DESS-ISC
Université de Bordeaux I
Année 2004-2005

Mardi 4 janvier 2005

ASD
Durée de l'épreuve : 1 heure 30
Documents non autorisés
Une attention toute particulière sera portée à la présentation et à la rédaction de la copie

1 Tableau

On considère un tableau T contenant n éléments avec $T[i] \in \{0, 1\}$ (remarquer que les indices commencent à 1). Ecrire un algorithme qui retourne la position i dans le tableau telle que $T[i]$ est le début de la plus longue suite consécutive de zéros. On expliquera les différentes étapes de l'algorithme proposé et on indiquera sa compléxité en temps.

2 Parcours d'arbres

1. On effectue un parcours préfixé d'un arbre binaire donné en notant les informations contenues en chaque sommet ($A, B, \ldots$), puis on effectue un parcours infixé en notant encore les informations. On obtient les deux listes suivantes :
   - ABDJGHIFCLMEK
   - JDGHIBFALCEMK
   Dessiner cet arbre et écrire en chaque sommet la lettre correspondante.
2. On suppose qu'on donne la liste des sommets d'un arbre binaire obtenue en effectuant un parcours préfixé et la liste des sommets obtenue en effectuant un parcours infixé. Indiquer pourquoi on peut alors reconstruire l'arbre.
3. Peut-on de même reconstruire l'arbre binaire si on a les listes des sommets obtenues en effectuant un parcours préfixé et un parcours postfixé ?
4. Que peut-on dire dans le cas où on a les listes des sommets obtenues en effectuant un parcours infixé et un parcours postfixé ?

3 Algorithme Inconnu

On considère l'algorithme *Inconnu* suivant qui prend en argument un arbre binaire donné par sa racine (racine est une variable de type pointeur contenant l'adresse du sommet racine de l'arbre).

```
Inconnu(racine) ; {
CréerFile(F) ;
Enfiler(F,racine) ;
tant que (non FileVide(F))
   faire { a ← ValeurFile(F) ;
      Défiler(F) ;
      si a≠ nil
         alors { traiter(a) ;
            Enfiler(F,gauche[a]) ;
            Enfiler(F,droit[a]) ;
            }
         }

}
```

Les primitives CréerFile, Enfiler, Défiler, ValeurFile, FileVide permettent respectivement : de créer une file vide, d'ajouter un élément en queue d'une file, de supprimer l'élément en tête de file, d'obtenir la valeur de l'élément en tête d'une file, de tester si une file est vide.

1. On suppose que la fonction *traiter(a)* affiche l'étiquette du sommet pointé par a (i.e. affiche la valeur de info[a]). On considère l'arbre suivant (figure 2) sur lequel on fait tourner l'algorithme *Inconnu*. Donner l'état de la file à chaque itération et indiquer dans quel ordre sont affichées les étiquettes des sommets de cet arbre.

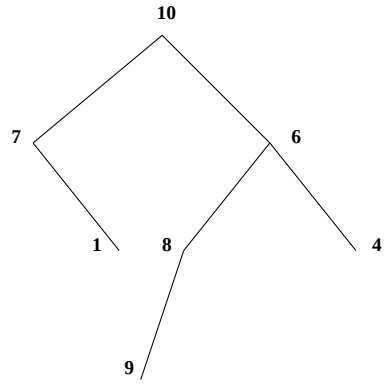

FIG. 1 – Algorithme *Inconnu*

2. Indiquer, de façon générale, l'ordre dans lequel les sommets d'un arbre binaire sont traités par l'algorithme *Inconnu*.
3. Ecrire la fonction *traiter* de telle sorte que seules les étiquettes des feuilles de l'arbre soient affichées.

4 Arbre Binaire de Recherche

1. Rappeler la définition d'un arbre binaire de recherche.
2. On suppose que les entiers compris entre 1 et 600 sont disposés dans un arbre binaire de recherche, et on souhaite retrouver 250. Parmi les séquences suivantes, lesquelles ne pourraient pas être la séquence de nœuds parcourus :
   a) 6,100,390,200,260,248,250
   b) 452,300,330,325,310,250
   c) 600,532, 100,310, 300,200,250
   d) 50,450,390,100,280,295,250