TP 6: Les listes (2 séances)

1 Premières fonctions sur les listes.

Rappel : la liste vide se code [] et sert de cas de base dans la plupart des fonctions récursives sur les listes. Une liste non vide peut se filtrer en **t ::l** où t est l'élément de tête et l le reste de la liste.

Ecrire les fonctions suivantes :

1. ***longueur*** qui renvoie la longueur d'une liste
2. ***n_ème*** qui étant donnés une liste et un entier renvoie le n-ème élément de la liste
3. ***zéros*** qui étant donnés une liste d'entiers et un entier p ajoute p fois 0 en tête de liste.
4. ***aligner*** qui étant donné deux listes, construit un couple de listes de même longueur, la plus courte étant alignée sur la plus longue par ajout de 0 en tête.
5. ***couple*** qui étant donné deux listes, les aligne sur la plus longue puis construit la liste des couples formés par les éléments de même rang dans les listes.

2 Fusion de listes croissantes.

1. Ecrire une fonction booléenne ***est_croissante*** qui indique si les éléments consécutifs d'une liste sont bien ordonnés en croissant.
2. Ecrire une fonction ***fusion*** qui étant donné deux listes supposées croissantes renvoie la fusion de ces deux listes, toujours ordonnée croissante.

3 Représentation de polynômes par des listes.

On propose de représenter un monôme à coefficient entier par un couple comprenant son degré et son coefficient et un polynôme à coefficients entiers par la liste ordonnée selon les puissances décroissantes de ses monômes non nuls.
Ex : Le polynôme $X^{17} - 5X^3 + 8$ est ainsi représenté par la liste [(17,1);(3,-5);(0,8)].
Écrire en utilisant cette représentation, les fonctions récursives :

1. ***derive*** qui retourne le polynôme dérivé.
2. ***valeur*** qui donne la valeur d'un monôme en un point entier.
3. ***pvaleur*** qui calcule la valeur d'un polynôme en un point entier.
4. ***somme*** qui réalise la somme de deux polynômes (attention : un coefficient peut s'annuler).

4 Nombres premiers par le crible d'Eratosthène

On souhaite écrire un programme permettant d'obtenir une liste de couples de nombres premiers jumeaux, c'est-à-dire des couples (a,b) avec a et b premiers et b-a=2. Par exemple (3,5) et (5,7) sont des couples de nombres premiers jumeaux.

Pour cela on commencera par écrire les fonctions suivantes :

1. ***generer*** : qui étant donné un entier n, construit la liste ordonnée des entiers de 2 à n.
2. ***eliminer*** : qui étant donnée une liste et un entier élimine tous les multiples de l'entier dans la liste.
3. ***eratos*** : qui étant donné un nombre entier n construit la liste des nombres premiers inférieurs ou égaux à n. Pour cela on utilisera la méthode dite du crible d'Eratostene : partant de la liste [2;...;n], son premier élément est premier mais on doit éliminer tous les multiples de cet élément qui eux ne le sont pas. On itère ce procédé jusqu'à la fin de la liste.
4. ***jumeaux*** : qui étant donnée une liste de nombres premiers, construit la liste des couples de nombres premiers jumeaux qu'elle contient.
5. Écrire enfin une fonction permettant de donner la liste des couples de nombres premiers jumeaux inférieurs à une limite n fixée.

5 Suite de Fibonacci

Vous connaissez tous la suite de Fibonacci : (1,1,2,3,5,8,13, …) qui commence par 1,1 et dont chaque terme est la somme des deux précédents.

1. Écrivez la fonction ***Fibonacci*** qui associe à l'entier n la liste des n premiers termes de la suite de Fibonacci.
   (*indication : définissez localement une fonction récursive qui au couple d'entiers* (a,b) *associe, dans le cas général, la liste de Fibonacci commençant par* [a;b; …])

   ```
   Fibo 8 ;;
   [1;1;2;3;5;8;13;21]
   ```

2. Écrivez également la fonction ***Fibonacci1*** qui associe à l'entier n le nème terme de la suite

6 Représentation des ensembles par des listes :

On choisit de représenter un ensemble fini par la liste Caml de ses éléments. Cette liste ne contient aucune répétition. Ainsi l'ensemble {1,2,3} est représenté par une liste contenant 1, 2 et 3 dans un ordre quelconque. [1;2;3], [1;3;2], [2;3;1] ou [3;1;2] sont donc des représentations acceptables de {1,2,3}.
Ecrire en Caml les fonctions définissant les opérations ensemblistes suivantes :

1. ***appartient*** qui détermine si un élément appartient à un ensemble,

2. ***cardinal*** qui donne le cardinal d'un ensemble fini,

3. ***union***,
4. ***intersection***,
5. ***inclus*** qui détermine si un premier ensemble est inclus dans un second,

6. ***disjoint*** qui détermine si deux ensembles sont disjoints,

7. ***egaux*** qui détermine si deux ensembles sont égaux,

8. ***complement*** qui calcule le complémentaire d'un premier ensemble dans un second,

9. ***ensemble*** qui à partir d'une liste quelconque construit une liste contenant les mêmes éléments mais privés de leurs redondances,

10. ***parties*** qui construit l'ensemble des parties d'un ensemble

7 Un petit traitement de texte

Découpage d'un texte en ses mots

Dans tout ce qui suit, on appelle texte une chaîne de caractères constituée de mots et de séparateurs. Convenons d'appeler mot toute chaîne formée de lettres ou de chiffres. Tout autre caractère est considéré comme séparateur.

Par exemple le texte " rien n'est jamais acquis à l'homme " est formé de 8 mots.

1) Ecrivez alors la fonction booléenne ***séparateur*** qui indique si un caractère est un séparateur au sens indiqué plus haut (pour les lettres accentuées, on peut tester l'appartenance à "éèêçàùü").
2) Ecrivez la fonction ***avance*** qui, appliquée à un texte retourne ce texte privé des séparateurs placés avant le premier mot.
3) Ecrivez la fonction ***suivant*** qui, appliquée à un texte retourne un couple formé du premier mot de ce texte et de la suite du texte, débarrassée des séparateurs.
   Par ex. **suivant** ";;abc;;def;;h-ij" doit retourner le couple ("abc","def;;h-ij").
   Pour cela, la fonction (non récursive) **suivant** *contiendra une fonction (récursive)* **transvase** *qui appliquée à un couple de chaînes, transvasera les caractères de l'une à l'autre jusqu'à la rencontre d'un séparateur ou de la fin de la chaîne.*
   Par ex. transvase ("ab","cd;;ef") *renvoie le couple* ("abcd",";;ef").
4) Ecrivez la fonction ***plus_long_mot*** qui détermine la longueur d'un mot le plus long d'un texte (commencez par écrire une fonction ***maximum*** qui renvoie le maximum de deux entiers).