EXAMEN DU BACCALAUREAT - SESSION DE JUIN 2010

SECTIONS : Mathématiques + Sciences Expérimentales + Sciences Techniques

Corrigé du sujet théorique d'informatique

Partie I (6 points)

Exercice 1 (3 points)

On suppose qu'un programme principal contient trois sous programmes (une procédure **Proc1**, une fonction **Fonct** et une procédure **Proc2**).
Compléter le tableau suivant par un exemple d'appel de chacun des sous programmes au niveau du programme principal, en se basant sur les entêtes et sur la liste des variables globales disponibles.

Entête du sous programme	Variables globales	Exemple d'appel du sous programme dans le programme principal
Procedure Proc1 (VAR X,Y:integer; Z:real);	A,B : integer M : real L : char Mot : string	***Proc1(A , B , M)*** ***ou*** ***Proc1(B , A , M)***
Function Fonct(X:integer; Z:string):Char;		***L := Fonct(A , Mot)*** ***ou*** ***L := Fonct(B , Mot)***
Procedure Proc2 (Ch:string ; VAR C:char);		***Proc2(Mot , L)***

Exercice 2 (3 points)

Soit le programme Pascal suivant :

```
Program ESSAI ;
Uses wincrt ;
Var  y : integer ;

Function Fonct (a : integer): Char;
Begin
Fonct := Chr(2*a);
End;

Procedure Proc  ;
Var
m : Char;
Begin
m:=Fonct(y);
Writeln(m);
End;

Begin
Readln(y);
Proc;
End.
```

Questions

1) Compléter le tableau suivant par la nature de chaque objet utilisé (objet local ou objet global)

Objet	Nature
y	***Global***
m	***Local***

2) Pour les objets **y**, **m**, **Proc** et **Fonct**, compléter le tableau ci-dessous en mettant une croix (**x**) dans la case correspondante si l'objet est visible par le programme principal "**ESSAI**" ou par les sous programmes :

	Programme principal	Sous programmes	
Objet	**ESSAI**	**Proc**	**Objet**
y	***x***	***x***	***x***
m		***x***	
Proc	***x***		
Fonct	***x***	***x***	

Partie II (14 points)

On se propose d'écrire une analyse et un algorithme d'un programme "**Tri**" qui permet de remplir un tableau **T** par **n** entiers distincts puis de former et d'afficher un autre tableau **Res** qui va contenir les **n** entiers du tableau **T** classés en ordre croissant selon le principe suivant :

Pour chaque élément du tableau **T**

1) Déterminer le nombre **Nbr** d'éléments de **T** qui lui sont inférieurs ou égaux.
2) Placer cet élément dans la position **Nbr** du tableau **Res**.

Exemple : pour les éléments du tableau suivant :

T	3	14	0	9	17	5	8	4
	1	2	3	4	5	6	7	8

- L'entier **T[1] = 3** a **2** éléments qui lui sont inférieurs ou égaux (**3 et 0**), il sera placé dans la position **2** du tableau **Res**.

Res		3						
	1	2	3	4	5	6	7	8

- L'entier **T[2] = 14** a **7** éléments qui lui sont inférieurs ou égaux (**3 , 14 , 0 , 9 , 5 , 8** et **4**), il sera placé dans la position **7** du tableau **Res**.

Res		3					14	
	1	2	3	4	5	6	7	8

- ainsi de suite pour les autres éléments …

Le tableau **Res** aura les éléments placés dans un ordre croissant comme suit :

Res	0	3	4	5	8	9	14	17
	1	2	3	4	5	6	7	8

Questions :

1. Analyser le problème en le décomposant en modules et déduire un algorithme du programme principal.
2. Analyser chacun des modules proposés.

Analyse du Programme Principal

Résultat = Affiche Tableau trié
[]Pour i de 1 à N faire
    Ecrire("RES[",i, "]= ",RES[i])
Traitement :
  2) PROC Trier(N,T,RES)
  1) PROC lecture(N,T)

Types

Types
TAB = Tableau de 50 entiers

Tableau de déclaration des Objets

Objet	Type/Nature	Rôle
N	Entier	Dimension du tableau.
T	Tab	Tableau d'entiers.
RES	Tab	Tableau trié
i	entier	compteur
Trier	procédure	Permet de trier le tableau T dans RES
Lecture	Procédure	Permet la saisie contrôlée de N et le remplissage du tableau T avec contrôle.

Analyse de la procédure Lecture :

```
DEF PROC Lecture(var N:entier;Var T:tab)
2) T= []répéter
        distinct← Vrai
        [] Pour  i de  1 à N  Faire
            T= Donnée ("T[",i, "]= ")
          FinPour
        [] Pour  i de  1 à N  Faire
             [] Pour  j de i+1 à n Faire
                  []Si t[i]= t[j] alors  distinct← faux  FINSI
             FinPour
         FinPour
        [] Si  NON distinct alors Ecrire("Saisir des éléments distincts FINSI
   Jusqu' à   distinct= Vrai
1) N=  []Répéter
        N= donnée("N= ")
        Jusqu'à  (N>=2) et (N<=100)
3) Fin Lecture
```

Tableau de déclaration des Objets

Objet	Type/Nature	Rôle
Distinct	Booléen	Pour s'assurer que le tableau contient des éléments distincts
i	Entier	Compteur
j	Entier	Compteur

Analyse de la procédure Trier:

```
0) DEF PROC Trier(N:entier;T:tab; var RES:TAB)
0) []Pour  i de 1 à  n faire
        [ K←0] Pour j de 1 à n do
              []SI t[j] < t[i] Alors  k← k+1    Finsi
                FinPour
              RES[K+1]← t[i]
   FinPour
1) Fin Trier
```

Tableau de déclaration des Objets

Objet	Type/Nature	Rôle
i	Entier	Compteur
j	Entier	Compteur
K	Entier	L'iIndice de l'emplacement des bonnes places des éléments dans le tableau RES

LES ALGORITHMES :

Algorithme du programme Principal

```
0) Début Prog_Princ
1) PROC Lecture(N,T)
2) PROC Trier(N,T,RES)
3) Pour i de 1 à N Faire
        Ecrire("RES[",i, "]= ",RES[i])
3) Fin Prog_Princ
```

Algorithme de la procédure Lecture

```
0) DEF PROC Lecture(var N:entier;Var T:tab)
1) Répéter
        Ecrire("N= ")
        Lire(N)
   Jusqu'à (N>=2) et (N<=100)
2) Répéter
        Distinct← Vrai
        Pour i de 1 à N Faire
            Ecrire("T[",i, "]= ")
            Lire(T[i])
        FinPour
        Pour i de 1 à N Faire
            Pour j de i+1 à n Faire
                SI t[i]= t[j] alors Distinct← Faux FINSI
            FinPour
        FinPour
        SI (NON distinct) Alors Ecrire("Saisir des éléments distincts") Finsi
  Jusqu' à (distinct= Vrai)
3) Fin Lecture
```

Algorithme de la procédure Trier

```
1) DEF PROC Trier(N:entier;T:tab; var RES:TAB)
2) Pour i de 1 à n faire
        K←0
        Pour j de 1 à n do
            Si T[j] < T[i] Alors
                k← k+1
            FinSi
        FinPour
        RES[K+1]← T[i]
   FinPour
3) FIN Trier
```

Le programme Pascal

```
Program Welcome;
Uses WinCrt;
Type
    TAB = Array[1..100] of integer;
Var
    N,j,k,i : integer;
    T,RES : TAB;
    distinct: boolean;
```

```
Procedure Lecture(Var N:integer ; var T:TAB);
Begin
        Repeat
                 Write('N= ');
                Readln(N);
        Until (N>=2) And (N<=100);
        Repeat
                distinct:= true;
                For i:= 1 to N do
                   Begin
                        Write('T[',i,']= ');
                        Readln(T[i]);
                   End;
                For i:=1 to n do
                        For j:=i+1 to n do
                                if t[i]= t[j] Then  distinct := false ;
                if (Not distinct) Then Writeln(' Saisir des éléments distincts');
        Until distinct= True;
End;

Procedure Trier(N:integer;T:tab;VAR RES:TAB) ;
Begin
    For i:=1 to N do
        Begin
                 k:=0;
                For j:= 1 to n do
                        If t[j] < t[i] Then k:= k+1 ;
                RES[K+1]:= t[i];
        End;
End;

Begin
        Lecture(N,T);
        Trier(N,T,RES);
        For i:= 1 to N do
                Writeln('RES[',i,']= ',RES[i]);
End.
```