Algorithmique (LI 043)
DST n° 2 & Examen final
Durée : 2 heures
Aucun document autorisé

1. On se propose d'implémenter une liste *doublement* chaînée (chaque cellule connaît la cellule suivante, et la cellule précédente), en utilisant des tableaux (méthode du « curseur »), de telle manière que, par exemple, la liste {3,1,4,1,5,9,2} soit représentée ainsi :

tête [4]

queue [6]

libre [5]

	val	prec	suiv
1	4	9	2
2	1	1	3
3	5	2	10
4	3	0	9
5			7
6	2	10	0
7			8
8			0
9	1	4	1
10	9	3	6

   (a) On définit dans un premier temps un jeu de primitives de "bas niveau" :

primitive	*argument(s)*	*résultat*
`suivant`	cellule	cellule
`precedent`	cellule	cellule
`pos2cel`	entier (positions)	cellule
`cel2pos`	cellule	entier (position)

   Proposer une implémentation de ces primitives.

   (b) On se propose maintenant de définir de la façon la plus efficace possible les primitives habituelles de manipulation de liste element, ajouter, supprimer) en utilisant les primitives de bas niveau précédemment définies.

primitive	*argument(s)*	*résultat*
`tete`		cellule
`element`	cellule	valeur
`ajouter`	valeur ; entier (position)	
`supprimer`	entier (position)	liste

   Proposer une implémentation de ces primitives

   (c) Quelle est la complexité (moyenne) des opérations que vous avez définies ?

   (d) Proposer un algorithme qui utilise ces primitives pour compter le nombre d'occurrences d'une valeur dans une liste.

2. En utilisant l'algorithme de parcours de votre choix (et en supposant définies les primitives nécessaires), proposer un algorithme qui calcule la longueur *moyenne* des branches d'un arbre quelconque donné.