Méthodologie de la Programmation

1ère année
17 Décembre 2021. (Durée 1h30)

Tous les programmes demandés **seront écrits en langage algorithmique ou bien en langage Ada**.
Ces programmes devront respecter scrupuleusement **TOUTES les consignes de bonne programmation définies en cours, TD et TP**
.

1 Exceptions (4 points)

On considère le programme suivant.

```
1  with Ada.Text_IO;          use Ada.Text_IO;
2  with Ada.Integer_Text_IO; use Ada.Integer_Text_IO;
3
4  procedure Exemple_Exception is
5
6      ZeroError, UnError, DeuxError: Exception;
7
8      procedure Faire(N: in Integer) is
9      begin
10         if n = 0 then
11             raise ZeroError;
12         elsif n = 1 then
13             raise UnError;
14         elsif n = 2 then
15             raise DeuxError;
16         else
17             Put (N, 1);
18         end if;
19     end Faire;
20
21     procedure Exemple(A : in Integer) is
22     begin
23         Faire (-A);
24         begin
25             Put (':');
26             Faire (A);
27             Put ('+');
28         exception
29             when ZeroError => Put ('Z');
30             when UnError   => Put ('U');
31         end;
32         Put ('F');
33     end Exemple;
34
35     X: Integer;
36 begin
37     Put ("Quel entier ? ");
38     Get (X);
39     Exemple (X);
40 end Exemple_Exception;
```

E.1. Qu'est ce qui s'affiche lors de l'exécution si l'utilisateur répond « 0 » ?
E.2. Qu'est ce qui s'affiche lors de l'exécution si l'utilisateur répond « 1 » ?
E.3. Qu'est ce qui s'affiche lors de l'exécution si l'utilisateur répond « 2 » ?
E.4. Qu'est ce qui s'affiche lors de l'exécution si l'utilisateur répond « 3 » ?

2 Questions de cours (5 points)

C.1. Qu'appelle-t-on « action complexe » dans la méthode des raffinages ?
C.2. Expliquer «flots de données » dans la méthode des raffinages. Quel est leur intérêt ?
C.3. Qu'est ce qu'un module générique ? Quel est l'intérêt ?
C.4. Comparer programmation offensive et programmation défensive ?
C.5. À quoi sert une précondition ? Une postcondition ?

Pour les exercices 3 et 4 suivants, on utilisera la déclaration de tableaux suivante.

```
Capacite : constant Integer := 12 ;

type T_TabPlein is ARRAY (1..Capacite) of Integer ;

type T_Tab is record
    Elements  : T_TabPlein ;
    Taille : Integer ;          -- Taille >=0 ET Taille <= Capacite
end record ;
```

Exercice 3. Recherche d'élément majoritaire (7 points)

Cet exercice traite de la recherche d'un élément majoritaire dans un tableau d'entiers.
On considère un tableau $T[1..Capacite]$ d'entiers contenant $Taille \leq Capacite$ entiers.
Un élément x de $T[1..Taille]$ est un élément majoritaire si et seulement si le nombre d'occurrences de x dans $T[1..Taille]$ est strictement plus grand que $Taille/2$ où / est la division entière.
Le nombre d'occurrences d'un élément x dans un tableau $T[1..Taille]$ désigne le nombre de fois où x apparaît dans $T[1..Taille]$.
Un élément majoritaire n'existe pas forcément, mais s'il existe, alors il est unique.
Exemple 1 : pour le tableau ci-après avec $Taille = 10$ et $Capacite = 12$, l'élément majoritaire est 6, il apparaît 6 fois dans ce tableau.

6	1	6	6	3	6	8	6	6	7		

Exemple 2 : pour le tableau ci-après avec $Taille = 10$ et $Capacite = 12$, il n'y a pas d'élément majoritaire.

6	1	6	7	3	5	8	4	2	7		

Questions

Q3.1 Donner en langage algorithmique le contrat et l'entête d'un sous-programme *Majoritaire* qui calcule l'élément majoritaire dans un tableau s'il existe. Le sous-programme *Majoritaire* signalera aussi que l'élément majoritaire existe ou n'existe pas.

Q3.2 Raffiner le sous-programme *Majoritaire* en langage algorithmique, de manière itérative, depuis le niveau R_0 jusqu'au niveau R_n correspondant à l'algorithme final.

Exercice 4. Compréhension d'un programme (4 points)

On considère la fonction suivante.

```
1  function Mystere(A: in T_Tab; B: in Integer; C: in Integer) return Integer is
2  begin
3      if C < 1 then
4          return 0;
5      else
6          return A.Elements (C) ** B + Mystere (A, B, C - 1);
7      end if;
8  end Mystere;
```

L'opérateur ** correspond à la puissance : `X ** N` correspond à `X` à la puissance `N`.
Répondre de manière concise et précise aux questions suivantes.

1. Cette fonction est-elle récursive ou itérative ?
2. Quel est le résultat de l'appel « Mystere(Tab1, 2, 5) » avec Tab1 un tableau de la forme
   Taille = 9 et Capacité = 12

4	3	2	1	0	-1	-2	-3	-4			

3. Quel argument peut-on donner pour montrer que cette fonction se termine ?
4. Dans un contexte de programmation offensive, quelles sont les préconditions à définir sur A, B et C.
5. Expliquer en français ce que calcule la fonction Mystère.
6. Donner un nom significatif pour Mystère, A, B et C.