Examen

Mardi 19/01/2021 — Durée : 2h — Documents autorisés

Exercice 1. Arbres binaires
On utilise les types suivants pour un arbre binaire

```
typedef struct node {
    int value;
    struct node *left, *right;
} Node, *Tree;
```

1. Écrire une fonction calculant le nombre de nœuds internes d'un arbre binaire.
2. Quelle est la complexité de cette fonction ?

Exercice 2. File d'attente
On considère le type suivant de listes simplement chaînées permettant d'implémenter une file d'attente FIFO contenant des arbres binaires de l'exercice 1.

```
typedef struct cellule {
    Tree elem;
    struct cellule* succ;
} Cellule;
typedef Cellule* Liste;
typedef struct {
    Liste debut;
    Liste fin;
} File;
```

Le pointeur `debut` pointe sur la première cellule de la liste et le pointeur `fin` pointe sur la dernière cellule de la liste.

1. Écrire une fonction `int empty(File f)`, qui teste si la file est vide.
2. Écrire une fonction `void append(Tree t, File* f)` qui ajoute un élément en fin de file. On écrira une fonction `cons` pour cela.
3. Écrire une fonction `Tree pop(File* f)` qui retire et renvoie l'élément en début de file.

Exercice 3. Ordre hiérarchique
On désire afficher les nœuds d'un arbre binaire en ordre hiérarchique : d'abord la racine, puis les nœuds à distance 1 de la racine de gauche à droite, puis les nœuds à distance 2 de la racine de gauche à droite, etc. Écrire une fonction `void hierarchique(Tree t)` qui affiche les nœuds de l'arbre binaire `t` en ordre hiérarchique. On utilisera une file d'attente FIFO. Quelle est la complexité de cette fonction ?

Exercice 4. Codage de Huffman

1. Construire un arbre de Huffman pour le texte "this is an example of a huffman tree". Dessiner l'arbre de Huffman.
2. Indiquer le codage du texte.

Exercice 5. Hachage

1. On considère une table de hachage de taille $B = 9$ où sont insérées des clés par hachage ouvert. Les clés sont des entiers, les éléments associés sont des chaines de caractères et la fonction de hachage est

```
int h(int x) {
   return x % B;
}
```

Indiquer le contenu de la table de hachage ouvert après les opérations suivantes. On indiquera les cases vides et supprimées.

Ajouter(317, David)
Ajouter(316, Marion)
Ajouter(396, Alex)
Ajouter(653, Morgan)
Ajouter(212, Pierre)
Enlever(317)
Enlever(653)
Ajouter(109, Paul)