TD 1

Exercice d'initiation à Ocaml

Exercice 1 : Donner les résultats des expressions suivantes, sans utiliser l'ordinateur, et corriger en cas d'erreur de syntaxe :

(a) `if 2>3 then 4 ;;`

(b) `1>2 and 5>3 ;;`

(c) `3.5 + 4.0 ;;`

(d) `1+4.0 ;;`

(e) `8./.3. ;;`

(f) `8/3 ;;`

(g) `1.0<2.0 || 3>4 ;`

Exercice 2 :

1. Définir deux fonctions pour renvoyer la valeur absolue dont on donne la signature :

   ```
   val_abs_int : int -> int = <fun>;
   ```

   et

   ```
   val_abs_float : float -> float = <fun>
   ```

2. Définir deux fonctions `signe` qui renvoie 1 si la donnée est positif, 0 si elle est nulle et -1 si elle est négative. :

   ```
   signe_int : int -> int = <fun>
   ```

   et

   ```
   signe_float : float -> int = <fun>
   ```

Exercice 3 :

1. Écrire une fonction `collatz` qui prend en argument un entier n, et rend $n/2$ si n est pair, $3n+1$ si n est impair.

   ```
   collatz : int -> int = <fun>
   ```

2. Écrire une fonction `syracuse` prend en argument un entier n et retourne le nombre d'utilisation de la fonction `collatz` pour que la suite définie par :

   $$\begin{cases} u_0 &= n \\ u_{n+1} &= \texttt{collatz}(u_n) \end{cases}$$

   atteigne la valeur 1.

   ```
   syracuse : int -> int = <fun>
   ```

   Exemple :

   ```
   syracuse 10;;
   - : int = 6
   ```

Exercice 4 : Écrire une fonction qui prend en arguments trois entiers a, b et c, et rend le plus petit des trois arguments. (on minimisera le nombre de comparaisons.)

Exercice 5 :

1. Écrire une fonction `nb_chiffre` qui renvoie le nombre de chiffres dans l'écriture décimale d'un entier positif.

   ```
   nb_chiffre : int -> int = <fun>
   ```

   Exemple :

   ```
   nb_chiffre 5746;;
   - : int = 4
   ```

2. Écrire une fonction `nb_zero` qui renvoie compte le nombre de fois où le chiffre 0 apparait dans l'écriture d'un entier positif.

   ```
   nb_zero : int -> int = <fun>
   ```

   Exemple :

   ```
   nb_zero 50200;;
   - : int = 3
   ```

3. Écrire une fonction `symetrie` qui renvoie le "symétrique" d'un entier positif.

   ```
   symetrie : int -> int
   ```

   Exemple :

   ```
   symetrie 5237;;
   - : int = 7325
   ```

Exercice 6 : On définit l'ensemble des nombres complexes avec le type suivant :

```
type complexe = { part_reelle : float ; part_imaginaire : float } ;;
```

1. Définir les complexes `zero` $= 0$,`un` $= 1$, `i` , $z_1 = 3 + 2i$ et $z_2 = 5 - i$.
2. Écrire les opérations usuelles sur l'ensemble des complexes.

   ```
   somme : complexe -> complexe -> complexe
   produit : complexe -> complexe -> complexe
   conjugue : complexe -> complexe
   inverse : complexe -> complexe
   quotient : complexe -> complexe -> complexe
   norme : complexe -> float (* le module *)
   ```