ESATIC ECOLE SUPÉRIEURE AFRICAINE DES TIC	Année scolaire 2014 - 2015 PROF : KONAN HYACINTHE LICENCE 1 SRIT

CHAPITRE 7 et 8 : VARIABLES STRUCTUREES TRAVAUX DIRIGES

EXERCICE 1 * ADN

Les bases d'un brin d'ADN sont codées par l'un des quatre caractères 'A', 'C', 'G', ou 'T'.
On peut représenter un brin d'ADN par un tableau contenant une séquence de bases, apparaissant dans un ordre quelconque. On peut alors calculer le brin complémentaire sachant que, dans deux brins complémentaires, les bases 'A' et 'T' se correspondent ainsi que les bases 'C' et 'G'
Ecrire un algorithme qui value un tableau avec les bases d'un brin d'ADN données par l'utilisateur (saisies contrôlées), construit, puis affiche le tableau représentant le brin complémentaire. La fin de saisie sera marquée par l'entrée d'un caractère spécial, par exemple le caractère 'X'.

Exemple :

Brin initial :	A	T	G	A	T	C	C	G	
	1	2	3	4	5	6	7	8	
Brin complémentaire :	T	A	C	T	A	G	G	C	

EXERCICE 2 * Classe d'ages

On veut étudier la répartition d'un échantillon de population dans les neuf classes d'âge définies ci-dessous.

age dans	**classe**
[0, 10[	0
[10, 20[	1
[20, 30[	2
[30, 40[	3
[40, 50[	4
[50, 60[	5
[60, 70[	6
[70, 80[	7
80 et plus	8

Ecrire un algorithme qui :

- saisit une liste d'entiers représentant les âges sans les mémoriser et construit, au fur et à mesure de la saisie, un tableau contenant l'effectif de chaque classe. Une valeur négative terminera la saisie.
- affiche une représentation de ce tableau en affichant une ligne par classe, et, sur chaque ligne un nombre d'étoiles égal à l'effectif de la classe.

Exemple avec la série :
68 92 60 24 71 14 52 12 16 40 80 18 20 40 10 6 48 43 25
on obtiendra la représentation ci-dessous :
classe 0 : *
classe 1 : *****
classe 2 : ***
classe 3 :
classe 4 : ****
classe 5 : *
classe 6 : **
classe 7 : *
classe 8 : **

EXERCICE 3 * Pluviométrie

On dispose d'un ensemble de relevés pluviométriques réalisés sur un territoire donné.
Les relevés ont lieu en différents points de ce territoire. Un relevé est constitué de 3 données inscrites sur une fiche :

le lieu	le numéro du mois	la hauteur de précipitation en ce lieu et **POUR** le mois donné (en mm)

Exemple de relevé :

Dijon	10	150

L'objectif de cet exercice est :

- de **FAIRE** effectuer la saisie des relevés **POUR** les stocker dans trois vecteurs nommés *Lieux*, *Mois*, *Hauteurs*. Cette saisie se fera par une **PROCEDURE** nommée *SaisirReleves* qui remplira les trois vecteurs et calculera le nombre de relevés *Nbsaisis*.
- d'exploiter ces données **POUR** remplir un vecteur nommé *Somme* qui contiendra **POUR** chaque mois la somme des hauteurs de précipitation enregistrées dans tous les lieux. Cette exploitation se fera à l'aide d'une **PROCEDURE** nommée *ExploiterReleves*.
- d'afficher le contenu du vecteur *Somme* avec une **PROCEDURE** *AfficherSomme*.

L'action *SaisirReleves* demande à l'utilisateur une suite de relevés (lieu, n° du mois, hauteur) et range ces relevés dans 3 vecteurs *Lieux*, *Mois*, *Hauteurs* de sorte que "i, *Lieux[i]*, *Mois[i]*, *Hauteurs[i]* relèvent de la même fiche.
Le nombre de relevés n'est pas connu a priori par l'utilisateur, celui-ci devra donc arrêter la saisie par une chaîne de caractères fictive, par exemple 'Z'. L'action devra aussi, en cours de saisie, compter le nombre de relevés **POUR** valuer la variable *Nbsaisis.*.
De plus la saisie d'un numéro de mois d'une part et celle de la hauteur d'autre part devront **FAIRE** l'objet d'une **PROCEDURE** de saisie avec contrôle de réponse (entre 10 et 1000 **POUR** la hauteur).

Exemple :

Exemple :

	Lieux
1	DIJON
2	NANTES
3	NANTES
4	DIJON
5	LYON
6	VIERZON

	Mois
1	10
2	5
3	10
4	5
5	1
6	8

	Hauteurs
1	150
2	80
3	200
4	50
5	100
6	80

	Somme
1	100
2	0
3	0
4	0
5	130
6	0
7	0
8	80
9	0
10	350
11	0
12	0

L'action *ExploiterReleves* prend les données nécessaires dans les vecteurs *Mois*, *Hauteurs* et dans la variable *Nbsaisis* et remplit le vecteur *Somme* comme indiqué ci-dessus.

EXERCICE 4 ** Pluviométrie 2

Refaites l'exercice précédent (pluviométrie 1) en utilisant la notion de structure.

FIN

ESATIC ECOLE SUPÉRIEURE AFRICAINE DES TIC	Année scolaire 2014 - 2015 PROF : KONAN HYACINTHE LICENCE 1 SRIT

CHAPITRE 7 et 8 : VARIABLES STRUCTUREES TRAVAUX DIRIGES

CORRECTION

EXERCICE 1 * ADN

```
ALGORITHMIQUE ADN
        CONSTANTES
        TAILLE_MAX = 100 /* taille maximale d'un brin d'ADN */
        TYPES
                t_ADN = TABLEAU[1..TAILLE_MAX] de caractere
        VARIABLES
        brinInitial : t_ADN /* le brin initial d'ADN */
        brinComplement : t_ADN /* le complémentaire du brin initial */
        nbBases : ENTIER /* nombre de bases dans les brins */
DEBUT
        remplir(brinInitial, nbBases)
        construire(brinInitial, nbBases, brinComplement)
        afficher(brinInitial, nbBases)
        afficher(brinComplement, nbBases)
FIN

/* Value un brin d'ADN */
PROCEDURE remplir( sortie brin : t_ADN, sortie nb : ENTIER)
        VARIABLES locales
                base : CARACTERE
DEBUT
        nb ← 0
        obtenirBase(base) /* on arrête avec 'X' */
        TANTQUE base != 'X' et nb < TAILLE_MAX FAIRE
                nb ← nb + 1
                brin[nb] ← base
                obtenirBase(base)
        FINTANTQUE
FIN
/*
* Saisit un CARACTERE et le redemande jusqu'à ce qu'il soit un 'X'
* ou corresponde à une base
*/
```

```
PROCEDURE obtenirBase (sortie base : CARACTERE)
DEBUT
        ECRIRE('Base ou X POUR terminer : ')
        LIRE(base)
        TANTQUE base != 'A' et base != 'T' et base != 'G' et base != 'C' et base != 'X' FAIRE
                ECRIRE ('erreur : répondre par A, T, G, C ou Z POUR terminer')
                LIRE(base)
        FINTANTQUE
FIN
```

NB : on aurait pu utiliser un tableau pour stocker les bases et le caractère d'arrêt X

```
/*
* Construit le brin complémentaire de brin qui possède nb bases
*/
PROCEDURE construire ( entrée brin : t_ADN, entrée nb : ENTIER, sortie compl : t_ADN)
VARIABLES locales
        i : ENTIER
DEBUT
        POUR i de 1 à nb FAIRE
                SELON brin[i] DANS
                        'A' : compl[i] ← 'T'
                        'T' : compl[i] ← 'A'
                        'C' : compl[i] ← 'G'
                        'G' : compl[i] ← 'C'
                FINSELON
        FINPOUR
FIN

/* Affiche un tableau de caractères = un brin d'ADN
*/

PROCEDURE afficher ( entrée t : t_ADN, entrée nb : ENTIER)
VARIABLES locales
        i : ENTIER
DEBUT
        POUR i de 1 à nb FAIRE
                ECRIRE(t[i], " , ")
        FINPOUR
        ECRIRE(CRLF) /* passage à la ligne */
FIN
```

EXERCICE 2 * Classe d'ages

on suppose que les classes d'âge sont des intervalles qui vont * de 10 en 10. Il y a 9 classes d'âge

E0

```
ALGORITHMIQUE ClassesDAge
        CONSTANTES
                NB_CLASSES = 9
                TAILLE_CLASSES = 10
        TYPES
                t_classes = TABLEAU[1..NB_CLASSES] d'ENTIER
        VARIABLES
                classes : t_classes
DEBUT
        calculer(classes)
        afficher(classes)
FIN
```

```
/*
* saisit un certain nombre d'âges ENTIERs jusqu'à une saisie
* négative et calcule les effectifs des classes d'âge au fur et à
* mesure
*/
PROCEDURE calculer(sortie t : t_classes)
        VARIABLES locales
                i : ENTIER
                age : ENTIER
DEBUT
        LIRE(age)
        TANTQUE age >= 0 FAIRE
                classe ← classement(age)
                t[classe] ← t[classe] + 1
        FINTANTQUE
FIN

/*
* retourne l'indice de la classe d'âge en fonction de l'âge passé
* en paramètre
*/

FONCTION classement(age : ENTIER) : ENTIER
        VARIABLES locales
                i : ENTIER
                clas : ENTIER
DEBUT
        clas ← age div 10
        SI clas = 0 ALORS
                clas ← 1
        SINON
                SI clas > 9 ALORS
                        clas ← 9
                FINSI
        FINSI
        RETOURNER clas
FIN
```

```
/*
* retourne une chaine composée de n étoiles
*/
fonction etoiles(n : ENTIER) : CHAINE
        VARIABLES locales
                ch : CHAINE
                i : ENTIER
DEBUT
        POUR i de 1 à n FAIRE
                ch ← ch + "*"
        FINPOUR
        RETOURNER ch
FIN

/*
* affiche les classes d'âge
*/

PROCEDURE afficher(entrée t : t_classes)
        VARIABLES locales
                i : ENTIER
DEBUT
        POUR i de 1 à 9
                ECRIRE("classe ", i-1, " : ", etoiles(classes[i]))
        FINPOUR
FIN
```

EXERCICE 3 * Pluviométrie

```
ALGORITHMIQUE Pluviométrie1
        CONSTANTES
                TAILLE_MAX = 500 /* choix arbitraire */
        TYPES
                t_tabLieu = TABLEAU[1.. TAILLE_MAX] de chaînes de CARACTEREs
                t_tabENTIER = TABLEAU[1.. TAILLE_MAX] d'ENTIER
                t_tabMois = TABLEAU[1..12] d'ENTIER
        VARIABLES
                lieux : t_tabLieu /* les lieux des relevés */
                mois : t_tabENTIER /* les mois des relevés */
                hauteurs : t_tabENTIER /* les hauteurs des relevés */
                nbSaisis : ENTIER /* nombre de relevés */
                somme : t_tabMois /* somme des hauteurs par mois */
DEBUT
        saisirReleves(lieux, mois, hauteurs, nbSaisis)
        exploiterReleves(mois, hauteurs, nbSaisis, somme)
        afficherSomme(somme)
FIN
```

/* saisit une séquence de relevés et value les vecteurs. Le dernier relevé saisi n'est pas mémorisé. La **PROCEDURE** obtenirEntre saisit un nb compris entre deux bornes */

PROCEDURE saisirReleves(sortie lieu : t_tabLieu, sortie mois : t_tab**ENTIER**, sortie hauteurs : t_tab**ENTIER**, sortie nbSaisis : **ENTIER**)

```
    VARIABLES locales
        lieuSaisi : CHAINE
DEBUT
    nbSaisis ← 0
    LIRE(lieuSaisi) /* acquérir premier élément */
    TANTQUE lieuSaisi != 'Z' et nbSaisis < TAILLE_MAX FAIRE
        nbSaisis ← nbSaisis + 1 /* traiter élément courant */
        lieu[nbSaisis] ← lieuSaisi
        obtenirEntre(mois[nbSaisis], 1, 12)
        obtenirEntre(hauteur[nbSaisis], 10, 1000)
        LIRE(lieuSaisi) /* acquérir élément suivant */
    FINTANTQUE
FIN
```

/* renvoie un **ENTIER** obtenu par saisie, compris entre inf et sup */

```
PROCEDURE obtenirEntre( sortie x : ENTIER, entrée inf : ENTIER, entrée sup : ENTIER)
DEBUT
    répéter
        LIRE(x)
    jusqu'à (x >= inf) et (x <= sup)
FIN
```

/* parcourt les **TABLEAU**x mois et hauteur et value le vecteur somme en cumulant les hauteurs par mois */

PROCEDURE exploiterReleves(entrée mois : t_tab**ENTIER**, entrée hauteurs : t_tab**ENTIER**, entrée nbSaisis : **ENTIER**, sortie somme : t_tabMois)

```
    VARIABLES
        i : ENTIER
DEBUT
    /* initialisation du vecteur somme */
    POUR i de 1 à 12 FAIRE
        somme[i] ← 0
    FINPOUR
    /* parcours de tous les relevés */
    POUR i de 1 à nbSaisis FAIRE
        somme[mois[i]] ¬← somme[mois[i]] + hauteur[i]
    FINPOUR
FIN
```

```
/* affiche le contenu du vecteur somme */
PROCEDURE afficherSomme(entrée somme : t_tabMois)
        VARIABLES
                i : ENTIER
DEBUT
        POUR i de 1 à 12 FAIRE
                ECRIRE("mois n° ", i, somme[i])
        FINPOUR
FIN
```

EXERCICE 4 ** Pluviométrie 2

```
ALGORITHMIQUE Pluviométrie2
        CONSTANTES
                TAILLE_MAX = 500 /* choix arbitraire */
        TYPES
                t_releve = ENRG /* un relevé */
                        lieu : CHAINE
                        mois : ENTIER
                        hauteur : ENTIER
                FINENRG

                t_tabReleves = TABLEAU[1.. TAILLE_MAX] de t_releve
                t_tabMois = TABLEAU[1..12] d'ENTIER

        VARIABLES
                releves : t_ tabReleves/* les relevés */
                nbSaisis : ENTIER /* nombre de relevés */
                somme : t_tabMois /* somme des hauteurs par mois */
DEBUT
        saisirReleves(releves, nbSaisis)
        exploiterReleves(releves, nbSaisis, somme)
        afficherSomme(somme)
FIN

/* saisit une séquence de relevés. */
PROCEDURE saisirReleves( sortie releves : t_tabReleves, sortie nbSaisis : ENTIER)
        VARIABLES locales
        lieuSaisi : CHAINE
DEBUT
        nbSaisis ← 0
        LIRE(lieuSaisi) /* acquérir premier élément */
        TANTQUE lieuSaisi != 'Z' et nbSaisis < TAILLE_MAX
                nbSaisis ← nbSaisis + 1 /* traiter élément courant */
                releves[nbSaisis].lieu ← lieuSaisi
                obtenirEntre(releves[nbSaisis].mois, 1, 12)
                obtenirEntre(releves[nbSaisis].hauteur, 10, 1000)
                LIRE(lieuSaisi) /* acquérir élément suivant */
        FINTANTQUE
FIN
```

```
/* renvoie un ENTIER obtenu par saisie, compris entre inf et sup */

PROCEDURE obtenirEntre( sortie x : ENTIER, entrée inf : ENTIER, entrée sup : ENTIER)
DEBUT
        répéter
                LIRE(x)
        jusqu'à (x >= inf) et (x <= sup)
FIN

/* parcourt les TABLEAUx mois et hauteur et value le vecteur somme en cumulant les hauteurs par
mois */

PROCEDURE exploiterReleves(entrée releves : t_tabReleves, entrée nbSaisis : ENTIER,
sortie somme : t_tabMois)
        VARIABLES locales
                i : ENTIER
DEBUT
        /* initialisation du vecteur somme */
        POUR i de 1 à 12 FAIRE
                somme[i] ← 0
        FINPOUR
        /* parcours de tous les relevés */
        POUR i de 1 à nbSaisis FAIRE
                somme[releves[i].mois] ← somme[releves[i].mois] + releves[i].hauteur
        FINPOUR
FIN

/* affiche le contenu du vecteur somme */
PROCEDURE afficherSomme(entrée somme : t_tabMois)
        VARIABLES locales
                i : ENTIER
DEBUT
        POUR i de 1 à 12 FAIRE
                ECRIRE("mois n° ", i, somme[i])
        FINPOUR
FIN
```

FIN