Correction Informatique
Session Juin 2009

PARTIE I (8 points)

Exercice N°1 : (1,5 points = 0,25 x 6)

Règles à appliquer :

Si la définition d'un module **M** nécessite la déclaration de **X** objets notés $\mathbf{O_i}$ et de **Y** sous-modules notés $\mathbf{M_i}$, alors les **X** objets $\mathbf{O_i}$ seront reconnus par le module **M** mais également par les **Y** sous-modules $\mathbf{M_i}$.

Application de la règle sur l'objet p :

La définition du module **Somme** nécessite la déclaration de l'objet **p** et d'une fonction **Produit**, donc **p** est reconnu par le module **Somme** mais également par le sous-module **Produit**.

Application de la règle sur les objets q et r :

La définition du module **Produit** nécessite la déclaration des objets **q** et **r** et d'aucun sous-module, donc **q** et **r** sont reconnus par le module **Produit** et **ne seront pas reconnus ailleurs**.

Objet	Reconnu par la fonction	
	Somme	Produit
p	O	O
q	N	O
r	N	O

NB: On n'acceptera que les réponses O/N et Oui/Non

Exercice N°2 : (3,5 points = 0,5 par croix et 0,25 par justification = 0.5*6 + 0.25*2)

Readln(jour) ; Cette instruction **n'est pas valide** car, on ne peut pas lire une variable de type scalaire énuméré.

langue :=Anglais ; Cette instruction est **valide**. En effet, **langue** étant une variable de type **langues_etrangeres** et **Anglais** étant une des valeurs de cet ensemble, l'affectation est possible.

aujourdhui := Dimanche ; Cette instruction est **valide**. En effet, **aujourdhui** étant une variable de type **jour_semaine** et **dimanche** étant une des valeurs de cet ensemble, l'affectation est possible.

Writeln(langue) ; Cette instruction **n'est pas valide** car, on ne peut pas afficher une variable de type scalaire énuméré.

res := aujourdhui<jour ; Cette instruction est **valide**. En effet, il s'agit d'affecter à une variable de type logique (**res**) le résultat de la comparaison **aujourdhui<jour,** qui est un résultat de type **logique**.

Exemple : *si **aujourdhui** vaut **Dimanche** et **jour** vaut **Lundi**, alors aujourdhui<jour est faux, car ord(Dimanche) > ord(jour)*

n := ord(langue) ; Cette instruction est **valide**. En effet, il s'agit d'affecter à une variable de type entier (**n**) le numéro d'ordre (**ord**) du contenu de la variable **langue**. Ce numéro est un entier.

Exemple : *si **langue** vaut **Anglais**, alors ord(langue) = 1 et n vaut 1.*

Exercice N°3 (3 points = 1 + 1 + 1)

1. *Function Essai (ch: string; nb, p: integer): string;* ***(1 pt=2*0,5)***

```
Var
  i : Byte ;      r : string ;   Il est possible d'accepter le type Integer pour la variable i.
Begin
      r := '';
      For i:=1 to length (ch) do
            if i in [p..p+nb-1] then r := r + ch[i];
      Essai := r;
End;
```

2. Le résultat de cette fonction pour les paramètres effectifs suivants :

Ch = 'Protocole' **p**= 3 et **nb**= 4

Ch	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'	'Protocole'
Nb	4	4	4	4	4				
P	3	3	3	3	3				
I	1	2	3	4	5	6	7	8	9
r	''	''	'o'	'ot'	'oto'	'otoc'	'otoc'	'otoc'	'otoc'

La fonction retourne la valeur : **otoc** ***(1 pt)***

3. la fonction **essai** renvoie une sous-chaîne de **nb** caractères et formée à partir d'une chaîne notée **Ch** en commençant à la position **p**. Il s'agit de la fonction prédéfinie **COPY (Ch, p, nb)**. (***1 pt)***

PARTIE II (12 points)

Analyse (8 Pts)	**Algorithme**
• On acceptera toute forme d'analyse descendante. • On acceptera toute solution correcte.	• *Pour être évalué, l'algorithme doit présenter un contenu en relation avec le problème demandé*
PP (1,5 pts) • Cohérence=0,5 • Modularité= 1 Saisie (1,5 pts) • Taille = 0,25*2 • Tableau= 0,25*4	• PP (1 pt) Saisie (1 pt)

Fusion (3 pts)	Fusion (1,5 pt)
• Union = 1 • Ordre = 1 • Redondance = 1 Affichage (1 pts=2*0,5) Divers (1 pts) • Entête des modules (mode de passage, ordre et types de paramètres) • T.D.O et des nouveaux types	Affichage (0,5 pt)

Nom = Distinct		
S	**L.D.E.**	**O.U.**
4	Résultat = Proc Afficher (K, V3)	K, V3
3	(K, V3) = Proc Fusion (N, M, V1, V2, K, V3)	Afficher
2	(N, V1) = Proc Saisie (N, V1)	Fusion
1	(M, V2) = Proc Saisie (M, V2)	N, V1
5	Fin	M, V2, saisie

Tableau de déclaration des nouveaux types

Type
Tabd =Tableau de 20 entiers
Tabr = Tableau de 40 entiers

T.D.O.G

Nom	**Type**	**Rôle**
K	Entier	Longueur du tableau final V3
N	Entier	Longueur du tableau V1
M	Entier	Longueur du tableau V2
V1	Tabd	Tableau donné
V2	Tabd	Tableau donné
V3	Tabr	Tableau final
Afficher	Procédure	Affiche le tableau final
Saisie	Procédure	Saisie d'un tableau et sa longueur
Fusion	Procédure	Fusionne V1 et V2 dans V3 sans redondance

DEF PROC Saisie (var Nb : entier ; var T :Tabd)		
S	**L.D.E.**	**O.U.**
1	Résultat = Nb, T Nb = [ ] Répéter Nb = donnée	

Commentaires

• Validation de la taille des tableaux

	Jusqu'à Nb dans [2..20]	
2	T= [T[1] = donnée]	
	Pour i de 2 à Nb faire	i
	Répéter	
	T[i]=donnée	
	Jusqu'à T[i] > T[i-1]	
	Fin pour	
3	Fin	

- Lecture de l'élément n°1
- Tous les autres éléments doivent être dans un ordre strictement croissant

T.D.O.L

Nom	Type	Rôle
i	**Entier**	**Compteur**

DEF PROC Fusion (N, M : Entier ; V1, V2 : Tabd ; Var K : Entier ; Var V3 : Tabr)

S	L.D.E.	O.U.
	Résultat = K, V3	
1	(K,V3)=[i←1, j← 1, K← 0]	i, j
	Répéter	
	K← K+1	
	Si V1[i]<V2[j] alors	
	V3[K]← V1[i]	
	i ← i+1	
	Sinon	
	Si V1[i]=V2[j] alors	
	V3[K]← V1[i]	
	i← i+1	
	j← j+1	
	Sinon	
	V3[K]← V2[j]	
	j ← j+1	
	FinSi	
	FinSi	
	Jusqu'à (i>N) ou (j>M)	
	Si j>M Alors	
2	Pour c de i à N faire	c
	K← K+1	
	V3[K] ← V1[c]	
	Fin pour	
	Sinon	
	Pour c de j à M faire	
	K← K+1	
	V3[K] ← V2[c]	
	Fin Pour	
	FinSi	
3	Fin	

Commentaires

- L'élément de **V1** est plus petit que celui de **V2**, on range **V1[i]** et on avance dans **V1**
- Ecriture d'un seul élément (**sans redondance**)
- L'élément de **V2** est plus petit que celui de **V1**, on range **V2[j]** et on avance dans **V2**
- Répétition du traitement jusqu'à **la fin de V1 ou de V2**.
- Ecriture des éléments de **V1** (**car c'est la fin de v2**)
- Ecriture des éléments de **V2** (**car c'est la fin de V1**)

T.D.O.L

Nom	Type	Rôle
i	Entier	compteur
j	Entier	compteur
c	Entier	compteur

DEF PROC afficher (Nb : entier ; T :tabr)		
S	**L.D.E.**	**O.U.**
1	Résultat = [ ] Pour i de 1 à Nb faire Ecrire (T[i]) Fin pour	i
2	Fin	

T.D.O.L

Nom	Type	Rôle
I	Entier	compteur

Les algorithmes

Programme Principal

0) Début Distinct
1) Proc Saisie (N, V1)
2) Proc Saisie (M, V2)
3) Proc Fusion (N, M, V1, V2, K, V3)
4) Proc Afficher (K, V3)
5) Fin Distinct

Procédure Saisie

0) **DEF PROC saisie (Var Nb : Entier ; Var T :Tabd)**
1) Répéter
Lire (Nb)
Jusqu'à Nb dans [2..20]
lire(T[1])
2) Pour i de 2 à Nb faire
Répéter
Lire (T[i])
Jusqu'à T[i] >T[i-1]
Fin pour
3) Fin saisie

Procédure Fusion

0) **DEF PROC Fusion (N, M : entier ; V1,V2 : tabd ; var K : entier ; var V3 : tabr)**
1) i← 1, j← 1, K← 0
Répéter
K← K+1
Si V1[i]<V2[j] alors
V3[K]← V1[i]

```
                    i ← i+1
  Sinon
    Si V1[i]=V2[j] alors
                    V3[K]← V1[i]
                    i ← i+1
                    j ← j+1
    Sinon
     V3[K]← V2[j]
     j ← j+1
    FinSI
  FinSi
  Jusqu'à (i>N) ou (j>M)
2) Si j>M Alors
                Pour c de i à N faire
                K← K+1
                V3[K] ← V1[c]
                Fin pour
  Sinon
  Pour c de j à M faire
  K← K+1
  V3[K] ← V2[c]
  Fin Pour
3) FinSi
  Fin Fusion
```

Procédure Afficher

```
0) DEF PROC Afficher ( Nb : entier ;  T :tabr)
1) Pour i de 1 à Nb faire
   Ecrire (T[i])
   Fin pour
2) Fin Afficher
```