TD 3 - Les listes et la complexité- Master 1 Bio-informatique

Décembre 2016

Exercice 1

Ecrivez un programme python qui permet de créer des listes simplement chaînés.
Pour cela, vous implémenterez l'api suivante :

```
def push_front(l, e):
  # Ajoute un élément en debut de list
def first( l ):
  # renvoie la première cellule de la liste
def end( l ):
  # renvoie la cellule de fin de liste
def next( l, cell ):
   # renvoie la cellule qi suit cell et None si c'est la cellule de fin
   # de list
def value( cell ):
   # renvoie la valeure de la liste
```

Exercice 2

A laide de l'implémentation de la liste chainée précédente, écrivez les fonctions suivantes :

```
def push_back(l, e):
  # Ajoute un élément en fin de liste
def insert( l, cell, e ):
  # Créé une nouvelle cellule contenant l'élément e et insère cette
  # cellule dans la liste juste après la cellule cell.
def copy( l ):
  # renvoie une copie de la liste l
def milieu(l):
  # renvoie l'élément situé en milieu de liste
def taille(l):
  # renvoie le nombre d'éléments de la liste
def transferer( l1, l2 ):
  # Transfère les éléments de l2 dans l1. La liste l2 devient vide.
def recherche_element(l,e):
  # renvoie la première cellule contenant e ou la cellule de fin de liste.
```

Donnez la compléxité de chacune de ces fonctions.

Exercice 3

Reprenez les deux exercices précédent en utilisant des listes doublements chaînées.

Pour aller plus loin, vous pouvez implémenter l'algorithme des pointeurs dansant de Knuth : `https://arxiv.org/abs/cs/0011047`.