KID CALA

Python ING1 – TP Noté

14 Décembre 2021

Durée : 3h00

Tous documents autorisés

Pour ce TP, vous aurez le droit d'utiliser le module *numpy* de Python, plus tout autre module que vous jugerez utile.

Attention : ce TP est noté. Seront pris en compte dans la notation:

- la clarté du code ;
- L'organisation (fonction et modules)
- la qualité de la documentation
- la présence des tests unitaires
- la gestion des exceptions
- la vitesse d'exécution (idéalement en secondes…)

Attention : Il ne vous est pas demandé de calculer automatiquement le temps d'exécution, mais seulement d'optimiser suffisamment votre code, de telle sorte que le temps de calcul soit acceptable.

Les codes devront être remis sur formationTemp dans le Dossier « ING1\Python\Rendus ». Vos fichiers devront être remis sous forme d'une archive .zip nommés sous la forme **NOM_PRENOM.zip** avant la fin du temps imparti. Tout rendu ne respectant ces critères ne sera pas évalué.

Toute tentative de triche entraînera l'exclusion de la salle ainsi que la note de 0.

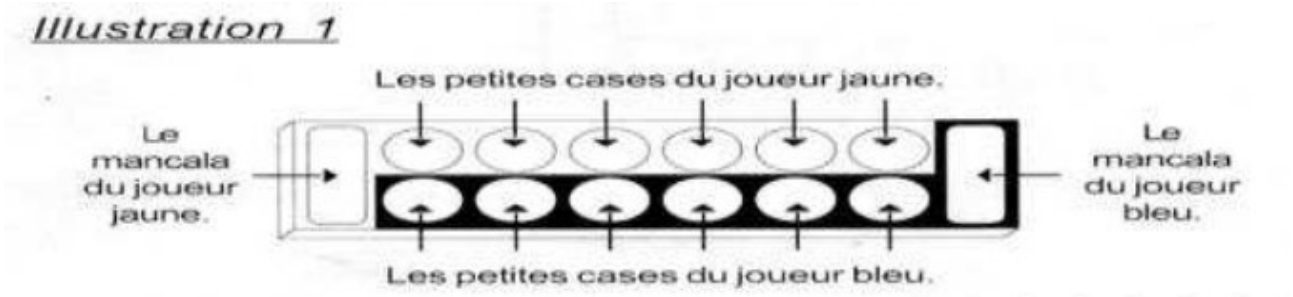
Toute tentative de réponse à une question, même incomplète sera prise en compte dans la notation. Vous avez le droit de traiter les questions dans l'ordre que vous voudrez. Il est conseillé de lire le sujet en entier avant de commencer à coder.

Kid Cala

Dans le cadre de ce TP nous allons coder le jeu Kid Cala.

Règle du jeu :

Le Kid-Cala est un dérivé du jeu Mancala et se joue à deux : un joueur bleu et un joueur jaune. Entre les deux joueurs se trouve le plateau que vous pouvez visualiser ci-dessus. Le 'terrain' d'un joueur est constitué par les petites cases de sa couleur ainsi que la grosse case (dit le 'mancala') de cette même couleur.



En début de partie, il y a dans les douze petites cases, toutes couleurs confondues, 4 fruits différents : une pomme, une orange, une banane et une fraise. Les mancalas sont vides en début de partie.

Le but du jeu est d'amasser un maximum de fruits dans son mancala en suivant le déroulement suivant :

Le déroulement du jeu :

1. A chaque tour, le joueur prend toutes les pièces de fruits d'une des 6 cases de sa propre couleur (le joueur choisi lui même la case mais ne peut jamais prendre les pièces de fruits des mancalas ou des cases de son adversaire). Le joueur dépose toutes les pièces de fruits, une par une dans les cases suivantes (dans le sens inverse des aiguilles d'une montre) ;
2. Lors du dépôt des fruits dans les cases, si le joueur arrive à un des deux mancalas, il y dépose également un fruit (que ce soit son propre mancala ou celui de son adversaire) ;
3. Dès qu'un joueur n'a plus de fruit dans ses petites cases, le jeu est fini. L'adversaire prend les fruits restant dans ses propres cases et les dépose dans son mancala ;
4. Le joueur avec le plus de pièces de fruits dans son mancala gagne le jeu.

Nous proposons donc de coder ce jeu. Nous ferons quelques hypothèses en cours de route.

Création du jeu

Question 1 : Créez un dictionnaire qui associe le nom du fruit à un chiffre (par exemple : « 0 : banane»). On rappel qu'il y a 4 sortes de fruits (Pomme, Orange, Banane, Fraise).

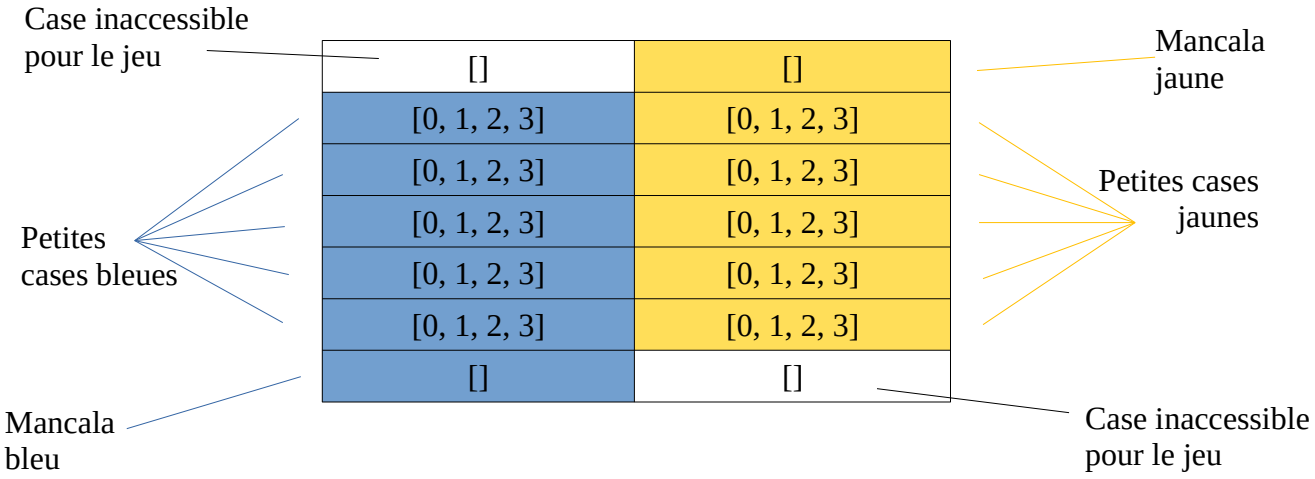
Question 2 : Proposez une fonction qui crée un tableau avec deux colonne et huit lignes. Chacune de ses cases sera remplie par une liste vide. Cette fonction devra renvoyer le tableau en question.

Question 3 : Proposez une fonction qui permet de remplir le tableau comme suit : chaque case doit avoir la liste suivante $[0, 1, 2, 3]$, sauf les cases de la première et dernière ligne. Cette fonction prendra en paramètre le tableau « vide » et devra renvoyer le tableau rempli.

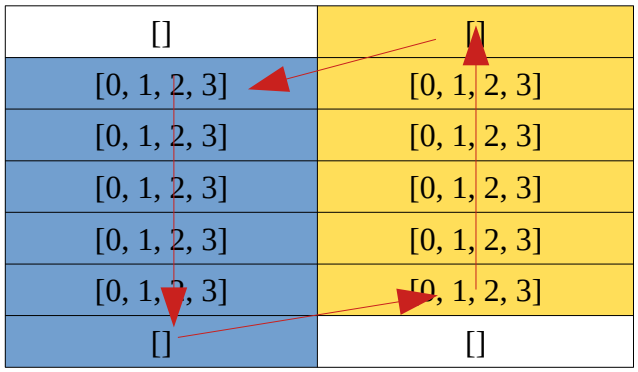
Nous devons donc avoir une tableau comme le suivant :

[]	[]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[]	[]

Nous allons donc considérer que le tableau sera notre plateau de jeu. On peut le visualiser ainsi :



Maintenant que nous avons notre « plateau de jeu », nous allons récupérer les fruits dans les petites cases et les déplacer dans les cases suivantes. Chaque fruit est représenté par son chiffre. Nous définissons le sens de circulation des fruits comme étant le sens inverse des aiguilles d'une montre.



Interaction

Nous allons introduire le fait qu'il y a deux joueurs dans une partie, nous pourrons considérer que le joueur 0 sera le joueur bleu, et le joueur 1 sera le joueur jaune.

Question 4 : Proposez une fonction qui permet au joueur de sélectionner une case à jouer. Cette fonction devra prendre en paramètre le tableau et le joueur. Le numéro du joueur impliquera la colonne à choisir, c'est-à-dire que le joueur ne doit seulement renseigner la ligne qu'il veut jouer. La fonction empêchera le joueur de choisir une case vide, ainsi que les mancala. La fonction renverra les indices (i, j) de la case choisie par le joueur.

Question 5 : Proposez une fonction qui renvoie le contenu d'une case à partir de ses indices (i, j).

Question 6 : Lorsqu'un joueur a choisi une case, il faut remplacer son contenu par une liste vide. Proposez une fonction qui permet remplacer le contenue d'une case choisie par une liste vide. Elle renverra le tableau modifié.

Question 7 : En utilisant certaines fonctions précédemment développées, proposez une fonction qui permet récupérer le contenu d'une case, de vider cette case et de répartir les fruits récupérés un à un dans les cases

suivantes. Cette fonction pourra, par exemple, prendre en paramètre le joueur, le tableau à modifier et la case choisie par le joueur. Elle renverra le tableau modifié.

Ex :

Etape 0 (aucun joueur n'a encore joué) :

[]	[]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[]	[]

Etape 1 (joueur 0 vient de jouer, les fruits déplacés sont en rouge) :

[]	[]
[0, 1, 2, 3]	[0, 1, 2, 3]
[0, 1, 2, 3]	[0, 1, 2, 3]
[]	[0, 1, 2, 3]
[0, 1, 2, 3, **1**]	[0, 1, 2, 3]
[0, 1, 2, 3, **2**]	[0, 1, 2, 3, **4**]
[**3**]	[]

Etape 2 (joueur 1 vient de jouer, les fruits déplacés sont en vert) :

[]	[**0**]
[0, 1, 2, 3, **1**]	[]
[0, 1, 2, 3, **2**]	[0, 1, 2, 3]
[**3**]	[0, 1, 2, 3]
[0, 1, 2, 3, **1**]	[0, 1, 2, 3]
[0, 1, 2, 3, **2**]	[0, 1, 2, 3, **4**]
[**3**]	[]

Jeu et Fin

Le jeu se termine quand un des joueurs n'a plus que des cases vides dans sa propre colonne. Le joueur adversaire récupère alors tous les fruits qu'il reste dans sa colonne et les ajoute dans son macala. Le vainqueur est celui qui a le plus de fruits dans son macala.

Question 8 : Proposez une fonction qui permet de tester si le jeu est fini. Cette fonction renverra un booléen.

Question 9 : Lorsque le jeu est fini, proposez une fonction qui permet au joueur qui a encore des fruits dans ses cases de les rapatrier dans son propre mancala.

Question 10 : Proposez une fonction qui permet de compter le nombre de fruits distincts dans les mancalas de chaque joueur. Cette fonction renverra deux dictionnaires (un pour chaque joueur) sous la forme :

{‘Banane’ : 8, ‘Orange’ : 2, ‘Pomme’ : 3, ‘Fraise’ : 5}

Question 11 : Proposez une fonction qui donne le vainqueur entre les deux joueurs. Cette fonction renverra le vainqueur, mais aussi le nombre total de fruits ainsi que le nombre de fruits distincts pour chacun des joueurs. (Donc un résultats qui pourra, par exemple, être sous la forme « Le joueur 0 gagne avec un total de 15 fruits. Le joueur 0 cumule 5 bananes, 2 oranges, 3 pommes et 5 fraises. Le joueur 1 cumul 3 bananes, 1 oranges, 2 pommes et 1 fraises. »).

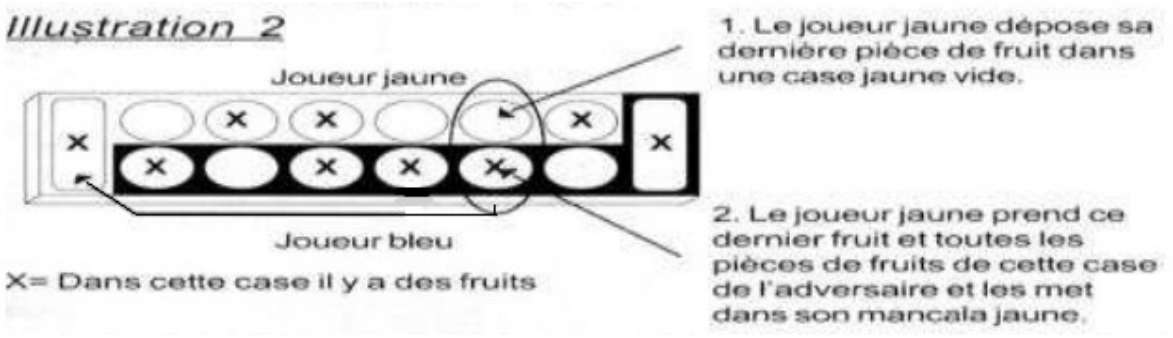
Question 12 : Proposez une fonction qui permet de jouer et d’alterner entre les deux joueurs, et ce tant que le jeu n’est pas fini. Pour cette question, vous devrez utiliser certaines fonctions précédemment développées.

Les spécificités du jeu

Si vous arrivez à cette partie, merci de coder les prochaines fonctions dans un fichier séparé (afin que vous ne modifiiez pas vos précédentes fonctions). Cela nous simplifiera la vie lors de la correction.

Pour simplifier ce TP, nous avons volontairement « oublié » certaines règles du jeu, à savoir :

- *Règle nouvelle 1* : Lors de son tour de jeu, si un joueur a posé sa dernière pièce de fruit dans son propre mancala, il a le droit de rejouer ;
- *Règle nouvelle 2* : Si un joueur a posé sa dernière pièce de fruit dans une de ses cases vides, il peut prendre cette pièce de fruit et en plus les pièces de fruits de la case en face de son adversaire et les mettre dans son mancala. S’il n’y a pas de fruit dans la case de l’adversaire, on ne récupère rien dans le mancala ;



Maintenant que nous avons ces informations :

Question 13 : Proposez une fonction qui permet à un joueur de rejouer s’il respecte les conditions de la *règle nouvelle 1*.

Question 14 : Proposez une fonction qui permet à un joueur de rejouer s’il respecte les conditions de la *règle nouvelle 2*.

Question 15 : Proposez une nouvelle fonction qui permet de jouer toute une partie avec ces nouvelles règles.

Question 16 : Proposez une fonction qui permet de jouer « contre l’ordinateur ».