fonction, il est nécessaire de la tester.
Q 4 . Copiez dans le programme principal, les instructions suivantes :

```
// tests.
// test exhaustif de la procedure afficher_mois
write('voici la liste des mois séparés par des virgules et terminée par un point : ');
for m := low(MOIS) to pred(high(MOIS)) do
begin
   afficher_mois(m); write(',');
end {for};
afficher_mois(high(MOIS));
writeln('.');
```

Puis compilez et testez votre programme.

1.3 Transformation de MOIS

Q 5 . Complétez la fonction dont la spécification est la suivante :

```
//-------------------------------
// FONCTION : mois_suivant
// OBJET :
//    calcule le mois  qui suit un mois donné.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//    mois_suivant(decembre) vaut janvier
//    mois_suivant(janvier) vaut fevrier
// CONTRAINTE D'UTILISATION :
//    aucune.
function mois_suivant((*mode de passage a completer*) A:MOIS):MOIS;
```

Q 6 . Complétez la procédure dont la spécification est la suivante :

```
//-------------------------------
// PROCEDURE : passer_au_mois_suivant
// OBJET :
//    calcule le mois  qui suit un mois donné.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//    {X = decembre}
//    passer_au_mois_suivant(X)
//    {X = janvier}
//    passe_au_mois_suivant(X)
//    {X = fevrier}
// CONTRAINTE D'UTILISATION :
//    aucune.
procedure passer_au_mois_suivant( (*mode de passage a completer*) A:MOIS);
```

Q 7 . Complétez la fonction dont la spécification est la suivante :

```
//-------------------------------
// FONCTION : mois_precedent
// OBJET :
//    calcule le mois  qui precede un mois donné.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//    mois_precedent(decembre) vaut novembre
//    mois_precedent(janvier) vaut decembre
// CONTRAINTE D'UTILISATION :
//    aucune.
function mois_precedent(const A:MOIS):MOIS;
```

1.4 Numérotation des mois

Q 8 . Implantez la fonction numero_de_mois, puis ajoutez au programme principal son test exhaustif.

```
//-------------------------------
// FONCTION : numero_de_mois
// OBJET :
//    calcule le numero du mois
// EXEMPLE DE COMPORTEMENT ATTENDU :
//    numero_de_mois(janvier) vaut 1
//    numero_de_mois(fevrier) vaut 2
// CONTRAINTE D'UTILISATION :
//    aucune.
```

Q 9 . Implantez la fonction mois_de_numero, puis ajoutez au programme principal son test exhaustif.

```
//-------------------------------
// FONCTION : mois_de_numero
// OBJET :
//    calcule le mois à partir de son numero.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//    mois_de_numero(1) vaut janvier
//    mois_de_numero(2) vaut fevrier
// CONTRAINTE D'UTILISATION :
//    aucune.
```

1.5 Lecture de valeurs du type MOIS

Pour lire une valeur de type mois, on peut procéder de deux façons,
- soit on demande a l'utilisateur de taper le numéro du mois ;
- soit on demande a l'utilisateur de taper le nom du mois en entier.

On va dans la suite implanter ces deux manières.

Q 10 . complétez la procédure dont la spécification est la suivante :

```
//-------------------------------
// PROCEDURE : lire_numero
// OBJET :
//    lit au clavier un entier entre
//    1 et 12. et affecte le parametre
//    avec le mois qui correspond à ce
//    numéro.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//     { m = ?? }
//     lire_numero(m); {l'utilisateur entre la valeur 1}
//     { m = janvier}
//     lire_numero(m); {l'utilisateur entre la valeur 3}
//     { m = mars}
// CONTRAINTE D'UTILISATION :
//    Le numéro entré par l'utilisateur doit être
//    un entier compris entre 1 et 12 au sens large.
//    ( une exception risque de se déclencher sinon)
procedure lire_numero((*mode de passage a completer*) m: MOIS);
```

Q 11 . Est-il facile de procéder à son test exhaustif ?

Q 12 . Ajouter au programme principal les instructions suivantes :

```
writeln('tapez un numero de mois');
lire_numero(m);
```

Q 13 . Que se passe t'il si l'utilisateur entre 13 ?

Q 14 . Réaliser la procédure suivante :

```
//-------------------------------
// PROCEDURE : lire_mois
// OBJET :
//    lit au clavier une chaine de caractere
//    correspondant à un nom de mois. et affecte le parametre
//    avec le mois qui correspond.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//     { m = ?? }
//     lire_mois(m); {l'utilisateur entre la valeur janvier}
//     { m = janvier}
//     lire_numero(m); {l'utilisateur entre la valeur mars}
//     { m = mars}
// CONTRAINTE D'UTILISATION :
//    La donnée entrée par l'utilisateur doit être
//    une chaine en minuscule représentant un mois.
```

```
//    (sinon le contenu de la variable m est indeterminé)
procedure lire_mois((*mode de passage a completer*) m: MOIS);
```

1.6 Nombre de jours dans un mois

On rappelle que lorsqu'une année n'est pas bissextile, le mois de février est composé de 28 jours, les mois d'avril, juin, septembre et novembre sont composés de 30 jours, et les autres mois de 31 jours.
Q 15 . En utilisant l'instruction **case**, complétez la fonction longueur_normale dont la spécification est la suivante :

```
//-------------------------
// FONCTION : longueur_normale
// OBJET :
//    calcule le nombre de jour d'un mois
//    en supposant que l'année en cours n'est pas
//    bissextile.
// EXEMPLE DE COMPORTEMENT ATTENDU :
//    longueur_normale(janvier) vaut 31
//    longueur_normale(fevrier) vaut 28
//    longueur_normale(mars) vaut 31
// CONTRAINTE D'UTILISATION :
//    aucune.
```

Q 16 . Complétez le programme principal de sorte qu'on procéde au test exhaustif de la fonction longueur_normale. Le programme doit produire l'affichage suivant.

```
janvier : 31
fevrier : 28
mars : 31
avril : 30
mai : 31
juin : 30
juillet : 31
aout : 31
septembre : 30
octobre : 31
novembre : 30
decembre : 31
```