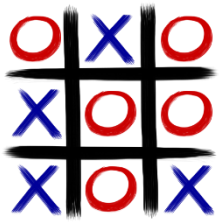
TD : Tic-Tac-Toe

L'objectif de ce TD est réaliser une intelligence artificielle pour le Tic-Tac-Toe, assez douée pour ne jamais perdre.

1) Représentation du problème

a) Rappel des règles



Le Tic-Tac-Toe est un petit jeu de réflexion dont les règles sont très simples :

- Deux joueurs jouent à tour de rôle sur une grille de 3x3 cases
- A chaque tour, un joueur trace un symbole dans une case libre
  - Une croix pour le premier joueur
  - Un rond pour le second joueur
- Si un joueur parvient à aligner 3 symboles identiques, il remporte la partie
- Si aucun joueur ne parvient à aligner 3 symboles identiques, c'est un match nul.

b) Représentation en Python

Pour représenter la grille en Python, on va utiliser des tableaux. Plus exactement, on va utiliser un tableau de tableaux nommé `grille`, tel que `grille[i][j]` décrira le contenu de la case située sur la ligne `i` et la colonne `j`.

Attention : En Python aussi, les indices des tableaux commencent à 0.

Le contenu de la case sera décrit par un entier :

- 0 pour une case vide
- 1 pour une case contenant une croix
- 2 pour une case contenant un rond

c) Création de la grille

Pour créer une grille (c'est-à-dire un tableau de tableaux), utilisez la fonction suivante

```
def creer_grille(taille):
    grid = [[]] * taille
    for ligne in range(taille):
        grid[ligne] = [0] * taille
    return grid;
```

Question 1.a. Recopier la fonction ci-dessus et l'utiliser pour créer une grille de taille 3.

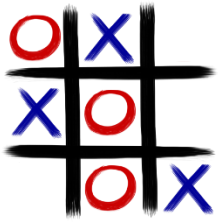
Question 1.b. Tester la fonction `print` sur cette grille.

d) Lecture et écriture dans la grille

La syntaxe pour lire et modifier la grille est relativement simple

- `grille[i][j]` décrit le contenu de la case située sur la ligne `i` et la colonne `j`.
- Pour modifier ce contenu, il suffit d'écrire `grille[i][j] = nouvelle_valeur`

Question 2. Créer une grille correspondant à la situation ci-dessous :



Question 3. Créer une fonction qui affiche la grille sur plusieurs lignes grâce aux symboles `X`, `O` et espace. Par exemple, la grille ci-dessus doit s'afficher de la façon suivante :

```
OX
XO
  OX
```

2) Simulation d'une partie

a) Jouer un coup

Question 4. Ecrire une fonction qui détermine si un coup est valide. Cette fonction aura pour argument la grille et les coordonnées du coup.

Question 5. Ecrire une fonction qui joue un coup aléatoire valide sur une grille. Cette fonction aura pour argument la grille et le numéro du joueur.

Remarque : S'il n'y a plus de coup valide à jouer, utiliser la syntaxe `raise Exception("Message à afficher")` pour provoquer une erreur.

b) Fin de partie

Question 6. Ecrire une fonction qui teste si la partie est terminée, et le cas échéant s'il y a un gagnant. Cette fonction aura pour argument la grille, et renverra un couple formé de :

- Un booléen indiquant si la partie est terminée
- Un entier indiquant le numéro de l'éventuel gagnant (0 en cas de match nul)

3) Intelligence artificielle

Question 7. Créer une intelligence artificielle pour le joueur 2 en écrivant une fonction qui renvoie pour une grille donnée le coup à jouer. Cette intelligence artificielle ne doit jamais perdre : elle doit provoquer un match nul si elle ne peut pas remporter la partie.

Question 8. Créer une intelligence artificielle pour le joueur 1 en écrivant une fonction qui renvoie pour une grille donnée le coup à jouer. Cette intelligence artificielle ne doit jamais perdre : elle doit provoquer un match nul si elle ne peut pas remporter la partie.