TD OCaml: le noyau fonctionnel

David Delahaye

Faculté des Sciences
David.Delahaye@lirmm.fr

Licence L3 2019-2020

Exercice 1

Listes

1. Écrire une fonction qui inverse les éléments d'une liste ;
2. Écrire une fonction qui donne le nombre d'occurrences d'un élément donné dans une liste ;
3. Écrire une fonction qui teste si une liste est triée ou non ;
4. Écrire une fonction qui réalise une insertion triée d'un élément donné dans une liste triée ;
5. Écrire une fonction qui trie une liste selon l'algorithme de tri par insertion (utiliser la fonction précédente).

Exercice 2

Types concrets

1. Écrire le type concret représentant les formules en logique propositionnelle (pour les variables, prendre au choix soit une chaîne de caractères représentant le nom de la variable, soit un entier numérotant la variable) ;
2. Écrire une fonction qui affiche en syntaxe concrète une formule (rendre une chaîne de caractères) ;
3. Écrire la fonction qui simplifie une formule selon les règles suivantes :
   - $(A \wedge \top) \hookrightarrow A$, $(A \wedge \bot) \hookrightarrow \bot$ (et vice versa) ;
   - $(A \vee \top) \hookrightarrow \top$, $(A \vee \bot) \hookrightarrow A$ (et vice versa) ;
   - $(A \to \top) \hookrightarrow \top$, $(A \to \bot) \hookrightarrow \neg A$;
   - $(\top \to A) \hookrightarrow A$, $(\bot \to A) \hookrightarrow \top$;
   - $(A \leftrightarrow \top) \hookrightarrow A$, $(A \leftrightarrow \bot) \hookrightarrow \neg A$ (et vice versa).

Exercice 2

Types concrets

4. Écrire une fonction qui étant donnée une assignation des variables (qui ont une valeur à `true` ou `false`), évalue une formule.
   Une assignation sera une liste d'association composée de couples de noms de variables et de valeurs booléennes. On pourra utiliser la fonction `List.assoc` qui à un nom de variable donné associe sa valeur booléenne dans la liste (si elle existe).
5. Écrire une fonction qui étant donnée une formule dit si cette formule est une tautologie, c'est-à-dire qu'elle est vraie dans toutes les assignations possibles.
   On utilisera la méthode naïve qui consiste à vérifier la formule dans toutes les assignations possibles.
   On écrira, au préalable, une fonction qui récupère la liste des variables de la formule, en prenant soin de ne pas avoir de doublons.

Exercice 3

Itérateurs sur les listes

Les itérateurs sur les listes sont décrits ici :
https://caml.inria.fr/pub/docs/manual-ocaml/libref/List.html

1. En utilisant `List.fold_right`, écrire une fonction qui fait la somme des éléments d'une liste d'entiers.
2. En utilisant `List.for_all`, écrire une fonction qui teste si tous les éléments d'une liste d'entiers sont positifs.
3. En utilisant `List.filter`, écrire une fonction qui retire tous les éléments négatifs d'une liste d'entiers.
4. En utilisant `List.fold_left`, écrire une fonction qui inverse les éléments d'une liste.
5. En utilisant `List.fold_left`, écrire une fonction ayant le comportement de la fonction `List.map`.

Exercice 4

Arbres binaires de recherche (ABR)

Pour plus d'informations sur les ABR, voir :
https://fr.wikipedia.org/wiki/Arbre_binaire_de_recherche

1. Écrire un type concret représentant les ABR.
2. Écrire la fonction d'insertion d'un élément dans un ABR.
3. Construire des exemples d'ABR en utilisant la fonction d'insertion.
4. Écrire la fonction de recherche d'un élément dans un ABR.
5. Écrire la fonction de suppression d'un élément d'un ABR.