ASI3 Mardi 31 Janvier 2006

Algorithmique et Base de la programmation

Durée : *3h00*
Documents autorisés : **AUCUN**

Remarques :
- Veuillez lire attentivement les questions avant de répondre.
- Le barème donné est un barème indicatif qui pourra évoluer lors de la correction.
- Rendez une copie propre.

1 Tri rapide (3 points)

1.1 Principe

Veuillez rappeler le principe de fonctionnnement du partitionnement dans le tri rapide en vous basant sur l'exemple suivant :

4	1	2	8	5	4	3	8	9

1.2 Algorithme

Veuillez donner le corps de la procédure suivante :

procédure partitionner **(E/S** t : **Tableau**[1..MAX] **d'Entier** , **E** d,f :**Naturel** , **S** indicePivot : **Naturel)**

1.3 Complexité

Démontrez que le compléxité du tri rapide dans le pire des cas est de n^2

2 Le nombre de Strahler[1] (4 points)

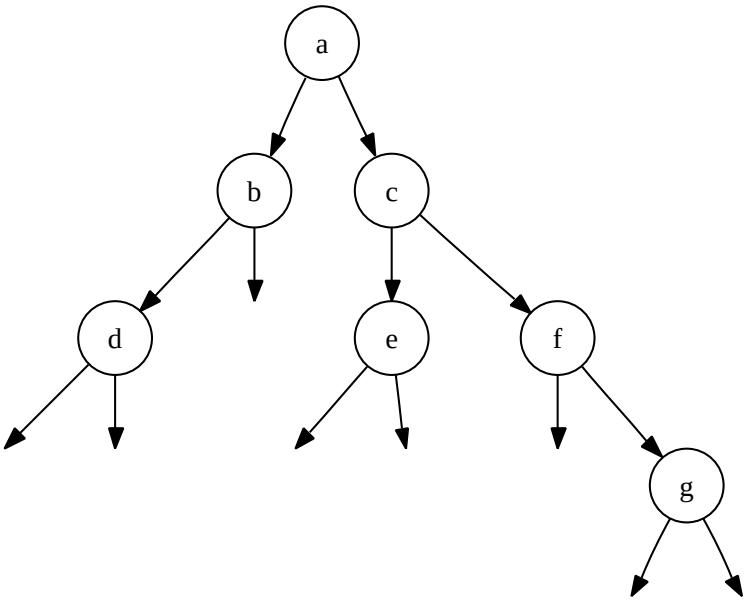
Le nombre de Strahler d'un arbre binaire est défini par :

$$\Phi(A) = \begin{cases} 0 & \text{si l'arbre } A \text{ est vide} \\ max(\Phi(A_g), \Phi(A_d)) & \text{si } A \text{ est non vide et } \Phi(A_g) \neq \Phi(A_d) \\ 1 + \Phi(A_d) & \text{si } A \text{ est non vide et } \Phi(A_g) = \Phi(A_d) \end{cases}$$

1. Donnez le TAD ArbreBinaire vu en cours

[1]exercice issu d'un examen d'IUP GMI de la Faculté des Sciences de l'Université de Rouen

2. Donnez les valeurs de Φ pour chaque nœud de l'arbre suivant :



3. Donnez un algorithme permettant de calculer $\Phi(A)$

3 Un peu de C[2] (3 points)

Identifiez les 6 erreurs se trouvant dans le programme ci-dessous. Indiquer brièvement la façon de les corriger.

```
1  #include"stdio.h"
2
3  int main() {
4    char * chaine, motDePasse="Mot2Passe";
5    scanf("%s",chaine);
6    if (chaine == motDePasse)
7      printf("Vous pouvez passer.\n")
8      return 0;
9    else
10     printf("C'est pas ca\n");
11     return 1;
12 }
```

[2]http ://www.enseirb.fr/ pelegrin/enseignement/enseirb/prog_c/examens/

4 Comment retrouver son chemin ? (10 points)

L'objectif de cet exercice est d'étudier le problème du labyrinthe. Comme l'indique la figure 1, l'objectif est de trouver un algorithme permettant de trouver le chemin qui mène de l'entrée à la sortie.

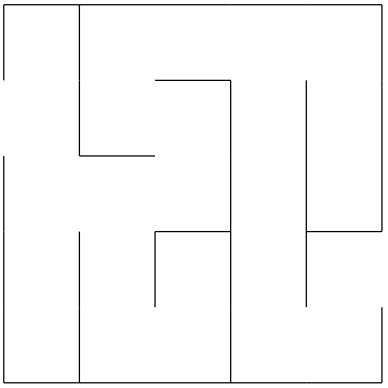


FIG. 1 – Un labytinthe

4.1 Partie publique

En fait, un labyrinthe est composé de cases. On accède à une case à partir d'une case et d'une direction. Les directions possibles sont `Nord`, `Sud`, `Est` et `Ouest`.

Par exemple, comme le montre la figure 2 le labyrinthe précédent peut être considéré comme étant composé de 25 cases. La case numéro 6 est la case d'entrée. La case 20 est la case de sortie. La case 8 est accessible depuis la case 13 avec la direction `Nord`.

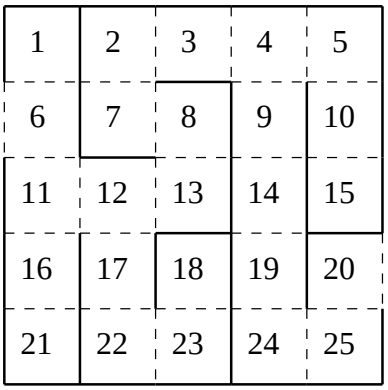


FIG. 2 – Un labytinthe composé de cases

4.1.1 Le TAD labyrinthe

Les opérations disponibles sur un labyrinthe sont les suivantes :
- créer un labyrinthe,
- obtenir la case d'entrée,

– savoir si une case est la case de sortie,
– obtenir une liste de directions possibles depuis une case donnée,
– obtenir la case accessible depuis une case avec une direction.

1. Donnez le type `Direction`
2. Donnez les TAD `Labyrinthe` et `Case` (TAD liés donc définis en même temps comme nous l'avons vu en TD)

4.1.2 Algorithme du petit-poucet

Une solution pour trouver la sortie est d'utiliser le principe du petit poucet, c'est-à-dire mettre un caillou sur les cases rencontrées.

Pour ne pas modifier le TAD Labyrinthe, plutôt que de marquer une case avec un caillou on peut ajouter une case à un ensemble. Pour vérifier si on a déja rencontré une case, il suffit alors de vérifier si la case est présente dans l'ensemble.

Après avoir rappelé les TAD Ensemble et ListeChaînée vu en cours, proposer le corps de la procédure suivante qui permet de trouver le chemin de sortie (s'il existe) à partir d'une case donnée[3] :
procédure calculerCheminDeSortie (**E** l : Labyrinthe, caseCourante : Case , **E/S** casesVisitees : Ensemble<Case> , **S** permetDAllerJusquALaSortie : **Booléen**, lesDirectionsASuivre : ListeChaînée<Direction>)

4.2 Partie privée

4.2.1 Le graphe

On peut représenter un labyrinthe à l'aide d'un graphe étiqueté et valué. On considère dans ce cas que les valeurs des nœuds du graphe sont les cases du labyrinthe et les arcs étiquetés par les directions.

Dessinez le graphe associé à l'exemple de la figure 3.

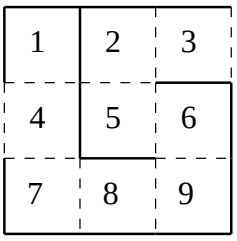


Fig. 3 – Un labytinthe composé de 9 cases

4.2.2 Représentation du graphe

Proposez la matrice d'adjascence du graphe précédent.

[3]Vous pouvez vous inspirer de la procédure *remplir* vu dans le dernier cours