Programmation fonctionnelle - TP 1

Les exercices proposés sont volontairement nombreux, de manière à ce que chacun y trouve son compte. Les prendre dans l'ordre.
Pour chaque question, il est demandé d'écrire, avant toute chose, une interface (en suivant le modèle donné en cours).
Chaque fonction sera testée : cas nominaux, cas erronnés. Mettre les tests et leurs résultats en commentaire dans votre fichier OCaml après le code de la fonction. L'exemple de la fonction valeur absolue est donnée ci-dessous. Dans ce cas, on peut explorer les cas de tests suivants : un entier négatif (-5 par exemple), 0 et un entier positif (3 par exemple). En effet la spécification d'une telle fonction (bien connue des mathématiciens) montre bien que le résultat dépend du signe de l'entier. D'autre part, 0 est un cas de test bien légitime car est-il positif ? négatif ? une erreur dans la manipulation des signes de comparaison est si vite arrivée ....

```
(** abs
@param a un entier
@return valeur absolue de a
*)
let abs a = if a <0 then (-a) else a;;

(*  abs(-5), abs 0, abs 3 *)
```

Pour ce premier TP, nous utiliserons la boucle interactive de OCaml (commande ocaml). Ouvrir dans un diteur un fichier tp1.ml qui recevra toutes les dfinitions.
Exercice 1 (**Fonctions simples**)

1. Ecrire une fonction qui teste si son argument est pair. On indique que l'expression $a \ mod \ b$ retourne le reste dans la division entière de a par b.
   Remarque : **qui teste** signifie la plupart du temps **qui retourne true si la condition est vraie et false sinon**.
2. Ecrire une fonction qui retourne -1 si son argument est négatif, 0 si c'est 0 et 1 si l'argument est positif.
3. Ecrire une fonction qui calcule le volume d'une sphère
4. Ecrire une fonction qui prend en argument 3 entiers et retourne le plus grand de ces entiers.
5. Ecrire une fonction qui ajoute de part et d'autre d'une chaîne de caractères quelconque la chaîne "!!" appelée cadre.

   Modifier la fonction de manière àce que le cadre devienne aussi un argument de la fonction (on dit que l'on *abstrait* le cadre).

   On veut maintenant encadrer dissymétriquement. Introduire les paramètres nécessaires et écrire la nouvelle fonction.
6. Ecrire une fonction qui détermine le nombre de solutions réelles d'une équation du second degré (0 quand pas de solution réelle, 1 si racine double, 2 sinon).

Exercice 2 (**Fonctions récursives simples**)

1. Ecrire une fonction récursive `u` telle que `u n` calcule le nième terme de la suite $(u_n)_n$ définie par : $u_0 = 1$ et $u_n = sqrt(u_{n-1} + 2), n > 0$. La fonction *sqrt* de type `float -> float` existe et calcule la racine carrée.

2. Ecrire une fonction récursive `somme` qui calcule la somme des n premiers entiers naturels : $somme\ n = \Sigma_{i=1}^{i=n} i$

3. Ecrire une fonction récursive `carre` qui calcule la somme des n premiers carrés : $carre\ n = \Sigma_{i=1}^{i=n} i^2$

4. Ecrire la fonction `puissance` en utilisant une méthode dichotomique, c'est-à-dire en utilisant les résultats suivants :
   $a^{2k} = (a^2)^k$
   $a^{2k+1} = (a^2)^k\ *\ a$
   On supposera a et n entiers naturels.

5. Reprendre la fonction `puissance` en utilisant les résultats suivants :
   $a^{2k} = (a^k)^2$
   $a^{2k+1} = (a^k)^2\ *\ a$
   On supposera a et n entiers naturels.

Exercice 3 (**Listes**)

1. Ecrire une fonction qui compte le nombre d'éléments d'une liste

2. Ecrire une fonction qui compte le nombre d'éléments pairs d'une liste d'entiers

3. Ecrire la fonction `somme` qui fait la somme des nombres d'une liste de flottants.

4. Ecrire une fonction `min` qui retourne le plus petit entier d'une liste d'entiers. Peut on l'utiliser avec une liste de chaînes de caractères ?

Exercice 4 (**couples**)
Le rationnel $\frac{p}{q}$ sera représenté par le couple (p, q) de type `int*int`.

1. Ecrire la fonction `inverse_ratio` qui calcule l'inverse d'un nombre rationnel.

2. Ecrire la fonction qui réalise l'addition de 2 nombres rationnels. On ne cherchera pas àsimplifier le rationnel résultat.