LMI 1 - Semestre 1

Université d'Artois
Faculté des Sciences Jean Perrin

ALGO1 – Algorithmique et Programmation 1

Fiche de TP numéro bonus - Le jeu du master-mind

Au jeu du master-mind, l'objectif du joueur est de trouver un code secret. Ce code est habituellement sur 4 chiffres, et les chiffres varient entre 0 et 5 (le jeu de plateau propose des combinaisons de pions de couleurs plutôt que de valeurs chiffrées).

À chaque proposition du joueur, on lui indique :

— le nombre de chiffres bien placés dans sa proposition, par rapport au code secret ;

— le nombre de chiffres mal placés dans sa proposition, par rapport au code secret.

L'objectif est de trouver le code secret en le moins de coups possibles. Voilà un exemple d'interaction :

```
>>> main()
Jeu du master-mind

Je viens de générer un code secret à 4 chiffres.
Les chiffres possibles vont de 0 à 5 .
Votre proposition ? (au format xxxx) 0123
Vous avez 2 bien placés et 0 mal placé.
Votre proposition ? (au format xxxx) 1045
Vous avez 0 bien placé et 2 mal placés.
Votre proposition ? (au format xxxx) 2354
Vous avez 1 bien placé et 1 mal placé.
Votre proposition ? (au format xxxx) 0424
Vous avez 0 bien placé et 0 mal placé.
Votre proposition ? (au format xxxx) 5153
Bravo ! Vous avez trouvé en 5 coup(s).
>>>
```

1 Jeu à un joueur

Dans cette version, c'est la machine qui choisit le code secret, et qui doit donner les bonnes indications (nombre de bien placés et de mal placés) à l'utilisateur. C'est ce qui correspond à l'exemple ci-dessus. C'est également ce qui a été traité en cours (cours numéro 7, deuxième cours sur les listes).

On choisit de représenter le code secret et les propositions du joueur sous la forme d'une liste d'entiers.

Cette version est décomposée avec les fonctions suivantes :

— `genere_secret()` qui retourne un code secret ;

— `lire_prop()` qui lit la proposition du joueur, vérifie qu'elle est bien conforme au format attendu, et la retourne sous la forme d'une liste d'entiers. Pour écrire cette fonction, vous aurez besoin de la décomposer à l'aide de deux fonctions :

  — `valide_mm(prop: list[int]) -> bool` qui teste si `prop` est correcte par rapport au format attendu (bon nombre d'éléments, chaque élément dans le bon intervalle)

  — `conversion` qui convertit une liste de chaînes de caractères en une liste d'entiers

Voilà un exemple d'intéraction :

```
>>> rep = lire_prop()
 Votre proposition ? (au format xxxx) 012345
 je n'ai pas compris.
 Il faut 4 chiffres dans l'intervalle [0,5].
 Votre proposition ? (au format xxxx) 012
 je n'ai pas compris.
 Il faut 4 chiffres dans l'intervalle [0,5].
 Votre proposition ? (au format xxxx) 5678
 je n'ai pas compris.
 Il faut 4 chiffres dans l'intervalle [0,5].
 Votre proposition ? (au format xxxx) 0121
>>> rep
[0, 1, 2, 1]
>>>
```

— `bp_mp(prop: list[int], code: liste[int]) -> tuple[int,int]` qui calcule le nombre de bien placé et de mal placés. Une solution est de décomposer en trois fonctions :
  — `bien_places(prop, code)` qui calcule le nombre de bien placés
  — `distribution(prop)` qui calcule combien il y a de chaque valeurs, et retourne une liste dont les indices représentent les valeurs, et le contenu représente le nomber d'occurrences de chaque valeur :

```
>>> distribution([2, 3, 3, 5]) ## cette proposition contient un 2,
                               ## deux 3 et un 5
[0, 0, 1, 2, 0, 1]  ## soit : 0 valeur 0, 0 valeur 1, 1 valeur 2,
                    ## 2 valeurs 3, 0 valeur 4 et 1 valeur 5
```

  — `valeurs_communes(prop, code)` qui calcule, après avoir obtenu la distribution du code et de la proposition, leur nombre de valeurs communes
  — on remarquera que le nombre de mal placés est le nombre de valeurs communes moins le nombre de bien placés…
— `affiche_reponse` qui affiche le résultat obtenu pour une proposition
— `main_1joueur` qui est la boucle principale du jeu.

Reprenez ce code.

2 Jeu à deux joueurs

La seconde version du jeu fait apparaître un deuxième joueur. La machine va générer deux codes secrets : un code secret pour le joueur humain, un code secret pour la machine. Chacun cherche à deviner son code secret, le premier qui a trouvé a gagné.

Bien sûr, la machine ne va pas tricher : elle calcule les bien placés et les mal placés des propositions qu'elle fera, mais lorsqu'elle fait une proposition, elle le fait sans regarder le code secret.

Voici un exemple d'interaction :

```
>>> main_2joueurs()
 Jeu du master-mind

 Je viens de générer deux codes secrets à 4 chiffres.
 À vous ! Les chiffres possibles vont de 0 à 5 .
 Votre proposition ? (au format xxxx) 0123
 Vous avez 0 bien placé et 3 mal placés.

 MasterMind-L1MathsInfo propose : 4132
 Il obtient 1 bien placé et 1 mal placé.

 Votre proposition ? (au format xxxx) 1234
 Vous avez 0 bien placé et 2 mal placés.

 MasterMind-L1MathsInfo propose : 4200
 Il obtient 0 bien placé et 3 mal placés.

 Votre proposition ? (au format xxxx) 2350
 Vous avez 1 bien placé et 2 mal placés.

 MasterMind-L1MathsInfo propose : 5042
 Il obtient 2 bien placés et 2 mal placés.

 Votre proposition ? (au format xxxx) 3610
 je n'ai pas compris.
 Il faut 4 chiffres dans l'intervalle [0,5].
 Votre proposition ? (au format xxxx) 3010
 MasterMind-L1MI a proposé 0542
 Il a trouvé en 4 coup(s).
>>>
```

Implémentez la version du jeu master-mind à deux joueurs sous la forme d'une fonction `main_2joueurs()`. On vous propose des indices qui vous aideront à implémenter cette fonction :

- Premier indice : les propositions faites par la machine se basent sur un historique qui stocke le nombre de bien placés et de mal placés, ainsi que toutes les propositions précédentes. (N'oubliez pas d'utiliser une liste de dictionnaire.)
- Deuxième indice : la première proposition de la machine est aléatoire.
- Troisième indice : ne pas envisager que la machine génère des combinaisons possibles pour trouver le code secret.

3 Fonction principale

Spécifiez puis écrivez la fonction principale du programme (main) qui ne reçoit pas d'arguments et ne retourne rien. Cette fonction devrait utiliser les fonctions créées précédemment. Elle aura pour but de demander à l'utilisateur de choisir le mode de jeu, que ce soit de jouer seul(e) ou de jouer avec la machine. Voici un exemple d'interaction :

```
============= Master-Mind =============
Choisir le mode du jeu : 1 (un seul joueur) ou 2 (deux joueurs) : a
 metter 1 ou 2 svp : b
 metter 1 ou 2 svp : 2
============= Mode avec deux joueurs =============
Jeu du master-mind

Je viens de générer deux codes secrets à 4 chiffres.
À vous ! Les chiffres possibles vont de 0 à 5 .
Votre proposition ? (au format xxxx)
```