

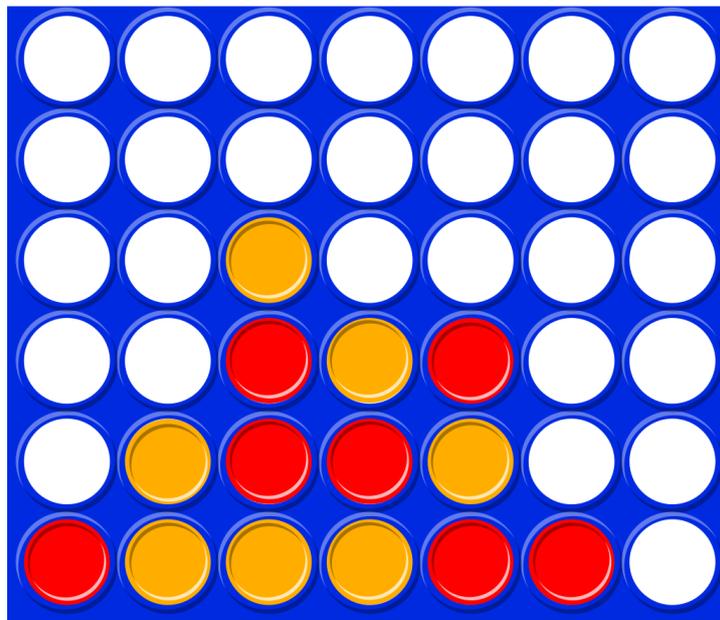


# Le puissance 4

## I- Présentation

Définition wikipédia :

[https://fr.wikipedia.org/wiki/Puissance\\_4](https://fr.wikipedia.org/wiki/Puissance_4)



Puissance 4 est un jeu de stratégie combinatoire abstrait, commercialisé pour la première fois en 1974 par MB et détenue depuis 1984 par la société Hasbro.

### Règles du jeu

Le but du jeu est d'aligner une suite de 4 pions de même couleur sur une grille comptant 6 rangées et 7 colonnes. Chaque joueur dispose de 21 pions d'une couleur (par convention, en général jaune ou rouge). Tour à tour, les deux joueurs placent un pion dans la colonne de leur choix, le pion coulisse alors jusqu'à la position la plus basse possible dans ladite colonne à la suite de quoi c'est à l'adversaire de jouer. Le vainqueur est le joueur qui réalise le premier un alignement (horizontal, vertical ou diagonal) consécutif d'au moins quatre pions de sa couleur. Si, alors que toutes les cases de la grille de jeu sont remplies, aucun des deux joueurs n'a réalisé un tel alignement, la partie est déclarée nulle.

## II- L'interface graphique

On vous fournit une interface graphique à travers la classe `GUIpuissance4` que vous pouvez importer depuis le module fourni. Si le répertoire `GUI_puissance4` est dans le même répertoire que votre fichier :

```
from GUI_puissance4.guiPuissance4 import GUIpuissance4
```

Constructeur de la classe :

- Attribut public : **aucun**.
- Le constructeur ne prend aucun argument. Il permet de créer un objet graphique vierge.

```
GUI = GUIpuissance4()
```

- Les méthodes :

- **refresh(g, t)** : Cette méthode rafraichie l'affichage du jeu de puissance 4 conformément à la grille passée en paramètre.

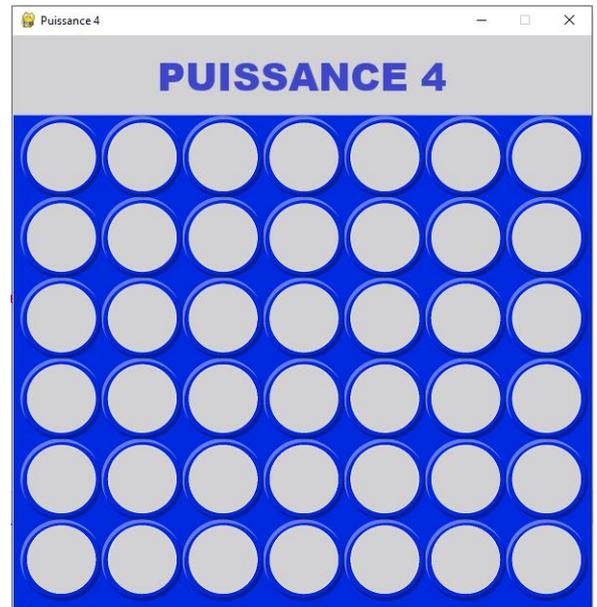
- **g** est une liste de 7 colonnes, `g[0]` est la colonne de gauche, `g[6]` la colonne de droite.

Chaque colonne est une liste de 6 éléments `c[0]` est l'élément du bas, `c[5]` est l'élément du haut.

Par exemple `g[0][0]` est la case en bas à gauche, `g[6][5]` est la case en haut à droite.

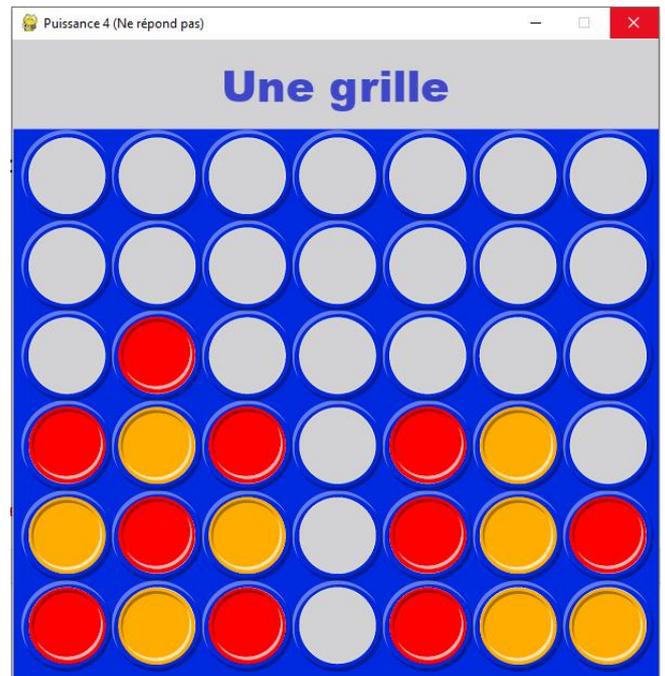
1 désigne une pièce jaune, 2 une pièce rouge, tout autre valeur n'affiche rien !

- **t** est un texte à afficher, centré dans le bandeau du haut.



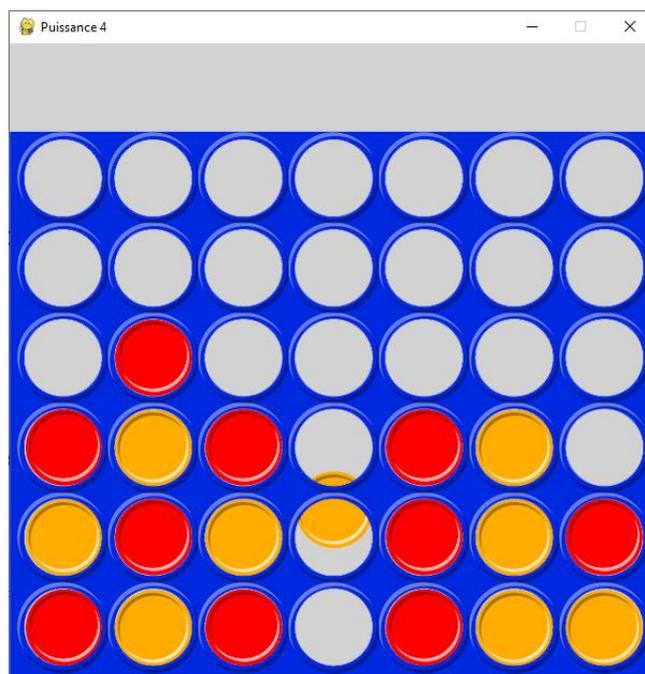
Par exemple :

```
grille = [[2, 1, 2, 0, 0, 0],  
          [1, 2, 1, 2, 0, 0],  
          [2, 1, 2, 0, 0, 0],  
          [0, 0, 0, 0, 0, 0],  
          [2, 2, 2, 0, 0, 0],  
          [1, 1, 1, 0, 0, 0],  
          [1, 2, 0, 0, 0, 0]]  
GUI.refresh(grille, "Une grille")
```



- **refresh(g, t, colonne, couleur, case)** : C'est la même méthode que la précédente mais les trois arguments supplémentaires, si ils sont renseignés, déclenche l'animation d'une pièce jusqu'à la position souhaitée, g n'est pas modifiée.
  - **colonne** est le numéro de la colonne dans laquelle tombe la nouvelle pièce.
  - **couleur** est la couleur de la nouvelle pièce.
  - **case** est le numéro de la case où doit s'arrêter la nouvelle pièce.

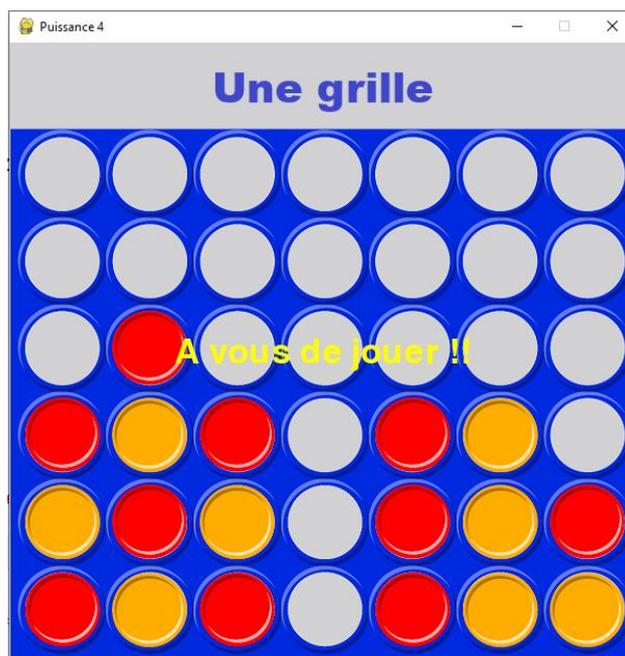
`GUI.refresh(grille, "", 3, 1, 0)`



- **waitClick()** : Cette méthode attend l'action d'un joueur. Elle gère trois types d'actions :
  - **demande fermeture de la fenêtre** : fermeture propre de la fenêtre pygame et fin du programme python.
  - **click sur la fenêtre** : retourne un entier entre 0 et 6 qui correspond à l'indice de la colonne choisie.
  - **appui sur des touches du pad numérique 0 à 6** : retourne l'entier (de type int) correspondant.
  - **appui sur des touches spéciales** :
    - fleche RIGHT : retourne '\_R';
    - fleche LEFT : retourne '\_L';
    - fleche DOWN : retourne '\_D';
    - fleche UP : retourne '\_U';
    - touche BACKSPACE : retourne '\_B';
    - touche RETURN : retourne '\_E';
    - touche ESCAPE : retourne '\_S';
  - **appui sur une autre touche du clavier** : retourne le caractère unicode correspondant.

**Attention**, une fois exécutée, on ne peut sortir de cette méthode que par l'une de ces actions.
- **messCentre(mess)** : Cette méthode permet d'afficher le message mess plein écran, au centre.

*GUI. messCentre("A vous de jouer !!")*



Images sous licence common creative : source **wikipedia**