Travaux pratiques

option informatique

Exercices de révision (1)

Exercice 1.
Rédiger une fonction qui « aplatit » une liste de listes.

```
flatten : 'a list list -> 'a list
```

Par exemple, `flatten [[1; 2; 3]; [4; 5]]` retournera la liste `[1; 2; 3; 4; 5]`.

Exercice 2.
Rédiger une fonction renvoyant le dernier élément d'une liste vérifiant une propriété donnée.

```
dernier : ('a -> bool) -> 'a list -> 'a
```

On s'interdira d'utiliser l'image miroir de la liste.

Exercice 3.
Rédiger une fonction calculant la liste de tous les préfixes stricts d'une liste donnée. On obtiendra par exemple :

```
# prefixes [1; 2; 3; 4] ;;
- : int list list = [[1]; [1; 2]; [1; 2; 3]; [1; 2; 3; 4]]
```

Exercice 4.
Rédiger une fonction retournant la liste des éléments qui apparaissent au moins deux fois dans une liste donnée. La liste retournée ne devra pas comporter de répétition.

```
au_moins_deux : 'a list -> 'a list
```

Exercice 5. Tri fusion
On rappelle le principe de l'algorithme de tri fusion pour trier une liste de longueur $n \geqslant 2$:

(i) la liste est scindée en deux parties de tailles respectives $\lfloor n/2 \rfloor$ et $\lceil n/2 \rceil$;

(ii) chacune de ces deux listes est triée à l'aide d'un appel récursif ;

(iii) les deux listes triées sont enfin fusionnées.

Rédiger une fonction réalisant le tri fusion d'un élément de type *'a list* .

```
merge_sort : 'a list -> 'a list
```

Exercice 6. Algorithme de Floyd
Si A est un ensemble, on appelle *itérateur* la donnée d'un élément $a \in \mathrm{A}$ et d'une fonction $f : \mathrm{A} \to \mathrm{A}$; un itérateur définit une suite u à l'aide des relations $u_0 = a$ et $u_{n+1} = f(u_n)$.
Lorsque A est un ensemble fini la suite u est périodique à partir d'un certain rang. Justifier l'existence d'un entier $p \geqslant 1$ tel que $u_{2p} = u_p$, puis rédiger une fonction **floyd1** qui, à partir de la donnée d'un itérateur retourne le plus petit entier p vérifiant cette égalité.

```
floyd1 : 'a -> ('a -> 'a) -> int
```

Cet entier est-il nécessairement égal à la plus petite période de la suite u ?
Rédiger une fonction **floyd2** qui retourne la plus petite période d'un itérateur.
En déduire une fonction **periode** qui calcule la période du développement décimal infini de $1/n$, où $n \in \mathbb{N}^*$.