UNIVERSITE IBN TOFAIL
Faculté des sciences
Département d'Informatique
Kenitra

Année : 2022/2023
Filières : SMI
Semestre : 3

Algorithmique II

Examen de rattrapage

Durée : 1h 30 mn

Exercice 1 : (Sur 7 points)

1. Définir le type Personne en tant que structure ayant pour champs :
   - Code de type entier,
   - Nom de type chaine de caractères,
   - Date_de_naissance de type tableau de trois entiers (jour, mois et année dans cet ordre).
2. Ecrire la procédure PlusAgee(T : Personne[1..N]) qui reçoit comme paramètre un tableau T de N Personne, détermine l'indice dans T de la personne la plus âgée du tableau et afficher le Code, le Nom et la Date_de_naissance de cette personne (s'il y'en a plusieurs personnes ayant la même date de naissance et qui sont les plus âgées, on choisit celle rencontrée en premier lors du parcours du tableau T, du début vers la fin).

Corrigé :

```
1.  Type Personne = Structure
                    Code : Entier
                    Nom : Chaine
                    Date_de_naissance : Entier[1..3]
     Fin Structure

2.
Procedure PlusAgee (T : Personne[1..N])
Var     i, k : Entier
Début
    //Recherche de la personne la plus âgée de T (s'il y'en a plusieurs on choisit celle
    //rencontré en premier)
    k←1
    Pour (i←2 à N) Faire
        Si (T[i].Date_de_naissance[3] < T[k].Date_de_naissance[3]) Alors
                k←i
        Sinon
                Si (T[i].Date_de_naissance[3] = T[k].Date_de_naissance[3]) Alors
                    Si (T[i].Date_de_naissance[2] < T[k].Date_de_naissance[2]) Alors
                        k←i
                    Sinon
                        Si (T[i].Date_de_naissance[2] = T[k].Date_de_naissance[2]) Alors
                            Si (T[i].Date_de_naissance[1] < T[k].Date_de_naissance[1]) Alors
                                k←i
                            Fin Si
                        Fin Si
                    Fin Si
                Fin Si
        Fin Si
    Fin Pour
```

```
    //Affichage des résultats
    Ecrire ("\nLa personne la plus âgée est :")
    Ecrire ("Code : ", T[k].Code,"\tNom : ",T[k].Nom,"\tDate de naissane : ")
    Ecrire ( T[k].Date_de_naissance[1],"/", T[k].Date_de_naissance[2],"/")
    Ecrire ( T[k].Date_de_naissance[3])
Fin
```

Exercice 2 : (Sur 5 points)

Ecrire la procédure plusLongSequence(Tab : Entier[1..n]) qui affiche le couple constitué de l'indice du premier élément d'une plus longue séquence d'éléments identiques du tableau Tab, ainsi que la taille de cette séquence. Si par exemple, Tab=[1,1,5,3,4,4,4,7,7,7,7,2] la procédure plusLongSequence(Tab) affichera le couple (8, 4).

Corrigé

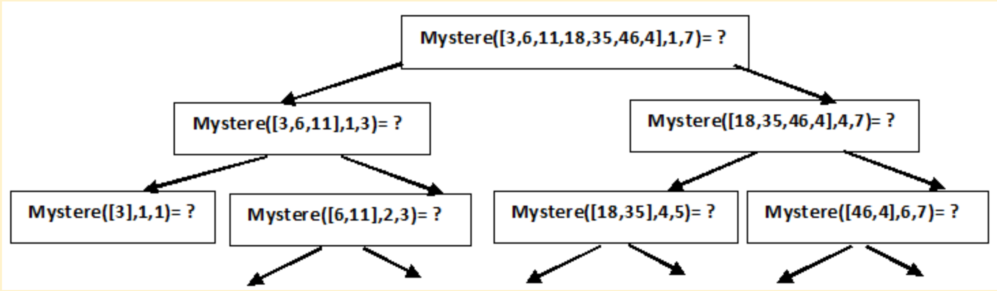
```
Procédure plusLongSequence(Tab : Entier[1..n])
Var      i, j, Taille, k : Entier
Début
    Taille ← 1      //taille de la séquence d'éléments identiques de Tab
    k←1             //l'indice du début de la séquence d'éléments identiques
    i←1
    Tant que (i<n-1) Faire
        j←i+1
        Tant que (j<=n et Tab[j]=Tab[i]) Faire
            j←j+1
        Fin Tant que
        Si (j-i > Taille) Alors
            Taille ← j-i
            k←i
        Fin Si
        i←j
    Fin Tant que
    Ecrire ("(",k,", ",Taille,")")
Fin
```

Exercice 3 : (Sur 9 points)

On considère la fonction récursive Mystere donnée par :

```
Fonction Mystere(T : Reel[1..N], deb : Entier, fin : Entier) : Reel
Var      d : Entier
         m1, m2 : Reel
Début
         Si (deb=fin) Alors
                  Retourner T[deb]
         Fin Si
         d←(deb+fin) Div 2
         m1←Mystere(T, deb, d)
         m2←Mystere(T, d+1, fin)
         Si (m1 < m2) Alors
                  Retourner m2
         Sinon
                  Retourner m1
         Fin Si
Fin
```

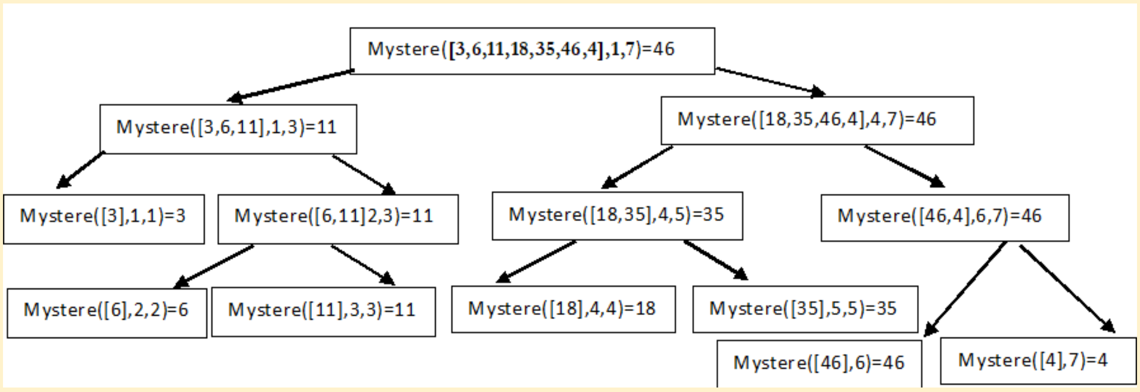
1. Appliquer la fonction Mystere au tableau T=[3,6,11,18,35,46,4], en dressant l'arbre des appels récursifs, comme ci-dessous, indiquant les valeurs retournées après chaque appel de la fonction Mystere aux sous-tableaux de T (obtenus par divisions successives en deux des sous-tableaux de T).



2. Montrer en utilisant un raisonnement par récurrence sur la taille du tableau T que la fonction Mystere retourne le maximum de T.
3. Déterminer la complexité temporelle dans les pires des cas de la fonction Mystere.

Corrigé

1. Application de la fonction Mystere au tableau T=[3,6,11,18,35,46,4]



2. Montrons par récurrence sur la taille du tableau T que la fonction Mystere retourne le maximum de T.
   Il est clair que si la taille de T est égale à 1 (lorsque deb=fin) la fonction retourne le maximum du tableau T.
   Supposons que ceci est vrai pour tout tableau T de taille inférieure à n et appliquons la fonction Mystere à un tableau T[deb..fin] de taille n+1, avec n≥1.
   Soit d=(deb+fin)/2, alors les deux sous-tableaux T[deb..d] et T[d+1..fin] sont de tailles inférieures à n. Donc, d'après l'hypothèse de récurrence, m1 et m2 sont respectivement les maximums des sous-tableaux T[deb..d] et T[d+1..fin]. Et comme la réunion de ces deux sous-tableaux est égale au tableau T alors le maximum de T est égal à max(m1, m2) et c'est bien la valeur retournée par la fonction Mystere appliquée au tableau T[deb..fin].
3. Soit t(n) la complexité temporelle de la fonction Mystere, où n est la taille du tableau T.
   t(n)=tcomp + tretour = C0, une constante réelle,   si n= 1
   t(n)=2*tcomp + 3*taffect + tdiv + taddi + tretour + 2*t(n/2) = C1 + 2*t(n/2), si n>1, où C1 est une constante réelle.

   D'où la relation de récurrence :
   t(n)=C0,   si n= 1
   t(n)=C1 + 2*t(n/2), si n>1

   Pour simplifier le calcul, on suppose que $n=2^k$, on a alors :
   t(n)=C1 + 2*t(n/2)
   $2*t(n/2)=2*C1 + 2^2*t(n/2^2)$

$2^2*t(n/2^2)=2^2*C1 + 2^3*t(n/2^3)$

……………………………………

$2^{k-1}*t(n/2^{k-1})=2^{k-1}*C1 + 2^k*t(n/2^k)$

$t(n)=C1(1+2+..+2^{k-1}) + 2^k*t(1)$

$t(n)=(2^k - 1)C1 + 2^k*C0 = (C1+C0)*n - C1 = \theta(n)$

D'où finalement, $t(n)= \theta(n)$