Licence Sciences et Technologies

Année 2003 – 2004 Session de Mai 2004
GU : INFO2, MATH2, LICENCE, UE : INF 102 ALGO
GU : MIAS41, UE : 411 MI ALGO

Épreuve d'Initiation à l'Algorithmique-Corrigé

Date : 12 Mai 2004 Durée : 1h30

Les algorithmes donnés dans la correction ne sont pas les seuls possibles. Si vous avez écrit des algorithmes différents, vous n'avez pas forcément tort. Ils doivent simplement être clairs et cohérents.

Épreuve de Mme Delest

Indiquez votre code d'**anonymat** : | No : |

La notation tiendra compte de la clarté de l'écriture des réponses

Barème

- Question 1 - Connaissances générales : 4 points
- Question 2 - Compréhension d'algorithmes de tri : 3 points
- Question 3 - Manipulation de tableaux : 6 Points
- Question 4 - Listes : 3 Points
- Question 5 - Arbres : 2,5 Points
- Question 6 - Table de hachage : 1,5 Points

Question 1. Cocher les affirmations qui sont correctes :

- ☐ Dans une boucle **Pour** le test s'effectue en fin de boucle
- ☒ Les instructions interne d'une boucle **Tant que** peuvent ne pas être exécutées
- ☒ Un tableau est une table d'association à clé unique
- ☐ On peut accéder directement au $k^{ième}$ élément d'une liste
- ☒ Il existe un algorithme de tri qui s'exécute toujours en temps O(n). Le tri par dénombrement
- ☒ La dichotomie est une méthode que l'on peut implémenter avec une fonction récursive
- ☒ Le nombre d'arêtes d'un arbre est égal au nombre de sommets auquel on retranche 1
- ☐ Une table de hachage est une table d'association à clé unique telle que le nombre d'éléments de la table est variable

Question 2. Cet exercice est une variante des algorithmes de tris. *Soit la fonction suivante*

```
fonction test(ref T:tableau[1..N] d'entiers):entier;
  var i,j,c: entier;
  début
    j=0;
    c=0;
    pour i allant de 1 à N faire
      si estPair(T[i]) alors
        echanger(T[i],T[j+1]);
        c=c+1;
        j=j+1;
      finsi
    finpour
  retourner c;
  fin
finfonction
```

La fonction estPair est une fonction Booléenne dont le résultat est vrai si le paramètre est un nombre pair.
On suppose que l'appel s'effectue avec un tableau W de taille $N = 12$ tel que $\forall i \in [1..N], W[i] = i$.

1. *Quel est le résultat de la fonction ? Y a-t-il des effets de bord ? Justifier.* **La justification consistant à éxécuter l'algorithme est correcte**
   j et c ont la même valeur à chaque itération. La valeur initiale de j est 0 et j est incrémenté chaque fois qu'une valeur du tableau est paire. Donc, j compte le nombre de valeurs paires et $j \leqslant i$. A chaque tour de boucle, pour i fixé, si la valeur $T[i]$ est paire, on échange la valeur d'emplacement i avec la valeur d'emplacement $j+1$. Or j compte le nombre de valeur paire par suite $T[j+1]$ est impaire. Ainsi à chaque itération, on place les valeurs paires en tête. Donc, la valeur retournée par la fonction pour le tableau W sera 6. Comme le tableau est passé par référence, il y a un effet de bord qui trie le tableau W valeurs paires puis valeur impaires. Après cette fonction on a : $W = [2, 4, 6, 8, 10, 12, 7, 1, 9, 5, 11, 3]$ (valeur de W non demandée).
2. *On remplace* `ref` *par* `val` *dans l'en-tête de la fonction. Le résultat est-il changé ?* : Non la valeur de c reste la même.
   Y a-t-il des effets de bord ? Justifier. : Non, puisque W est passé par valeur, la fonction travaille sur une copie du tableau W donc la valeur de W ne sera pas modifiée.

Question 3. Cet exercice est une variante des exercices posés en devoir surveillé et des algorithmes de fusion de tableaux *On prendra soin dans ce qui suit d'écrire soigneusement l'en-tête de la fonction et notamment de préciser les mots clés* **ref** *et* **val***. Soit deux tableaux $T1$ de taille $N1$ et $T2$ de taille $N2$.*

1. *Ecrire une fonction qui teste si les deux tableaux sont identiques.*

```
fonction testeEgalité(ref T1 :tableau[1..N1] d'éléments;
                      ref T2 :tableau[1..N2] d'éléments) :booléen;
  var i :entier;
  début
    si N1<> N2 alors
      retourner(faux);
    sinon
      pour i allant de 1 à N1 faire
        si T1[i]<>T2[i] alors
          retourner(faux);
        finsi;
      finpour;
      retourner(vrai);
    finsi;
  fin
finfonction
```

2. *Donnez la complexité de votre algorithme. Réponse :* $C^{<}(n) = O(1)$ et $\overline{C}(n) = C^{>}(n) = O(n)$
3. *Ecrire une fonction qui calcule en au plus $max(N1, N2)$ itérations l'élément le plus petit et l'élément le plus grand parmi l'ensemble des valeurs présentes dans les deux tableaux.*
   Il était suffisant d'écrire les algorithmes sur des éléments entiers.
   Soit MAX (resp. MIN) la valeur maximale (resp.minimale) possible des éléments.

```
fonction minMax(ref T1 :tableau[1..N1] d'éléments;
                ref T2 :tableau[1..N2] d'éléments;
                ref min,max :éléments) :vide;
  var i :entier;
  fonction minMaxElement(ref U1 :tableau[1..NU1] d'éléments;
                         ref U2 :tableau[1..NU2]d'éléments) :vide;
    début
      si U1[i]<min alors
        min=U1[i];
      finsi;
      si U2[i]>max alors
        max=U2[i];
      finsi
    fin
  fonction minMaxUn(ref U :tableau[1..NU] d'éléments) :vide;
    début
      tant que i<=NU faire
        si U[i]<min alors
          min=U[i];
        sinon
          si U[i]>max alors
            max=U[i];
          finsi;
        finsi;
        i=i+1;
      fintantque;
    fin;
  début
    i=1;
    min=MAX;max=MIN;
    tant que i<=N1 et i<=N2 faire
      si T1[i]<T2[i] alors
        minMaxElement(T1,T2);
      sinon
        minMaxElement(T2,T1);
      finsi;
      i=i+1;
    fintantque;
    minMaxUn(T1);
    minMaxUn(T2);
  fin;
finfonction
```

Question 4.

1. *Quelle différence faites vous entre les listes simplement chainées et les listes doublement chainées ?* Les listes simplement chainées permettent uniquement l'accès en $O(1)$ à l'élément de tête de liste et au suivant d'un élément. Les listes doublement chainées ont la même fonctionalité mais de plus on peut accéder en $O(1)$ à la queue de la liste et à l'élément précédent.
2. *Ecrire une fonction qui prend en paramètre un tableau d'entiers et dont le résultat est une liste simplement chainée contenant les éléments du tableau dans l'ordre croissant de l'indice.*

```
fonction tableauListe(ref T:tableau[1..N] d'entiers):listeSC;
  var L: listeSC d'entiers;
  var P:^entier;
  var i:entier;
  début
    creerListeSC(L);
    insererEnTeteP(T[1],L);
    P=premier(L);
    pour i allant de 2 à N faire
      insérerAprès(T[i],L,P);
      P=suivant(L,P);
    finpour;
  fin;
finfonction
```

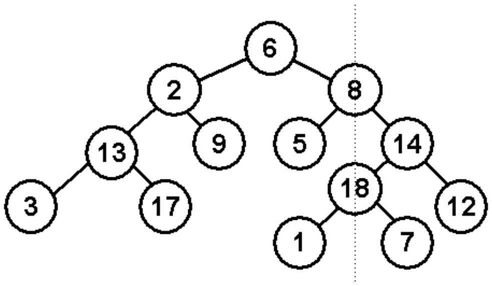

FIG. 1 – Figure question 5.

Question 5.

1. *Pour l'arbre binaire dessiné Figure 1 , répondre aux questions suivantes.*
   - *Donnez l'étiquette du fils gauche du sommet d'étiquette* 14. *Réponse :* 18.
   - *Donnez la hauteur de l'arbre. Réponse :* 4.
2. *Dessinez un arbre planaire non binaire ayant* 7 *sommets.* Réponse Figure 2.

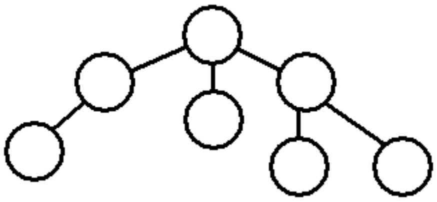

FIG. 2 – Une réponse possible à l'exercice 5 Question 2.

3. *Dessinez tous les arbres binaires ayant 7 sommets.* Réponse Figure 3.

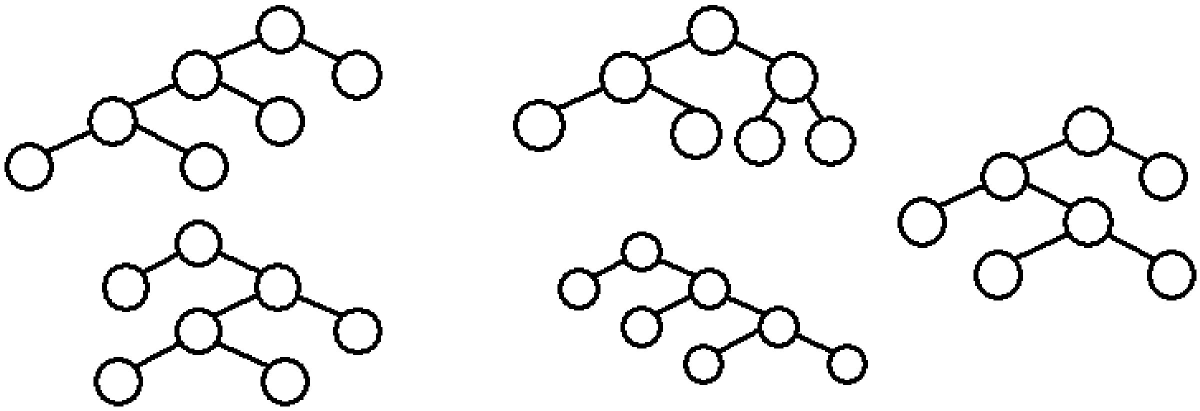

FIG. 3 – Réponse Exercice 5 question 3.

Question 6. O*n considère la fonction de hachage qui à tout entier naturel n associe* $h_k(n) = n[k]$.

1. *Dans le cas de l'adressage ouvert, quel sera la taille de la table de hachage pour cette fonction.* *Réponse :* k.

2. *Soit $k = 11$. On choisit un sondage linéaire pour la gestion des collisions. Donnez l'état de la table après l'insertion de la suite de valeur* $2, 12, 22, 5, 23$.

Indice	0	1	2	3	4	5	6	7	8	9	10
Contenu	22	12	2	23	NULL	5	NULL	NULL	NULL	NULL	NULL

FIN.