Puissance 4 : version console

Le jeu

Puissance 4 est un jeu de stratégie combinatoire qui se joue exclusivement à deux joueurs.

Les règles de Puissance 4

Début de partie
Pour commencer une partie de puissance 4, on désigne le joueur qui commence.

Déroulement de la partie
Il met un de ses jetons de couleur dans l'une des colonnes de son choix. Le jeton se retrouve en bas de la colonne. Le deuxième joueur insère à son tour son jeton, de l'autre couleur dans la colonne de son choix. Les joueurs répètent la manœuvre jusqu'à ce qu'un joueur aligne 4 pions.

Fin de la partie
Le premier joueur à aligner 4 pions, que ce soit horizontalement, verticalement ou en diagonale, remporte la partie.

Analyse du programme à réaliser

Préparation du jeu

Le jeu puissance 4 se joue sur une grille verticale de six lignes et sept colonnes avec 21 jetons rouges et 21 jetons jaunes.

Nous devons donc mémoriser tout au long du jeu, l'emplacement des jetons de chacun des joueurs. Nous utiliserons donc une matrice 6 lignes et 7 colonnes, dans laquelle une case vide sera à 0, une case contenant le jeton du premier joueur sera à 1 et une case contenant le jeton du second joueur sera à 2.

Matrice au démarrage du jeu :

```
[0. 0. 0. 0. 0. 0. 0.]
[0. 0. 0. 0. 0. 0. 0.]
[0. 0. 0. 0. 0. 0. 0.]
[0. 0. 0. 0. 0. 0. 0.]
[0. 0. 0. 0. 0. 0. 0.]
[0. 0. 0. 0. 0. 0. 0.]
```

Le joueur 1 joue et choisit la colonne 3

```
Joueur 1 : Entrez la colonne sélectionnée (0-6):3
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0.]]
```

La case [0][3] de la matrice contient maintenant la valeur 1

Le joueur 2 joue et choisit la colonne 2

```
Joueur 2 : Entrez la colonne sélectionnée (0-6):2
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 2. 1. 0. 0. 0.]]
```

La case [0][2] de la matrice contient maintenant la valeur 2

Le joueur 1 joue et choisit de nouveau la colonne 3

```
Joueur 1 : Entrez la colonne sélectionnée (0-6):3
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 2. 1. 0. 0. 0.]]
```

La case [1][3] de la matrice contient maintenant la valeur 1

```
#la bibliothèque numpy permet de créer et gérer facilement les matrices
import numpy as np
NB_LIGNE = 6
NB_COLONNE=7
#créer une matrice NB_LIGNE, NB_COLONNE remplie de zeros
def creer_plateau():
    plateau = np.zeros((NB_LIGNE,NB_COLONNE))
    return plateau

#afficher la matrice en inversant l'ordre des lignes :
#la matrice créée par numpy a sa première ligne en haut.
#Nous inversons simplement l'affichage des lignes, de façon à avoir
#la ligne 0 en bas, comme dans la grille du jeu
def afficher_plateau(plateau):
    print(np.flip(plateau,0))
#=====================================================
plateau = creer_plateau()
afficher_plateau(plateau)
```

Tester ce début de jeu

Décider si un coup est possible

Lorsque le joueur a choisi une colonne, nous devons voir si dans cette colonne il existe encore une place vide.

Pour savoir s'il est possible de jouer, il suffit de regarder si la colonne jouée est pleine ou non, c'est-à-dire si sa dernière case (celle du haut) est vide ou non.

Si elle est vide, on peut jouer dans la colonne. Si elle n'est pas vide, on ne peut pas y jouer. Notons que si elle est vide, ça ne veut pas dire que le pion restera dans cette case. Il tombera...

Nous ajoutons à notre programme la fonction :

```
#Y a t'il encore de la place dans la colonne choisie
```

```
def emplacement_valide(plateau,colonne):
    if plateau[NB_LIGNE-1][colonne]==0:
        return True
    else:
        return False
```

Pour tester nous ajoutons en fin de programme :

```
colonne = int(input("Joueur 1 : Entrez la colonne sélectionnée (0-6):"))
print(emplacement_valide(plateau,colonne))
```

Tester : l'instruction print doit afficher True, puisque notre plateau est entièrement vide.

Lacher un jeton dans une colonne

Une fois que l'on s'est assuré que la colonne choisie par le joueur n'est pas pleine, il faut trouver à quelle ligne de la colonne le jeton va tomber.

Lorsque la grille est vide, le jeton tombe au fond de la colonne, soit sur la ligne 0, sinon il tombe sur la première ligne libre de la colonne.

Le plus simple est donc de partir du fond de la colonne et de chercher la première case vide en remontant.

La fonction doit renvoyer le numéro de la ligne trouvée.

```
#quelle est la ligne libre dans la colonne choisie
def trouver_Ligne_Vide(plateau,colonne):
    for r in range(NB_LIGNE):
        if plateau[r][colonne]==0:
            return r
```

Une fois que le numéro de la ligne où tombe le jeton est trouvé, il faut mettre à jour la matrice, avec le numéro du joueur.

```
#placer le numéro du joueur dans la colonne choisie, sur la ligne libre
def lacher_jeton(plateau, ligne, colonne, jeton):
    plateau[ligne][colonne] = jeton
```

Tester en ajoutant ces deux fonctions et en écrivant comme programme principal :

```
#=======================================================

plateau = creer_plateau()

afficher_plateau(plateau)

for i in range(4):

   colonne = int(input("Joueur 1 : Entrez la colonne sélectionnée (0-6):"))

   if emplacement_valide(plateau, colonne):

      ligne=trouver_Ligne_Vide(plateau,colonne)

      lacher_jeton(plateau,ligne,colonne,1)

      afficher_plateau(plateau)
```

Nous faisons une boucle qui va demander 4 fois au jour 1 d'entrer un numéro de colonne.

Cela va permettre de voir si les jetons s'entassent bien les uns sur les autres dans une colonne.

```
Joueur 1 : Entrez la colonne sélectionnée (0-6):2
[[0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 0. 1. 0. 0. 0.]
 [0. 0. 1. 1. 0. 0. 0.]]
```

Dans cet essai, j'ai choisi 3 fois la colonne 3 et une fois la colonne 2.

Le coup joué est-il gagnant ?

Nous devons compter à partir d'une position, le nombre de jeton de la même couleur à l'horizontal, à la verticale et en diagonal. Si nous avons 4 pions alignés, alors le joueur a gagné.

Il n'est pas utile de passer toutes les positions en revue. En effet en verticale par exemple, à partir de la ligne 3, il n'est plus possible de faire un alignement de 4 jetons de la même couleur.

De même à l'horizontal, à partir de la colonne 4, il n'est plus possible de faire un alignement de 4 jetons de la même couleur.

Sur les diagonales allant vers la droite, à partir de la ligne 3, colonne 4, il n'est plus possible de faire un alignement de 4 jetons de la même couleur.

Enfin sur les diagonales allant vers la gauche, pour les lignes 0 et 1 et colonne 4, il n'est pas possible de faire un alignement de 4 jetons de la même couleur.

```
#le dernier jeton placé est-il gagnant
#On teste dans toutes les directions, s'il y a 4 jetons identiques qui se suivent
def coup_gagnant(plateau, jeton):
    #test de toutes les positions horizontales
    for c in range (NB_COLONNE-3):
        for r in range (NB_LIGNE):
            if plateau[r][c]== jeton and plateau[r][c+1]== jeton and plateau[r][c+2]== jeton and plateau[r][c+3]== jeton:
                return True

    #test de toutes les positions verticales
    for c in range (NB_COLONNE):
        for r in range (NB_LIGNE-3):
            if plateau[r][c]== jeton and plateau[r+1][c]== jeton and plateau[r+2][c]== jeton and plateau[r+3][c]== jeton:
                return True

    #test de toutes les positions diagonales orientées vers la droite
    for c in range (NB_COLONNE-3):
        for r in range (NB_LIGNE-3):
            if plateau[r][c]== jeton and plateau[r+1][c+1]== jeton and plateau[r+2][c+2]== jeton and plateau[r+3][c+3]== jeton:
                return True

    #test de toutes les positions diagonales orientées vers la gauche
    for c in range (NB_COLONNE-3):
        for r in range (3,NB_LIGNE):
            if plateau[r][c]== jeton and plateau[r-1][c+1]== jeton and plateau[r-2][c+2]== jeton and plateau[r-3][c+3]== jeton:
                return True
```

Pour tester, ajouter cette dernière fonction et modifier le programme principal comme suit :

```
#=====================================================
plateau = creer_plateau()
afficher_plateau(plateau)
for i in range(7):
    colonne = int(input("Joueur 1 : Entrez la colonne sélectionnée (0-6):"))
    if emplacement_valide(plateau, colonne):
        ligne=trouver_Ligne_Vide(plateau,colonne)
        lacher_jeton(plateau,ligne,colonne,1)
        afficher_plateau(plateau)
        #tester si le joueur 1 a gagné
        if coup_gagnant(plateau,1):
            print("Le joueur 1 a gagné!!! Bravo")
```

Pour arrêter la boucle, il suffit de taper autre chose qu'un chiffre. Cela provoque une erreur qui arrête le programme.

Lancer plusieurs fois le programme pour vérifier qu'il détecte bien le coup gagnant à la verticale et à l'horizontal.

Pour les diagonales, on ne peut pas tester sans ajouter le deuxième joueur.

La boucle de jeu

Les joueurs doivent jouer alternativement. Nous devons donc mémoriser à qui est le tour.

La partie doit s'arrêter lorsque l'un des joueurs a gagné ou que toutes les cases du plateau sont remplies.

Nous ajoutons une fonction qui détecte si la partie est nulle :

```
#Y at'il encore des places vides dans la matrice
def partie_nulle(plateau):
    for c in range (NB_COLONNE):
        for r in range(NB_LIGNE):
            if plateau[r][c] == 0:
                return False
    return True
```

Nous sécurisons aussi l'entrée du numéro de colonne.

```
#=======================================================
plateau = creer_plateau()
game_over = False
tour = 0
afficher_plateau(plateau)
while not game_over:
    #joueur 1
    if tour==0:
        #choix du numéro de colonne par le joueur 1
        correct = False
        while not correct:
            colonne = int(input("Joueur 1 : Entrez la colonne sélectionnée (0-6):"))
            if colonne >= 0 and  colonne <= 6 :
                correct = True
        if emplacement_valide(plateau,colonne):
            #trouver la ligne libre dans la colonne choisie
            ligne = trouver_Ligne_Vide(plateau,colonne)
            #mettre dans la matrice la valeur 1 dans la ligne, colonne
            lacher_jeton(plateau, ligne, colonne,1)
            #tester si le joueur 1 a gagné
            if coup_gagnant(plateau,1):
                print("Le joueur 1 a gagné!!! Bravo")
                game_over = True
        tour = 1
    #joueur 2
    else:
        #choix du numéro de colonne par le joueur 2
        correct = False
```

```
        while not correct:
            colonne = int(input("Joueur 2 : Entrez la colonne sélectionnée (0-6):"))
            if colonne >= 0 and  colonne <= 6 :
                correct = True
        if emplacement_valide(plateau,colonne):
            #trouver la ligne libre dans la colonne choisie
            ligne = trouver_Ligne_Vide(plateau,colonne)
            #mettre dans la matrice la valeur 2 dans la ligne, colonne
            lacher_jeton(plateau, ligne, colonne,2)
            #tester si le joueur 2 a gagné
            if coup_gagnant(plateau,2):
                print("Le joueur 2 a gagné!!! Bravo")
                game_over = True
        tour = 0


    afficher_plateau(plateau)
    if partie_nulle(plateau):
        print("Partie nulle")
        game_over = True
```

Tester les alignements en diagonale.