Travaux Pratiques de programmation fonctionnelle n°7

Licence d'informatique

——

▸ **Exercice 1.** ***Tri par insertion***

La fonction suivante effectue l'insertion à la bonne place d'un entier dans une liste triée :

```
# let rec inserer element = function
    | tete::suite when element > tete -> tete::(inserer element suite)
    | liste -> element::liste
val inserer : 'a -> 'a list -> 'a list = <fun>

# inserer 3 [1;5;9];;
- : int list = [1; 3; 5; 9]
```

1. *Programmer la fonction* `tri_par_insertion` *en utilisant la fonction précédente.*
2. *Une fonction de tri peut trier une liste d'éléments de n'importe quel type (entiers, mais aussi chaînes de caractères, par ordre lexicographique) si on lui passe en paramètre la fonction de comparaison pour ce type (*`compare:'a -> 'a -> int`*, où* `compare a b <= 0` *signifie que* `a` *est avant* `b` *dans l'ordre).*

   Modifier les fonctions `inserer` *et* `tri_par_insertion` *pour qu'elles remplissent cette condition.*

▸ **Exercice 2.** ***Manipulation d'une suite de cartes***

On veut manipuler un jeu de 52 cartes. Pour cela on représente les valeurs et les couleurs par des entiers selon le principe suivant :

As $\leftrightarrow 1$, *2* $\leftrightarrow 2$, $\ldots$, *10* $\leftrightarrow 10$, *Valet* $\leftrightarrow 11$, *Dame* $\leftrightarrow 12$, *Roi* $\leftrightarrow 13$

Trèfle $\leftrightarrow 1$, *Carreau* $\leftrightarrow 2$, *Coeur* $\leftrightarrow 3$, *Pique* $\leftrightarrow 4$

On représente alors une carte par un couple (valeur,couleur).

Exemple : Valet de pique $\leftrightarrow (11, 4)$

1. *Utiliser les fonctions de l'exercice 2 pour trier une liste de cartes par ordre de valeur puis de couleur.*
2. *Même question en triant par couleur puis par valeur.*
3. *Écrire une fonction permettant de couper une suite de cartes à la position n (on suppose que n est inférieur au nombre d'éléments de la liste). Exemple :*

```
# couper 3 [(1,2);(7,4);(12,1);(4,3)];;
- : (int * int) list = [4, 3; 1, 2; 7, 4; 12, 1]
```

4. *Écrire une fonction* `generer_couleur` *générant la liste triée des cartes de la couleur passée en paramètre.*
5. *En déduire une fonction* `generer_jeu` *générant un jeu de 52 cartes trié.*
6. *Grâce aux fonctions* `Random.self_init` *(initialisation du générateur de nombres aléatoires) et* `Random.int` *(*`Random.int n` *renvoie un entier compris entre 0 et n), écrire une fonction* `battre` *prenant un jeu complet et le mélangeant aléatoirement. Essayer de trouver une méthode qui mélange bien le jeu (adaptée de votre expérience de joueur).*

▸ **Exercice 3.** ***Non aux tricheurs***

1. *Écrire la fonction* `suite_de_cartes_valide` *qui prend une liste de cartes en paramètres et qui renvoie* `true` *si tous ses éléments ont bien une valeur comprise entre 1 et 13 et une couleur comprise entre 1 et 4, et* `false` *sinon.*
2. *Si on tire honnêtement deux listes de cartes d'un même jeu, elles ne doivent avoir aucune carte commune. Programmez la fonction* `tricherie` *qui prend deux listes de cartes en paramètres et qui vérifie si leur intersection est bien vide.*