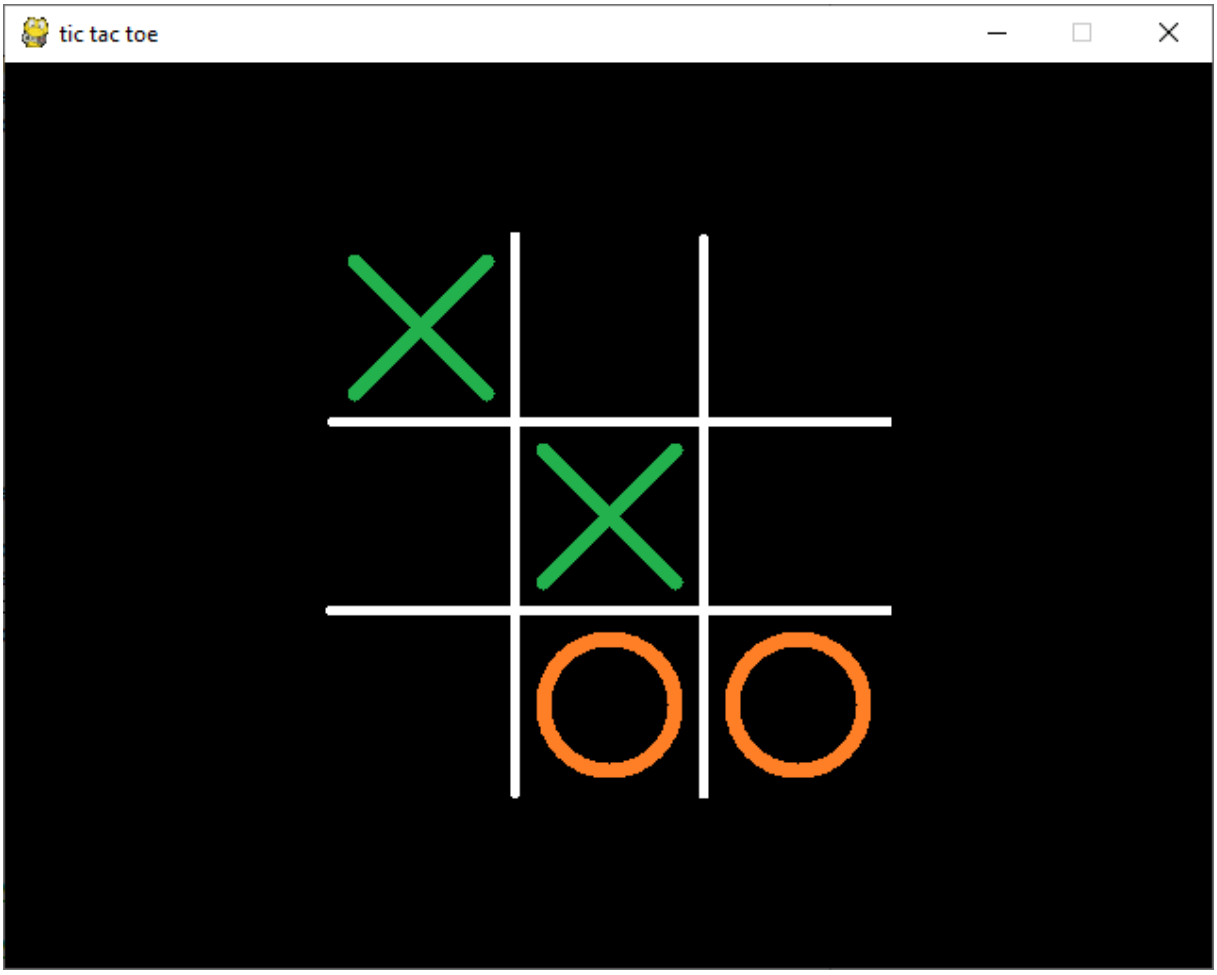
Tic-Tac-Toe : Console

Le jeu



Le tic-tac-toe se joue à deux.

Les joueurs posent tour à tour un rond, pour l'un, une croix, pour l'autre, dans une grille de 3 cases par 3. Le but du jeu est d'obtenir un alignement (en ligne, colonne ou diagonale) de ses trois signes.

Analyse du programme à réaliser

Préparation du jeu

Pour le joueur les cases du plateau de jeu sont numérotées comme ci-dessous.

1	2	3
4	5	6
7	8	9

Nous devons donc mémoriser tout au long du jeu, l'emplacement des croix et des ronds de chacun des joueurs. Nous utiliserons donc une matrice 3 lignes et 3 colonnes, dans laquelle une case vide sera à 0, une case contenant une croix sera à 1 et une case contenant un rond sera à 2.

Déroulement du jeu

En suposant que le joueur 1 commence, nous devons lui demander le numéro de la case qu'il veut jouer.

- Nous devrons vérifier que ce numéro est compris entre 1 et 9
- Nous devrons vérifier que la case choisie est libre
- Nous devrons enregistrer 1 dans la case choisie, après avoir converti son numéro en n° ligne, n° colonnne.

Un exemple de début de partie :

Joueur 1 quel est votre choix (1-9) ?

1

1	0	0
0	0	0
0	0	0

Joueur 2 quel est votre choix (1-9) ?

3

1	0	2
0	0	0
0	0	0

On continue comme cela jusqu'à ce qu'un des deux joueurs gagne ou jusqu'à ce qu'il y ait une partie nulle.

Si le joueur veut arrêter le jeu avant la fin, il pourra taper la valeur 10.

Coup gagnant

Pour vérifier si un coup est gagnant nous recherchons sur le plateau l'alignement de trois fois le chiffre du joueur, en ligne, en colonne et sur les deux diagonales.

Partie nulle

Une partie est nulle lorsqu'il ne reste plus de case à jouer et qu'aucun des deux joueurs n'a été déclarés gagnant.

Première version du programme

```
import numpy as np

import random


# Constantes

NB_LIGNE = 3

NB_COLONNE = 3

HUMAIN = 1

ORDINATEUR = 2

# ----------Partie console----------

# Creer un plateau vide

def creer_plateau():

        return(np.array([[0, 0, 0],

                                [0, 0, 0],

                                [0, 0, 0]]))
```

```
#Renvoie le numéro de la case jouée par le joueur après vérification de ce numéro
#La valeur 10 indique que le joueur veut arrêter le jeu avant la fin
def choix_joueur(plateau, joueur):
    choix = 0
    entree= True
    while entree:
        print('Joueur', joueur,' votre choix? (1-9): 10 pour arrêter')
        choix = int(input())
        if choix == 10:
            return 10
        if choix > 0 and choix < 10:
            if emplacement_libre(plateau,choix-1):
                entree = False
    #On renvoie choix - 1, car dans la matrice mémorisant les coups joués
    #le numéro des lignes et des colonnes commence à 0
    return choix-1


#Vérifie que la case jouée est libre
def emplacement_libre(plateau, numCase):
    #Conversion du numéro de la case jouée en un numéro de ligne et de colonne
    ligne = numCase//3
    colonne = numCase%3
    #si la case contient 0 alors elle est libre
    if plateau[ligne][colonne]==0:
        return True
    else:
        return False
```

```
def jouer_coup(plateau, joueur, numCase):
    #Conversion du numéro de la case jouée en un numéro de ligne et de colonne
    ligne = numCase//3
    colonne = numCase%3
    #enregistrement du chiffre du joueur dans la case
    plateau[ligne][colonne] = joueur

#Recherche un alignement de 3 fois le chiffre du joueur
def coup_gagnant(plateau, joueur):
    #test de toutes les positions horizontales
    for r in range (3):
        if plateau[r][0]== joueur and plateau[r][1]== joueur and plateau[r][2]== joueur:
            return True
    #test de toutes les positions verticales
    for c in range (3):
        if plateau[0][c]== joueur and plateau[1][c]== joueur and plateau[2][c]== joueur:
            return True
    #test des diagonales
    if plateau[0][0]== joueur and plateau[1][1]== joueur and plateau[2][2]== joueur:
        return True
    if plateau[0][2]== joueur and plateau[1][1]== joueur and plateau[2][0]== joueur:
        return True

#Détermine s'il reste encore des cases à jouer
def partie_nulle(plateau):
    for c in range (3):
        for r in range (3):
            if plateau[r][c]== 0:
                return False
```

```
    return True
# ============================================================
# -------------------Partie Principale ------------------
game_over = False
#tirage au sort du premier joueur
tour = random.randint(1, 2)
plateau = creer_plateau()
print(plateau)
#-------------------Boucle de jeu ----------------------------------------
while not game_over:
    if tour == HUMAIN:
        choix = choix_joueur(plateau, HUMAIN)
        if choix == 10:
            game_over = True
        else:
            jouer_coup(plateau,HUMAIN,choix)
            print(plateau)
            if coup_gagnant(plateau, HUMAIN):
                print('Joueur',HUMAIN,' gagne')
                game_over = True
            else:
                if partie_nulle(plateau):
                    print('Partie nulle')
                    game_over = True
                else:
                    tour = ORDINATEUR
```

```
#------------------
else:
    choix = choix_joueur(plateau, ORDINATEUR)
    if choix == 10:
        game_over = True
    else:
        jouer_coup(plateau,ORDINATEUR,choix)
        print(plateau)
        if coup_gagnant(plateau, ORDINATEUR):
            print('Joueur',ORDINATEUR,' gagne')
            game_over = True
        else:
            if partie_nulle(plateau):
                print('Partie nulle')
                game_over = True
            else:
                tour = HUMAIN
```

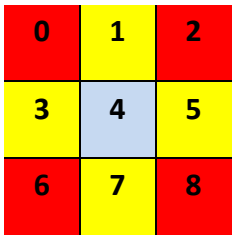
Jouer contre l'ordinateur

Nous devons maintenant faire vraiment jouer l'ordinateur en déterminant qu'elle serait pour lui la meilleure case à jouer.

Comme il y a peu de cases à analyser sur le plateau, à chaque tour de l'ordinateur, nous chercherons dans l'ordre :

1. À voir si une des cases restantes permet de gagner. Si oui l'ordinateur joue cette case
2. À voir si une des cases restantes permet au joueur de gagner. Si oui l'ordinateur joue cette case pour contrer le joueur.
3. À voir si un des coins est libre. Si oui on choisit au hasard parmis ceux qui sont libres.
4. À voir si le centre est libre. Si oui l'ordinateur joue cette case
5. Enfin l'ordinateur joue au hasard une des cases situées sur un bord.

Rappel : les cases de la matrice mémorisant les coups joués, ont des numéros qui vont de 0 à 8.



Les coins sont donc dans la matrice représentant le plateau les cases 0, 2, 6, 8

Le centre est la case 4

Les cases sur les bords sont les cases 1, 3, 5, 7

La fonction qui fait le choix pour l'ordinateur

```
def Ordi_Joue(tableau):
    #1. trouver une case gagnante pour l'ordinateur
    for numCase in range(9):
        #Il faut travailler sur une copie du plateau
        #pour tester si le coup joué est gagnant
        tableau2 = np.copy(tableau)
        if emplacement_libre(tableau2, numCase):
            jouer_coup(tableau2, ORDINATEUR, numCase)
            if coup_gagnant(tableau2, ORDINATEUR):
                return numCase

    #2.Tester s'il reste une case gagnante pour le joueur et le contrer
    for numCase in range(9):
        tableau2 = np.copy(tableau)
        if emplacement_libre(tableau2, numCase):
            jouer_coup(tableau2, HUMAIN, numCase)
            if coup_gagnant(tableau2, HUMAIN):
                return numCase
```

```
    #3. Essayer de prendre un des coins s'ils sont libres
    coins=[0,2,6,8]
    coinsPossibles = []
    for i in coins:
        if emplacement_libre(tableau2, i):
            coinsPossibles.append(i)
    if len(coinsPossibles) != 0:
        return random.choice(coinsPossibles)

    #4. Essayer de prendre le centre s'il est libre
    if emplacement_libre(tableau2, 4):
        return 4

    #5. Choisir une des cases au bord
    bords=[1, 3, 5, 7]
    bordsPossibles = []
    for i in bords:
        if emplacement_libre(tableau2, i):
            bordsPossibles.append(i)
    if len(bordsPossibles) != 0:
        return random.choice(bordsPossibles)
```

Pour terminer, dans la boucle de jeu, il suffit de remplacer :

```
else:
        choix = choix_joueur(plateau, ORDINATEUR)
        if choix == 10:
            game_over = True
        else:
            jouer_coup(plateau,ORDINATEUR,choix)
            print(plateau)
```

par

```
else:
    choix = Ordi_Joue(plateau)
    jouer_coup(plateau,ORDINATEUR,choix)
    print(plateau)
```