UNIVERSITE DE JIJEL

FACULTE DES SCIENCES EXACTES ET INFORMATIQUE

DEPARTEMENT D'INFORMATIQUE

Février 2020

Examen d'algorithmique et structures de données
(Durée 1H45)

Exercice 1 (5 points):

Soit la procédure PA suivante :

```
Procedure PA (Val CH : chaîne de caractères, Ref A : ABR)
  Variables
    I, N: Entier
  Début
    N ← Taille(CH)  {le Nombre de caractères de la chaîne CH}
    Pour I ← 1 Jusqu'à N Faire
          InsererABR(CH[I], A)
    FPour
  Fin
FinFonction
```

Soit AB un arbre binaire de recherche de caractères. Dessinez l'arbre AB après l'exécution de chacune des instructions (I1, I2, I3 et I4) de l'algorithme suivant:

```
Début
   AB ← Nil  .....(I1)
   PA("EXAMENASDLMD",AB).....(I2)
   SupprimerABR('E',AB).....(I3)
   SupprimerABR('X',AB).....(I4)
Fin
```

Exercice 2 (7 points)

Soit Tab un tableau de N entiers trié en ordre croissant.

1. Ecrivez une fonction récursive efficace **TestExist** qui teste l'existence d'une valeur donnée X dans Tab.
2. Ecrivez une procédure récursive efficace **AjouterTab** qui permet d'Ajouter une valeur X dans Tab de telle sorte que Tab reste toujours un tableau trié. On suppose que la valeur de N est inférieure à la taille maximale du tableau Tab.
3. Exprimez les complexités temporelles asymptotiques de la fonction **TestExist et** de la procédure **AjouterTab** sous forme d'équations de récurrences.

N.B : Pour les questions 1 et 2 écrivez la meilleure solution possible.

Exercice 3 (8 points)

1. On se donne le graphe orienté G sur l'ensemble de sommets **S = {A, B, C, D, K, L, M, N, P}** dont la matrice d'adjacence est donnée ci-dessous:

	A	B	C	D	K	L	M	N	P
A	0	0	0	1	0	0	0	0	0
B	0	0	0	1	0	0	0	0	0
C	1	0	0	0	0	0	0	0	0
D	1	1	0	0	0	0	0	0	0
K	0	0	0	0	0	0	1	1	0
L	0	0	0	0	0	0	0	1	0
M	0	0	0	0	0	1	0	1	0
N	0	0	0	0	0	0	0	0	0
P	0	0	0	0	0	1	0	1	0

Représentez le résultat du parcours en profondeur du graphe G, à partir du sommet N, sous forme d'une arborescence ou d'une forêt.

2. On considère le cas des graphes orientés, simples, non valués d'ordre N. On considère l'implémentation avec matrice d'adjacence suivante:

```
Const N=100
Type
Sommet= Enregistrements
            Valeur : caractère
            Indice : entier
            Marque : booléen
            Père : entier
        FinEnregistrement

Graphe =  Enregistrements
            Sommets : Tableau [1..N] de Sommet
            MAdj : Tableau [1..N, 1..N] d'entiers
        FinEnregistrement
```

Ecrivez une fonction qui calcule le nombre cyclomatique d'un graphe G. Le nombre cyclomatique $\mu(G)$ pour un graphe G d'ordre n avec m arcs et p composantes connexes est $\mu(G) = m - n + p$.

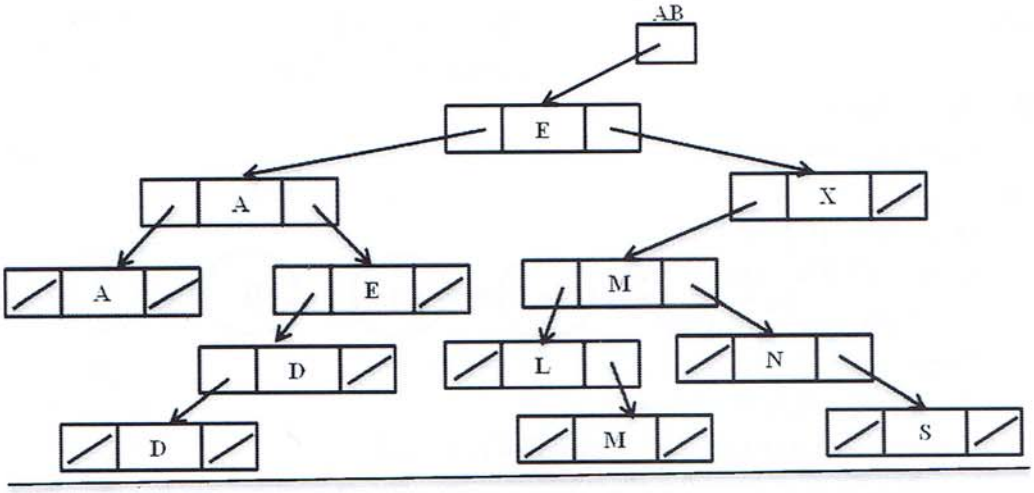
Bonne chance.

Correction de l'Examen d'algorithmique et structures de données

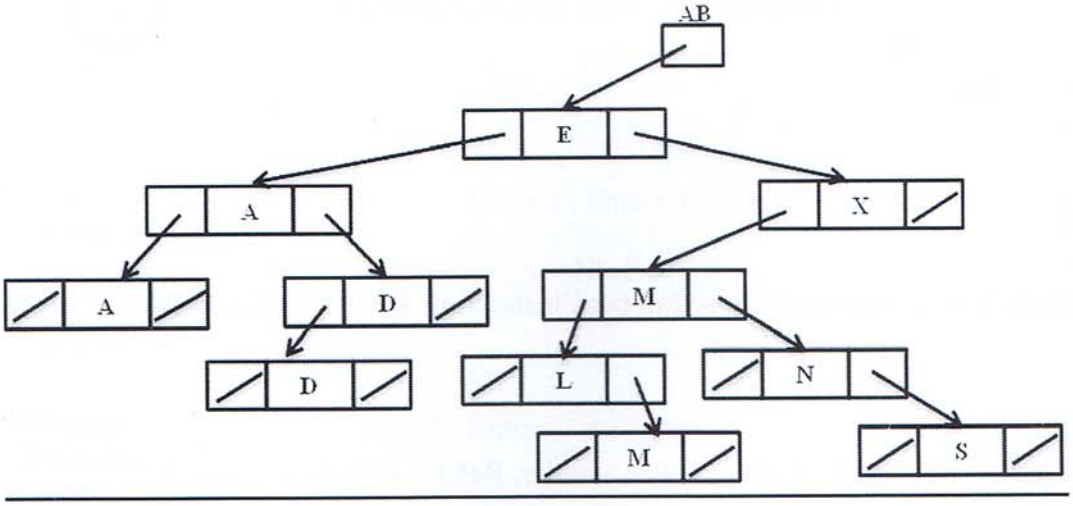
Exercice 1 (5 points):

I1: (0.5 pt)

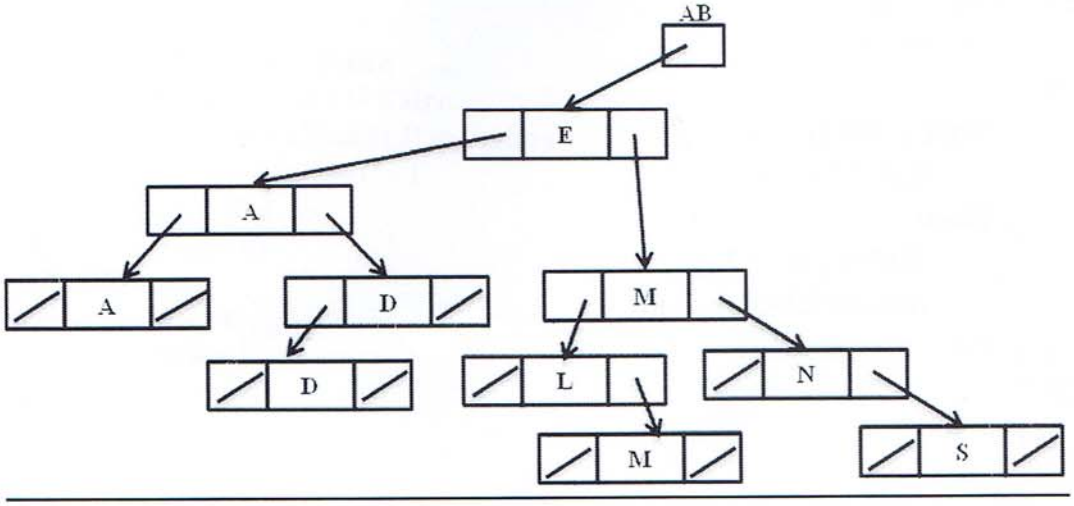
I2: (3 pts)



I3: (0.75 pt)



I4: (0.75 pt)



Exercice 2 (7 points):

1. **2.5 pts**

```
Fonction TestExist (Val X : Entier, Val ID : Entier, Val IF : Entier,
                                    Val T : Tableau d'entiers) : Booléen
  Variables
       M : entier
  Début
       Si (ID > IF )  Alors
             TestExist  ←  Faux
        Sinon
             M  ← ID + IF Div 2
             Si (X = T[M])  Alors
                  TestExist  ←  Vrai
             Sinon
                  Si (X > T[M])  Alors
                        TestExist  ←  TestExist(X,M+1,IF,T)
                  Sinon
                        TestExist  ←  TestExist(X,ID,M-1,T)
                  Fsi
             Fsi
        Fsi
    Fin
FinFonction
```

Le premier appel de la fonction TestExist se fait par l'instruction **TestExist(X,1,N,Tab)**.

2. **2.5 Pts**

```
Procédure AjouterTab (Val X : Entier, Val IF : Entier, Ref T : Tableau d'entiers)
   Début
       Si (IF = 0)  Alors
             T[1] ←  X
        Sinon
             Si (X ≥ T[IF])  Alors
                  T[IF+1] ←  X
             Sinon
                  T[IF+1] ←  T[IF]
                  AjouterTab(X,IF-1,T)
             Fsi
        Fsi
     Fin
FinFonction
```

Le premier appel de la fonction AjouterTab se fait par l'instruction **AjouterTab(X,N,Tab)**.

Pts

- Complexité temporelle de la fonction TestExist: **1 Pts**

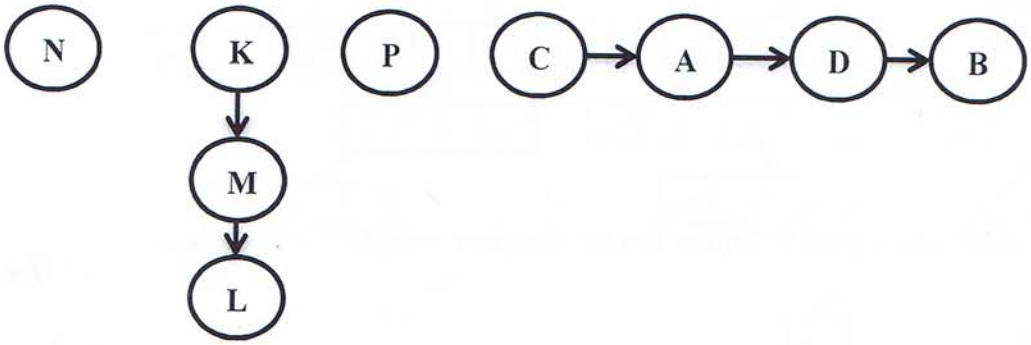
$$T(N)=\begin{cases}\theta(1) & Si\ ID > IF\ ou\ T[M] = X \\ T(N/2) + \theta(1) & Sinon\end{cases}$$

- Complexité temporelle de la fonction AjouterTab: **1 Pts**

$$T(N)=\begin{cases}\theta(1) & Si\ \ N = 0\ ou\ X \geq T[IF] \\ T(N-1) + \theta(1) & sinon\end{cases}$$

Exercice 3 (8 points):

1. **2 pts**



2. **6 pts**

```
Fonction NbrCyclomatiqe (Val G : Graphe) : Entier  0.75 pts
  Début
        NbrCyclomatiqe ← Taille(G) - N + NbrCC(G)
  Fin
FinFonction

Fonction Taille (Val G : Graphe) : Entier     1.50 Pts
 Variables
   C,I,J : Entier
 Début
       C ← 0
       Pour I ← 1 à N Faire
           Pour J ← 1 à N Faire
                 Si G.Madj[I,J] = 1 Alors
                     C ← C +1
                 Fsi
           Finpour
       Finpour
       Taille ← C
   Fin
FinFonction
```

```
Fonction NbrCC (Ref G : Graphe) : Entier     1.75 Pts
  Variables
    NCC,I : Entier
  Début
       NCC ← 0
       Pour I ← 1 à N Faire
            G.Sommets[I].Marque ← Faux
       Finpour
       Pour I ← 1 à N Faire
            Si G.Sommets[I].Marque = Faux Alors
                  NCC ← NCC +1
                  CompConnexe(I,G)
             Fsi
       Finpour
       NbrCC ← NCC
  Fin
FinFonction

Procédure CompConnexe (Val I : Entier, Ref G : Graphe)     2 pts
  Variables
     J : Entier
  Début
       G.Sommets[I].Marque ← Vrai
       Pour J ← 1 à N Faire
            Si (G.Madj[I,J] = 1) ou (G.Madj[J,I] = 1) Alors
               Si G.Sommets[J].Marque = Faux Alors
                    CompConnexe (J,G)
                Fsi
            Fsi
        Finpour
     Fin
FinProcédure
```