Programmation Impérative – Session 2

1ère année
25 Avril 2019. (Durée 1h30)

Tous les programmes demandés **seront écrits en langage algorithmique**. Ces programmes devront respecter scrupuleusement **TOUTES les consignes de bonne programmation définies en cours, TD et TP**
Pour la question E.2, vous répondrez sur la feuille numéro 2 après avoir inscrit votre nom et votre prénom.

1 Questions de cours (4 points)

C.1. Définir un type pour représenter un relevé de températures réelles de tous les jours d'une année. Une seule température sera enregistrée chaque jour.

C.2. Reprendre le type précédent pour 1) avoir l'heure du relevé de chaque température sous la forme heures et minutes (exemples : `10h30`, `0h00`) 2) associer à un relevé un lieu donné (exemple `Toulouse`).

C.3. Qu'est ce que la signature d'un sous-programme ?

C.4. Comment différenciez-vous programmation offensive et programmation défensive ?

2 Trouver les erreurs (5 points)

Considérons les entêtes (signatures) des deux sous-programmes suivants.

```
PROCEDURE F(X : IN OUT ENTIER)

PROCEDURE G(X : IN OUT ENTIER ; N : IN ENTIER)
```

Soient `a`, `M`, `P`, `X` et `Y` des variables de type `ENTIER`, déjà initialisées, et considérons la liste suivante des appels à ces sous-programmes

```
1. G (P, 3)
2. Y := G(Y, M)
3. G (P, 3+X);
4. G (Y*Y, M);

5. G (Y, F(M));
```

Question

E.1. Pour chacun des appels précédents, déterminer si cet appel est correct. Les réponses doivent être justifiées.

Pour la question suivante, considérons le programme Ada `Existe_Elt`.

NOM : **PRENOM :**

```
PROCEDURE Existe_Elt IS
TYPE T_Tableau IS aRRaY (1 .. 10) OF INTEGER ;

FUNCTION Est_Dans (FTab : IN T_Tableau  ; Fe:IN INTEGER) RETURN BOOLEAN IS

BEGIN


   WHILE I <= FTaille   aND      FTab(FIndice) /= Fe   LOOP

         FIndice := FIndice + 1 ;

   END LOOP;


   IF FTab(Findice)= Fe THEN

                       b:= FALSE ;
   ELSE

                       b:= TRUE ;
   END_IF ;

END Est_Dans ;

-- Déclaration des variables
   T : T_Tableau ;

   Elt, Ind, N: INTEGER

BEGIN
--  lecture des éléments du tableau
--  1- Lecture/Entrée  de N nombre d'éléments du tableau T

    LOOP
        Put ("Donner un entier compris entre 1 et 10");
        Get (N) ;
    EXIT WHEN N<=10 and N>=1 ;
    END LOOP ;

--  2- Lecture/Entrée des N éléments entiers positifs  du tableau T

    PUT ("Donner les éléments du tableau ");
    FOR i IN 1..N LOOP
           GET (T(i)) ;
    END LOOP ;

--  3- Lecture/Entrée de l'élément pour lequel on veut savoir s'il est  dans T[1..N]

    PUT ("Donner un entier ");
    GET (Elt) ;

-- 4- Recherche si  Elt dans T[1..N] et affichage du résultat

    Est_Dans(T, Elt,  b);


    IF b = TRUE THEN
            PUT ("La valeur de ", Elt, " est dans le tableau");

    ELSE
            PUT ("La valeur de ", Elt, " n'est pas dans le tableau");

    END_IF ;

END Recherche_Elt;
```

Le programme principal initialise un tableau (1 et 2), demande à l'utilisateur un entier (3), puis affiche si l'élément est présent ou pas (4).

Cet algorithme présente plusieurs erreurs de programmation et conception.

Questions

E.2. Corriger le programme ci-dessus pour qu'il indique bien si un élément est dans un tableau ou pas.

- **On ne demande pas de re-écrire l'algorithme, mais bien de corriger celui qui est donné dans le sujet.**
- **Pour cette question, vous répondrez directement sur la feuille numéro 2 du sujet.**

E.3. En déduire une spécification de la fonction `Est_Dans`

3 Conception par raffinage

Dans cette partie, nous nous intéressons au problème de la recherche de sous-tableaux dans un tableau de caractères. Ce problème correspond à un traitement courant en informatique, en particulier, dans les systèmes de recherche d'informations.

Deux algorithmes, traités sous forme de sous-programmes, sont étudiés dans cette partie.

Pour la suite, on considérera des tableaux de caractères déclarés sous-la forme suivante.

```
CONSTANTE Capacite : ENTIER := 10

TYPE T_Tabl_Caract EST TABLEAU(1..Capacite) DE Caractere

TYPE T_Tab EST ENREGISTREMENT de
 Le_Tab : T_Tabl_Caract
 Taille : Entier    -- Taille >=0 ET Taille <= Capacite
FIN ENREGISTREMENT
```

On dispose d'un tableau T de caractères à une dimension non-totalement rempli. Le tableau peut contenir plusieurs occurrences d'une même valeur.

On souhaite définir un sous-programme ELIMINE permettant de ne conserver dans le tableau T qu'une seule occurrence d'une même valeur.

Pour résoudre ce problème, on n'utilisera pas de fonction intermédiaire ni de tableau intermédiaire.

Exercice 1. Elimination des éléments redondants dans un tableau trié (5 pts)

Dans cet exercice, on supposera que le tableau est **PREALABLEMENT TRIÉ**

L'exemple ci-dessous montre l'utilisation du sous-programme demandé dans le cas d'un tableau TRIÉ.

Le tableau suivant

`Taille =8`

1	2	3	4	5	6	7	8	9	10
'a'	'a'	'a'	'b'	'c'	'c'	'c'	'e'		

devient le tableau
`Taille =4`

1	2	3	4	5	6	7	8	9	10
'a'	'b'	'c'	'e'						

Q1.1. Spécifier un sous-programme qui, à partir d'un tableau donné, fournit, ce même tableau, dans lequel une seule occurrence des éléments initiaux est présente dans ce tableau. **Les éléments ne sont plus redondants mais l'ordre initial est conservé.**

Q1.2. En utilisant la méthode des raffinages, concevoir un algorithme qui implante la spécification issue de la question Q1.1. On décrira les différentes étapes de raffinement ainsi que les différents sous-programmes qui résulteraient de ces raffinages.

Q1.3. Comment faire pour généraliser l'algorithme précédent à des types d'éléments quelconques (par exemple des réels, des complexes, etc.) ?

Exercice 2. Elimination des éléments redondants dans un tableau NON TRIÉ (3 pts)

Nous reprenons la question précédente avec le même objectif d'éliminer les éléments redondants dans un tableau, qui cette fois **N'EST PAS TRIÉ**.

Pour cette question, on **NE TRIERA PAS** le tableau au préalable.

L'exemple ci-dessous montre l'utilisation du sous-programme demandé dans le cas d'un tableau qui n'est pas trié.
Le tableau suivant
`Taille =8`

1	2	3	4	5	6	7	8	9	10
'b'	'a'	'a'	'b'	'f'	'f'	'e'	'f'		

devient le tableau
`Taille =4`

1	2	3	4	5	6	7	8	9	10
'b'	'a'	'f'	'e'						

Q2.1. Spécifier un sous-programme qui, à partir d'un tableau donné, fournit, ce même tableau, dans lequel une seule occurrence d'un élément initial est présente dans ce tableau. Les éléments ne sont plus redondants.

Q2.2. En utilisant la méthode des raffinages, concevoir un algorithme qui implante la spécification issue de la question Q1.. On décrira les différentes étapes de raffinement ainsi que les différents sous-programmes qui résulteraient de ces raffinages

Exercice 3. Puissance entière (3 pts)

Écrire un sous-programme qui calcule la puissance entière d'un nombre réel. On se limitera au cas où l'exposant est positif. Par exemple, si le nombre est 2 et l'exposant est 3, la puissance est 8 (2 * 2 * 2). Si le nombre est 2 et l'exposant est 0, la puissance est 1. On supposera que 0 à la puissance 0 est aussi 1. Ce sous-programme doit être implanté en utilisant la récursivité.