Programmation Impérative

1ère année
Novembre 2017. (Durée 2h)

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 A2, vous répondrez sur la feuille numéro 2 après avoir inscrit vos nom et prénom.

1 Questions de cours (5 points)

Définition de types

C1. Définir un type pour représenter l'année d'étude d'un élève A1, A2, A3

C2. Définir une note avec une valeur (un réel compris entre 0 et 20) et un coefficient (un entier strictement positif). Par exemple 15.5 coefficient 1 et 9.0 coefficient 1 sont deux notes possibles.

C3. Définir un élève caractérisé par un nom, un prénom, un identifiant (entier), une année d'étude et plusieurs notes.

Programmation par contrat

C4. Qu'est ce qu'implique (ou signifie) de définir une pré-condition `Pre_1` sur un sous-programme `SP1` ?

Modules

C5. Expliquer pourquoi on peut être amené à faire apparaître un sous-programme seulement dans l'implantation d'un module et pas dans sa spécification.

2 Trouver les erreurs (6 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` et `Y` des variables de type `ENTIER`, déjà initialisées, et considérons la liste suivante des appels à ces sous-programmes

```
1. F (Y);

2. F (-Y);

3. G (P, A);

4. G (Y*Y, M);

5. G (Y, M-2);
```

A1. Pour chacun des appels précédents, déterminer si cet appel est correct. Les réponses doivent être justifiées.

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

NOM : **PRENOM :**

```
PROCEDURE Recherche_Elt IS

TYPE T_Tableau IS ARRAY (1 .. 10) OF INTEGER ;

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

   BEGIN

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

         FIndice := FIndice + 1 ;

   END LOOP;

   Fe := 0;

END Trouve_Elt ;

-- 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
        LOOP
           GET (T(i)) ;
        EXIT WHEN T(i) > 0 ;
        END LOOP ;
    END LOOP ;


--  3- Lecture/Entrée de l'élément pour lequel
--     on recherche l'indice dans T[1..N]


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


-- 4- Recherche de l'indice Ind de l'élément Elt dans T[1..N]

    Trouve_Elt(T, Elt, N, Ind);

-- Ecriture/sortie de l'indice du tableau ou se trouve l'élément Elt

    PUT ("La valeur de l'indice est ");
    PUT (Ind);

END Recherche_Elt;
```

Le programme principal initialise un tableau (1 et 2), demande à l'utilisateur un entier (3), recherche et affiche l'indice du tableau où se trouve cet entier. Elle affiche 0 si l'entier ne s'y trouve pas.

Cet algorithme présente plusieurs erreurs de programmation et conception.

Question

A2. Corriger le programme ci-dessus pour qu'il réalise bien cette recherche.

Pour cette question, vous répondrez directement sur la feuille numéro 2 du sujet.

A3. En déduire une spécification de la fonction `Trouve_Elt`

3 Conception par raffinage (9 points)

Dans cet exercice, nous nous intéressons à un algorithme de tri : le tri par sélection. Dans la suite, on considèrera le tri par ordre croissant d'un tableau d'entiers de capacité constante, dont les index (indices) commencent à 1. A cet effet, on définira un type `T_Tab` caractérisé par le tableau de capacité constante et le nombre effectif de valeurs enregistrées dans le tableau.

Le principe de cet algorithme consiste à rechercher le minimum d'un tableau et à le placer au début de ce tableau, puis de rechercher le minimum du sous tableau restant et de le placer au début de ce sous-tableau, et ainsi de suite jusqu'à avoir placé la totalité des éléments du tableau.

Questions

Q1. Définir le type `T_Tab` caractérisant les tableaux décrits ci-dessus

Q2. Spécifier un sous-programme qui réalise le tri par sélection d'un tableau donné

Q3. En utilisant la méthode des raffinages, concevoir un algorithme qui implante la spécification issue de la question Q2. On décrira les différentes étapes de raffinement ainsi que les différents sous-programmes qui résulteraient de ces raffinages

Q4. Spécifier et réaliser par la méthode des raffinages un sous-programme nommé `Est_Trie` qui vérifie si un tableau de type `T_Tab` donné est bien trié par ordre croissant.

Q5. Ecrire un algorithme qui décrit un programme principal appelant le sous-programme spécifié en question Q2

Q6. Comment faire pour généraliser l'algorithme de tri à des types d'éléments quelconques (par exemple des réels, des complexes, etc.) ?