Algorithmique
Correction Contrôle n° 1

INFO-SPÉ – EPITA

9 nov 2012 - 10 :00

Solution 1 **(hachages – *7 points*)**

1. Pour les valeurs d'exemple suivantes, en appliquant la fonction $h(x) = x \ mod \ m$, nous obtenons les valeurs de hachage *Th* suivantes :

Elément	15	24	125	4	26	6	78	55	89
Valeur de hachage	3	0	5	4	2	6	6	7	5

2. *Hachage coalescent (voir figure 1) (liens : -1 pour case vide et M pour absence de collision).*

	Th	
0	24	12
1		-1
2	26	12
3	15	12
4	4	12
5	125	10
6	6	11
7	55	12
8		-1
9		-1
10	89	12
11	78	12

FIGURE 1 – Hachage coalescent.

3. Déclaration des types nécessaires à **Th** :

```
contantes
   m=12

types
   t_element = ...
   t_pelt = ^t_elt
   t_elt = enregistrement
      t_element elt
      t_pelt = suiv
   fin enregistrement t_elt
   t_table = m t_pelt
   t_ptable = ^table
   t_hachage = enregistrement
      entier m
      t_ptable table
   fin enregistrement t_hachage

variables
   t_hachage th
```

4. Algorithme de la fonction `estpresent(th,x)` :

```
algorithme fonction estpresent : Booléen
paramètres locaux
   t_hachage th
   t_element x
variables
   Entier v
   t_pelt p
debut
   v ← h(x)               /* calcul de la valeur de hachage primaire */
   p ← th.table^[v]       /* adresse du 1er élément de valeur de hachage égale à v */

   tant que p<>NUL & p^.elt<>x faire   /* boucle de recherche dans la liste chaînée */
      p ← p^.suiv
   fin tant que

   retourne p<>NUL                     /* retour de résultat */
fin algorithme fonction estpresent
```

Solution 2 **(Arbres 2.3.4 : Recherche d'un élément – *6 points*)**

Spécifications :

La fonction `recherche234 (t_element` x`, t_a234` A`)` retourne un pointeur vers le nœud contenant la valeur x dans l'arbre A ou la valeur NUL si x n'est pas présent dans l'arbre.

Principe :

Si l'arbre est vide, la recherche est négative.
Sinon :

- On parcourt la liste des clés jusqu'à la position i telle que : soit $i >$ nombre de clés, soit la clé n° i est la première $\geq$ à celle recherchée.
- Si la clé a été trouvée, la recherche est positive, sinon on relance (on recommence) la recherche sur le fils n° i.

1. *Version récursive :*

```
algorithme fonction recherche234 : t_a234
    parametres locaux
        t_element    x
        t_a234       A

    variables
        entier    i
debut
    si A = NUL alors
        retourne NUL
    sinon
        i ← 1
        tant que (i <= A↑.nbcles) et (x > A↑.cles[i]) faire
            i ← i+1
        fin tant que
        si (i <= A↑.nbcles) et (x = A↑.cles[i]) alors
            retourne A
        sinon
            retourne recherche (x, A↑.fils[i])
        fin si
    fin si
fin algorithme fonction recherche234
```

2. *Version itérative :*

```
algorithme fonction recherche234_iter : t_a234
    parametres locaux
        t_element    x
        t_a234       A

    variables
        entier    i
debut
    tant que A <> NUL faire
        i ← 1
        tant que (i <= A↑.nbcles) et (x > A↑.cles[i]) faire
            i ← i+1
        fin tant que
        si (i <= A↑.nbcles) et (x = A↑.cles[i]) alors
            retourne A
        sinon
            A ← A↑.fils[i]
        fin si
    fin tant que
    retourne NUL
fin algorithme fonction recherche234_iter
```

Solution 3 **Arité moyenne d'un arbre général – *7 points***

```
algorithme procedure rec_arite_nuplet
  parametres locaux
    t_nuplet          A
  parametres globaux
    entier            nbnoeud, nbfils
  variables
    entier            i
debut
  si (A↑.nbFils <> 0) alors
    nbnoeud ← (nbnoeud + 1)
    nbfils ← (nbfils + A↑.nbFils)
    pour i ← 1 jusqu'a A↑.nbFils faire
      rec_arite_nuplet(A↑.fils[i], nbnoeud, nbfils)
    fin pour
  fin si
fin algorithme procedure rec_arite_nuplet
```

```
algorithme procedure rec_arite_dyn
  parametres locaux
    t_arbre_dyn       A
  parametres globaux
    entier            nbnoeud, nbfils
  variables
    entier            i
debut
  si (A↑.fils <> NUL) alors
    nbnoeud ← (nbnoeud + 1)
    A ← A↑.fils
    tant que (A <> NUL) faire
      rec_arite_dyn(A, nbnoeud, nbfils)
      nbfils ← (nbfils + 1)
      A ← A↑.frere
    fin tant que
  fin si
fin algorithme procedure rec_arite_dyn
```