TP 10— Révisions

http://www.liafa.jussieu.fr/~nath/tpcaml/tp10/tp10.pdf

Les questions sont les bienvenues et peuvent être envoyées à nathanael.fijalkow@gmail.com.

Ce dernier TP propose une liste (non exhaustive) des programmes que vous devez savoir écrire rapidement et sans erreurs sous CaML. On n'utilisera pas de références. Pire, l'utilisation de références de listes est passible de $5/2$.

1 Tris et manipulations de listes

▷ **Question 1.** [3 lignes] Écrire une fonction **list_map** qui applique une fonction à chacun des éléments d'une liste et la retourne, sans modifier l'ordre des éléments. En d'autres termes, "écrivez la seule fonction intéressante de type `('a -> 'b) -> 'a list -> 'b list`". ◁

▷ **Question 2.** [3 lignes] Écrire une fonction **list_select** qui retourne la liste des éléments vérifiant un prédicat parmi une liste donnée. ◁

▷ **Question 3.** [3 lignes] Écrire une fonction **insere** qui ajoute un élément dans une liste triée. ◁

▷ **Question 4.** [3 lignes] Écrire une fonction **tri_insertion** qui trie une liste par insertion. ◁

▷ **Question 5.** [9 lignes] Écrire une fonction **tri_rapide** qui trie une liste par tri rapide. ◁

▷ **Question 6.** [5 lignes] Écrire une fonction **list_rev** qui retourne une liste *en temps linéaire*. ◁

2 Entiers et bases

▷ **Question 7.** [4 lignes] Écrire une fonction **puissance**. ◁

▷ **Question 8.** [1 ligne] Écrire une fonction **pgcd** qui calcule le pgcd de deux entiers. ◁

▷ **Question 9.** [3 lignes] Écrire une fonction **base** qui calcule la décomposition d'un entier dans une base donnée. ◁

▷ **Question 10.** [3 lignes] Écrire une fonction **eval_base** qui calcule un entier donné par sa décomposition dans une base. ◁

▷ **Question 11.** [8 lignes] Écrire une fonction **addition** qui additionne deux entiers donnés dans une base commune. ◁

▷ **Question 12.** [4 lignes] Écrire une fonction **fibo** qui calcule la suite de Fibonacci. ◁

3 Polynômes

Un polynôme est donné sous forme de liste de ses coefficients.

▷ **Question 13.** Quelle fonction écrite dans ce TP permet d'évaluer un polynôme en un point ? ◁

▷ **Question 14.** [7 lignes] Écrire une fonction **derive** qui dérive un polynôme. ◁

4 Recherche d'éléments

▷ **Question 15.** [5 lignes] Écrire une fonction **recherche** qui teste la présence d'un élément dans un tableau. ◁

▷ **Question 16.** [8 lignes] Écrire une fonction **recherche_dicho** qui teste la présence d'un élément dans un tableau ordonné. ◁

On considère des arbres binaires donnés par le type arbre.

▷ **Question 17.** [4 lignes] Écrire une fonction **max_min** qui retourne le maximum et le minimum des étiquettes d'un arbre. ◁

```
type arbre = Feuille of int | Noeud of arbre * arbre * int
```

▷ **Question 18.** [6 lignes] Écrire une fonction **abr** qui teste si un arbre est un arbre binaire de recherche. ◁

▷ **Question 19.** [3 lignes] Écrire une fonction **cherche_abr** qui cherche un élément dans un arbre binaire de recherche. ◁

▷ **Question 20.** [3 lignes] Écrire une fonction **max_sum** qui renvoit la valeur maximale de la somme des éléments sur une branche. ◁

▷ **Question 21.** [6 lignes] Écrire une fonction **hierarchie** qui retourne la liste des étiquettes des éléments, pris depuis la racine jusqu'aux feuilles, de gauche à droite. ◁

5 Expressions arithmétiques

On considère des expressions arithmétiques données sous forme d'arbre de type `arbre_expr`.

```
type arbre_expr = Value of int
| Op_bin of arbre_expr * arbre_expr
| Op_un of arbre_expr
```

▷ **Question 22.** [4 lignes] Écrire une fonction **prefixe** qui retourne l'expression arithmétique sous forme préfixe. ◁

▷ **Question 23.** [4 lignes] Écrire une fonction **infixe** qui retourne l'expression arithmétique sous forme infixe. ◁

▷ **Question 24.** [4 lignes] Écrire une fonction **postfixe** qui retourne l'expression arithmétique sous forme postfixe. ◁

6 Graphes

Pour représenter le graphe on utilise les listes d'adjacence : Succ_u est la liste des successeurs (immédiats) de u.

▷ **Question 25.** [7 lignes] Écrire une fonction **accessible** qui retourne l'ensemble des sommets accessibles depuis un sommet donné. ◁