IF111 - Algorithmes et structures de données-TD6 : Rappels avancés sur les listes

Jonathan Narboni, Rohan Fossé

jonathan.narboni@labri.fr, rfosse@labri.fr

Dans cette feuille de TD, vous pourrez utiliser les fonctions suivantes :

- $vide(l)$: Renvoie $True$ si la liste l est vide et $False$ sinon;
- $cons(a, l)$: Renvoie la liste obtenue en ajoutant l'élément a devant la liste l;
- $tete(l)$: Renvoie le premier élément de la liste l;
- $queue(l)$ Renvoie la queue de la liste, ie l sans le premier élément.

Exercice 1

Écrivez les fonctions suivantes sur les listes en utilisant les données sur les listes:

1. $plus\_souvent(x, y, l)$ qui renvoie $True$ si l'élément x apparaît plus de fois que l'élément y dans la liste l et $False$ sinon;
2. $millieme(l)$ qui renvoie le millième élément de la liste l (on suppose que la liste l a au moins mille éléments donc il n'est pas nécessaire de traiter le cas d'erreur);
3. $facto\_liste(l)$ qui renvoie la liste contenant la factorielle de chacun des éléments de la liste l (si l contient les valeurs 4, 3 et 2, la fonction doit renvoyer une liste contenant les valeurs 24, 6 et 2) ;
4. $produit(l1, l2)$ qui renvoie la liste contenant les produits des éléments de l_1 et l_2 que l'on suppose être de même longueur (si l_1 contient les entiers 1, 2 et 4 et l_2 contient les entiers 2, 3 et 4, la fonction doit renvoyer une liste contenant les valeurs 2, 6 et 16) ;
5. $double(l)$ qui renvoie True si chaque élément de la liste est supérieur au double de son prédécesseur, et $False$ sinon (par exemple sur la liste 2, 5, 10 et 23 la fonction renverra $True$ mais sur la liste 2, 4, 9, 12 et 25 elle renverra $False$ car 12 est inférieur au double de 9) ;

Exercice 2

1. Écrivez les fonctions suivantes de manière récursive (ici encore en utilisant les fonctions données sur les listes) :
   - $append(a, l)$ qui ajoute l'élément a à la fin de la liste l;
   - $concat(l_1, l_2)$ qui concatène les listes l_1 et l_2 (les éléments de l_1 suivis des éléments de l_2 dans une même liste) ;
   - $reverse(l)$ qui renvoie la liste l retournée (premiers éléments à la fin).

2. Quelles sont les complexités des fonctions $append(a, l)$ et $reverse(l)$ en fonction de la longueur de la liste ?
   On se propose d'améliorer la version récursive de $reverse(l)$ en définissant une fonction $reverse\_concat(l1, l2)$ qui renvoie la concaténation de reverse(l_1) et l_2 (par exemple, sur les entrées 1, 2, 3 et 4, 5, 6, la fonction $reverse\_concat$ doit renvoyer la liste 3, 2, 1, 4, 5, 6).

3. Écrivez directement la fonction $reverse\_concat(l1, l2)$ de manière récursive sans utiliser les fonctions $concat(l_1, l_2)$ et $reverse(l)$.
   Indication: il est recommandé de faire un schéma pour voir ce qui se passe et comprendre pourquoi cette fonction est facile à écrire.

4. En utilisant la fonction $reverse\_concat(l1, l2)$, ré-écrivez la fonction $reverse(l)$ (en deux lignes)