Contrôle continu n°3 : Les Listes

Durée 1h30 – Aucun document autorisé – respecter les types des exemples donnés

Le but de ce problème est d'écrire quelques fonctions en Caml permettant de simuler la manipulation d'un jeu de cartes.

Dans tout le problème, on considère un jeu de 32 cartes allant du sept à l'as. Une carte sera représentée par un couple constitué d'une chaîne de caractère représentant sa couleur « pique », « coeur », « carreau » ou « trèfle » et un entier représentant sa valeur : les cartes sept, huit, neuf, dix, valet, dame, roi et as seront respectivement représentées par les entiers de 7 à 14.

Un paquet de cartes est alors représenté par une liste de couples (couleur,valeur). Le jeu complet est donné par :

```
 let jeucomplet=[("pique",7) ;("pique",8) ;("pique",9) ;("pique",10) ;("pique",11) ;
("pique",12) ;("pique",13) ;("pique",14) ; ("coeur",7) ;("coeur",8) ;("coeur",9) ;
("coeur",10) ;("coeur",11) ;("coeur",12) ;("coeur",13) ;("coeur",14) ; ("carreau",7) ;
("carreau",8) ;("carreau",9) ;("carreau",10) ;("carreau",11) ;("carreau",12) ;("carreau",13)
;("carreau",14) ; ("trefle",7) ;("trefle",8) ;("trefle",9) ;("trefle",10) ;("trefle",11) ;
("trefle",12) ;("trefle",13) ;("trefle",14)   ] ;;
```

La « main » qui nous servira d'exemple est :

```
let main=[("trefle",8) ;("pique",7)  ;("coeur",11) ;("trefle",11)  ;("coeur",7) ;
("carreau",10);("pique",10) ;("pique",11)    ] ;;
```

1. On compte les cartes et on les distribue

- **Q1 :** Ecrire une fonction ***long*** calculant le nombre de cartes d'un jeu de cartes.

```
long(main);;
- : int = 8
```

- **Q2 :** Ecrire une fonction ***nbCoul*** qui à un jeu de cartes et une couleur associe le nombre de cartes de cette couleur dans le jeu.

```
nbCoul(main,"pique");;
- : int = 3
```

- **Q3 :** Ecrire une fonction ***retourne*** qui inverse l'ordre des cartes d'un jeu.

```
inverse(main);;
- : (string * int) list = ["pique", 11; "pique", 10; "carreau", 10; "coeur", 7; "trefle",
11;  "coeur", 11; "pique", 7; "trefle", 8]
```

(rappel : ajouter un élément en fin de liste nécessite une concaténation)

- **Q4 :** Ecrire une fonction ***distribue*** qui à partir d'un jeu de carte construit un quadruplet de jeux de cartes obtenu en distribuant une à une les cartes du jeu initial.


```
distribue (main);;

 ["trefle", 8; "coeur", 7], ["pique", 7; "carreau", 10],
 ["coeur", 11; "pique", 10], ["trefle", 11; "pique", 11]
```

(Respecter l'ordre de l'exemple même si ce n'est pas l'ordre « naturel » lorsqu'on distribue des cartes)

2. On trie le jeu de cartes

Les deux fonctions suivantes vont permettre de classer les cartes par couleurs dans l'ordre pique, cœur, carreau, trèfle :

- **Q5 :** Ecrire une fonction ***range*** définie par filtrage qui étant donnés une carte et un quadruplet de jeux de cartes (les piques, les cœurs, les carreaux et les trèfles dans cet ordre), range la carte dans le bon paquet.

```
range(("pique",7),[8;12],[],[7;8;9],[14]);;
[7; 8; 12], [], [7; 8; 9], [14]
```

(Comme sur l'exemple, on ne conservera dans les listes résultats que les valeurs des cartes)

- **Q6 :** Ecrire une fonction ***triCoul*** qui à partir d'un jeu, construit les quatre listes des valeurs des piques, cœurs, carreaux, trèfles.

```
triCoul(main);;
[7; 10; 11], [11; 7], [10], [8; 11]
```

On souhaite maintenant trier chaque liste de valeurs. Pour cela nous allons mettre en œuvre un tri par insertion :

- **Q7 :** Ecrire une fonction ***insère*** qui insère un entier à sa place dans une liste triée

```
insert(7,[2;4;6;9;12;14]);;
[2; 4; 6; 7; 9; 12; 14]
```

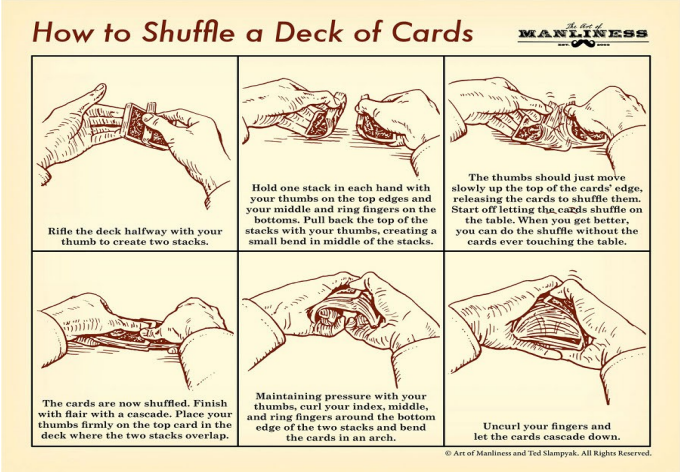
- **Q8 :** Ecrire une fonction ***tri*** mettant en œuvre le tri par insertion d'une liste d'entier (on utilisera la fonction *insère*).

- **Q9 :** Ecrire enfin une fonction ***triCartes*** utilisant *triCoul* et *tri* donnant pour chaque couleur, la liste triée des valeurs des cartes d'un jeu.

```
triCartes (main);;
 [7; 10; 11], [7; 11], [10], [8; 11]
```

3. On mélange le jeu de cartes

Nous allons mettre en place le mélange d'un jeu de carte par la technique du « riffle shuffle », la technique préférée des joueurs de poker...et des tricheurs.



- **<u>Q10 :</u>** Ecrire une fonction ***passe*** qui étant donnés un entier p et deux jeux l1 et l2, fait passer les p premières cartes de l1 en tête de l2.

```
passe(2,[1;2;3;4],[5;6;7;8]);;
- : int list * int list = [3; 4], [2; 1; 5; 6; 7; 8]
```

(Respecter l'ordre de l'exemple)

- **<u>Q11 :</u>** Ecrire une fonction ***coupe*** permettant de couper un jeu de carte en deux. (On devra sûrement utiliser les fonctions passe et long)

```
coupe(main);;
 ["coeur", 7; "carreau", 10; "pique", 10; "pique", 11],
 ["trefle", 11; "coeur", 11; "pique", 7; "trefle", 8]
```

(Respecter l'ordre de l'exemple, même si ce n'est pas l'ordre « naturel » quand on coupe un jeu de carte)

- **<u>Q12 :</u>** Ecrire une fonction ***riffle*** fusionnant deux jeux l1 et l2 en un seul par la technique du riffle-shuffle.

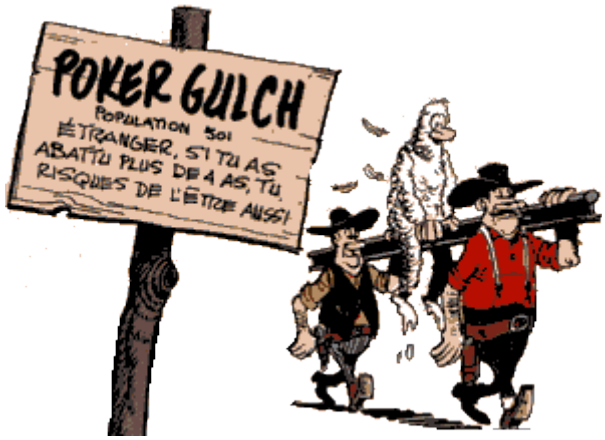
```
riffle ([1;2;3;4],[5;6;7;8;9]);;
- : int list = [1; 5; 2; 6; 3; 7; 4; 8; 9]
```

- **<u>Q13 :</u>** Ecrire une fonction ***shuffle*** qui mélange un jeu de carte en répétant n fois le fait de couper le jeu en deux et de le fusionner par riffle-shuffle.

```
shuffle (5,jeucomplet);;
 ["trefle", 14; "trefle", 12; "trefle", 10; "trefle", 8; "carreau", 14;
  "carreau", 12; "carreau", 10; "carreau", 8; "coeur", 14; "coeur", 12;
  "coeur", 10; "coeur", 8; "pique", 14; "pique", 12; "pique", 10; "pique", 8;
  "pique", 7; "pique", 9; "pique", 11; "pique", 13; "coeur", 7; "coeur", 9;
  "coeur", 11; "coeur", 13; "carreau", 7; "carreau", 9; "carreau", 11;
  "carreau", 13; "trefle", 7; "trefle", 9; "trefle", 11; "trefle", 13]
```

Observer ce qui se passe lorsqu'on effectue un «riffle shuffle » sur le jeu complet 3 fois ou 6 fois.

Vous venez d'apprendre à tricher, je ne vous félicite pas !



Question subsidiaire *: Quel est le nom du célèbre tricheur de l'album de Lucky Luke.*