Fonctions récursives sur les listes chaînées

ajouteEnFin

Données: un élément *e* une liste *L*
Résultat: une liste identique à *L*, sauf que *e* a été ajouté en fin
si *estVide ?(L)* **alors**
| **retourner** *cons(e,NULL)*
sinon
| **retourner** *cons(first(L),ajouteEnFin(e, succ(L)))*
fin

dernier

si *L a un seul élément* **alors retourner** *le seul élément de L*;
retourner *dernier(cdr(L));*

ajouteEnFin

Données: un élément *e* une liste *L*
Résultat: une liste identique à *L*, sauf que *e* a été ajouté en fin
si *estVide ?(L)* **alors**
| **retourner** *cons(e,NULL)*
sinon
| **retourner** *cons(first(L),ajouteEnFin(e, succ(L)))*
fin

dernier

si *L a un seul élément* **alors retourner** *le seul élément de L*;
retourner *dernier(cdr(L));*

concatener

si *L1 est vide* **alors retourner** L2;
retourner *cons(first(L1), concatener(succ(L1),L2))*

Un mauvais algorithme, mais de combien ?

concatenerStupide

Données: $L_1,\ L_2$
Résultat: $L = L_1 L_2$
si *EstVide*(L_2) **alors**
  retourner L_1
sinon
  retourner
  ajouteEnFin(*Dernier*(L_2),
    concatenerStupide$(L_1,\ $*ToutSaufDernier*$(L_2)))$
fin

Difficile à analyser

concatenerAnalyseDifficile

Données: $L_1,\ L_2$
Résultat: $L = L_1 L_2$
si *EstVide*(L_2) **alors**
  retourner L_1
sinon
  retourner
  concatenerAnalyseDifficile(*ajouteEnFin*(*first*(L_2),L_1), *succ*(L_2))
fin