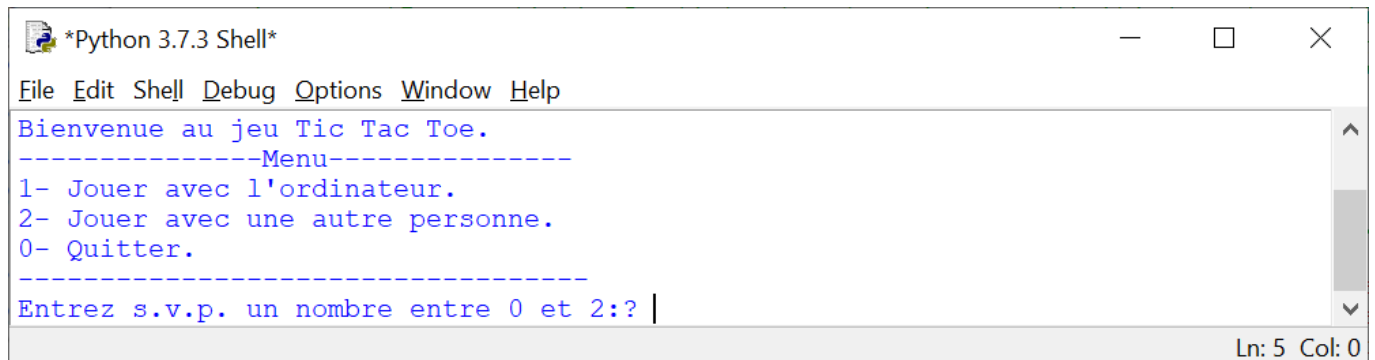


Mini-Projet (travail d'équipe de 2 étudiants maximum)

À remettre, (par email) avant 23h55 le dimanche 05 janvier 2020

Nous vous demandons dans ce mini-projet¹ de programmer un jeu classique de [Tic-Tac-Toe](http://www.calculatorcat.com/games/tic_tac_toe.phtml). Si vous ne connaissez pas ce jeu, vous pouvez l'essayer par exemple sur ce site : http://www.calculatorcat.com/games/tic_tac_toe.phtml . Voici les spécifications du programme que nous vous demandons d'exécuter :

Vous devez faire un programme qui commence par afficher un menu et demande à l'utilisateur d'entrer un chiffre entre 0 et 2. Voici comment le menu devrait être affiché :

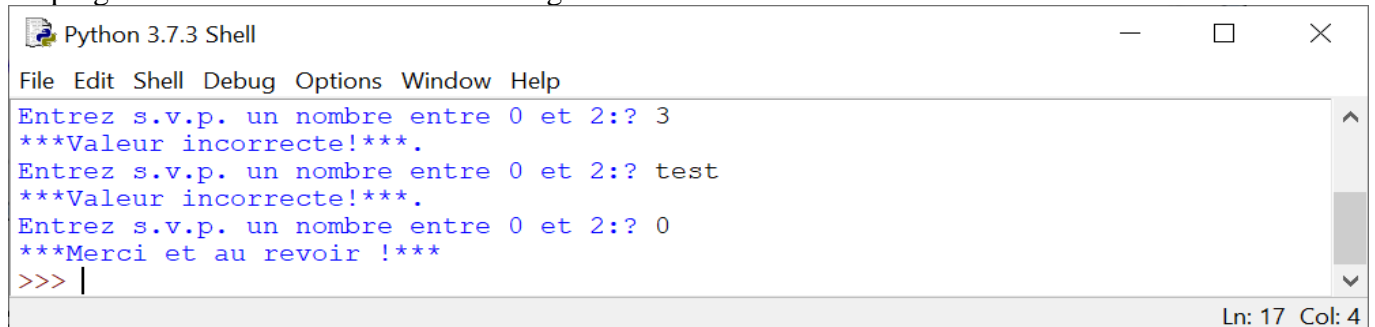


```
*Python 3.7.3 Shell*
File Edit Shell Debug Options Window Help
Bienvenue au jeu Tic Tac Toe.
-----Menu-----
1- Jouer avec l'ordinateur.
2- Jouer avec une autre personne.
0- Quitter.
-----
Entrez s.v.p. un nombre entre 0 et 2:? |
Ln: 5 Col: 0
```

Vous devez afficher le message « ***Valeur incorrecte!*** » si l'utilisateur entre un nombre inférieur à 0 ou supérieur à 2. De plus, vous devez utiliser la méthode `isnumeric()` afin de vous assurer que l'utilisateur entre une valeur numérique et non pas une chaîne de caractères.

Si l'utilisateur entre 0 :

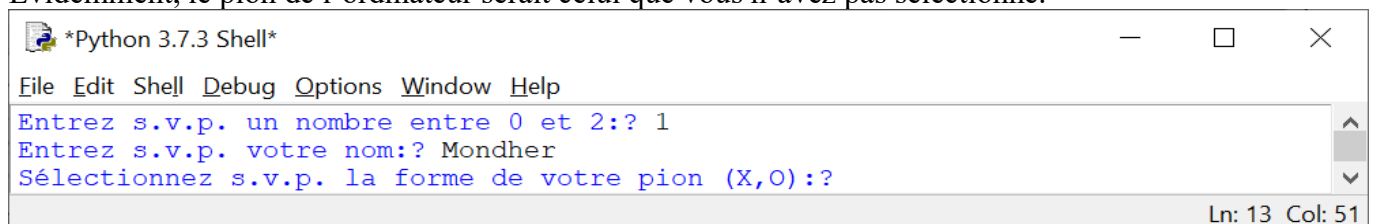
Le programme s'arrête et affiche le message « ***Merci et au revoir !*** »:



```
Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Entrez s.v.p. un nombre entre 0 et 2:? 3
***Valeur incorrecte!***.
Entrez s.v.p. un nombre entre 0 et 2:? test
***Valeur incorrecte!***.
Entrez s.v.p. un nombre entre 0 et 2:? 0
***Merci et au revoir !***
>>> |
Ln: 17 Col: 4
```

Si l'utilisateur entre 1 :

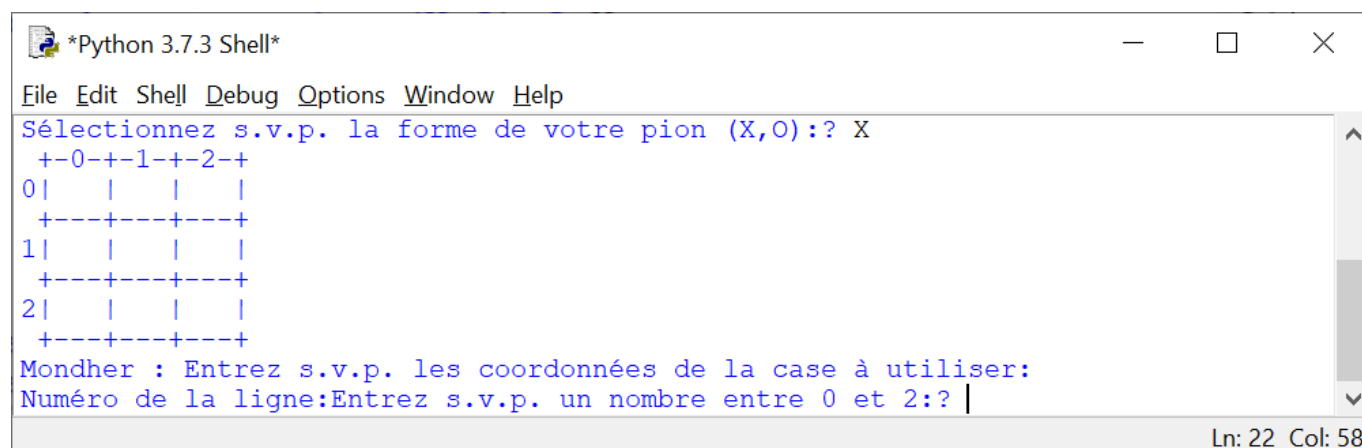
Le programme doit commencer une partie entre vous et l'ordinateur. Il faut entrer votre nom et sélectionner la forme de votre pion ('X' ou 'O'). Le nom de l'ordinateur par défaut est [Colosse](#). Évidemment, le pion de l'ordinateur serait celui que vous n'avez pas sélectionné.



```
*Python 3.7.3 Shell*
File Edit Shell Debug Options Window Help
Entrez s.v.p. un nombre entre 0 et 2:? 1
Entrez s.v.p. votre nom:? Mondher
Sélectionnez s.v.p. la forme de votre pion (X,O):?
Ln: 13 Col: 51
```

¹ Il est conseillé de lire les chapitres 11 et 12 du manuel du cours afin de comprendre les notions de base de l'orienté objet de Python (Classe, objet, etc.).

Le programme doit par la suite afficher le plateau du jeu composé de 9 cases initialement vides (contenant une chaîne de caractères représentant un espace). Les coordonnées d'une case représentent le numéro de la ligne et le numéro de la colonne sur le plateau du jeu. Le programme doit donc demander le numéro de la ligne et le numéro de la colonne de la case que le joueur courant (dans ce cas vous) veut choisir afin de jouer son tour.

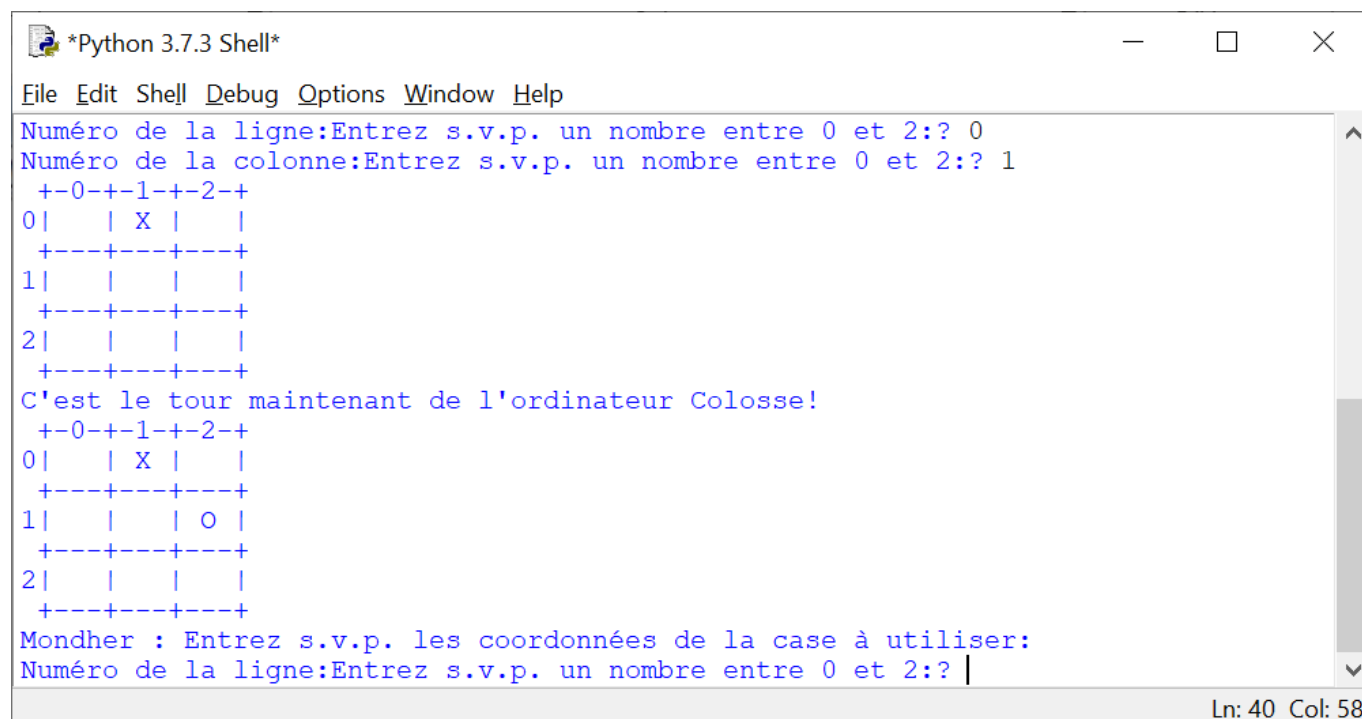


```

*Python 3.7.3 Shell*
File Edit Shell Debug Options Window Help
Sélectionnez s.v.p. la forme de votre pion (X,O):? X
+-0-+-1-+-2-+
0|  |  |  |  |
+---+---+---+
1|  |  |  |  |
+---+---+---+
2|  |  |  |  |
+---+---+---+
Mondher : Entrez s.v.p. les coordonnées de la case à utiliser:
Numéro de la ligne:Entrez s.v.p. un nombre entre 0 et 2:? |
Ln: 22 Col: 58

```

Le programme affiche alors le nouveau plateau incluant le choix du premier joueur en utilisant bien sûr son pion. Ensuite, c'est le tour de l'ordinateur Colosse qui doit choisir la case à jouer. Ce choix doit se faire en fonction de la configuration actuelle du plateau. L'algorithme que vous allez concevoir permettant de faire jouer l'ordinateur n'a pas besoin d'être optimal. Cela permettra à l'adversaire (dans ce cas vous) de gagner de temps en temps. Il faut par contre essayer de mettre le pion de l'ordinateur dans une ligne, une colonne ou une diagonale contenant deux pions de l'adversaire pour que ce dernier ne gagne pas facilement. Il faut aussi essayer de mettre le pion de l'ordinateur dans une ligne, une colonne ou une diagonale contenant deux pions de l'ordinateur pour que ce dernier puisse gagner. Vous pouvez utiliser ici la fonction `randrange()` du module `random`. Par exemple: `randrange(1,10)` vous retourne une valeur entre 1 et 9 au hasard.



```

*Python 3.7.3 Shell*
File Edit Shell Debug Options Window Help
Numéro de la ligne:Entrez s.v.p. un nombre entre 0 et 2:? 0
Numéro de la colonne:Entrez s.v.p. un nombre entre 0 et 2:? 1
+-0-+-1-+-2-+
0|  | X |  |  |
+---+---+---+
1|  |  |  |  |
+---+---+---+
2|  |  |  |  |
+---+---+---+
C'est le tour maintenant de l'ordinateur Colosse!
+-0-+-1-+-2-+
0|  | X |  |  |
+---+---+---+
1|  |  | O |  |
+---+---+---+
2|  |  |  |  |
+---+---+---+
Mondher : Entrez s.v.p. les coordonnées de la case à utiliser:
Numéro de la ligne:Entrez s.v.p. un nombre entre 0 et 2:? |
Ln: 40 Col: 58

```

À la fin de chaque partie, le programme doit afficher les statistiques sur le jeu (le nombre de parties gagnées par chaque joueur ainsi que le nombre de parties nulles). Voici un exemple (il faut respecter ce format d'affichage) :

```

Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
Numéro de la ligne:Entrez s.v.p. un nombre entre 0 et 2:? 2
Numéro de la colonne:Entrez s.v.p. un nombre entre 0 et 2:? 2
+--+--+--+--+
0| X | X | O |
+--+--+--+--+
1|   | X | O |
+--+--+--+--+
2|   | O | X |
+--+--+--+--+

-----
Partie terminée! Le joueur gagnant est: Mondher
Parties gagnées par Mondher : 1
Parties gagnées par Colosse : 0
Parties nulles: 0
Voulez-vous recommencer (O,N)? |
Ln: 90 Col: 31

```

Il faut vous assurer que l'utilisateur ne puisse pas entrer un numéro de ligne ou un numéro de colonne incorrecte. Il faut également que ces coordonnées soit valides (représente une case vide et non pas une case déjà utilisée par un des deux joueurs). De plus, il ne faut pas oublier d'initialiser le plateau avec des cases vides avant de recommencer le jeu. Si l'utilisateur ne veut plus recommencer, il faut arrêter le programme et afficher ce message: ***Merci et au revoir !***.

Si l'utilisateur entre 2 :

Le programme demande le nom du premier joueur ainsi que la forme de son pion. Il demande ensuite le nom du deuxième joueur. Évidemment et comme c'était le cas précédemment, le pion du deuxième joueur serait celui que le premier joueur n'a pas sélectionné.

```

Python 3.7.3 Shell
File Edit Shell Debug Options Window Help
-----Menu-----
1- Jouer avec l'ordinateur.
2- Jouer avec une autre personne.
0- Quitter.
-----
Entrez s.v.p. un nombre entre 0 et 2:? 2
Entrez s.v.p. votre nom:? Mondher
Sélectionnez s.v.p. la forme de votre pion (X,O):? X
Entrez s.v.p. le nom de l'autre joueur:? Siwar
+--+--+--+--+
0|   |   |   |
+--+--+--+--+
1|   |   |   |
+--+--+--+--+
2|   |   |   |
+--+--+--+--+

Mondher : Entrez s.v.p. les coordonnées de la case à utiliser:
Numéro de la ligne:Entrez s.v.p. un nombre entre 0 et 2:?
Ln: 23 Col: 58

```

Voici d'ailleurs un exemple d'une partie nulle entre deux joueurs :

```
*Python 3.7.3 Shell*
File Edit Shell Debug Options Window Help
Mondher : Entrez s.v.p. les coordonnées de la case à utiliser:
Numéro de la ligne:Entrez s.v.p. un nombre entre 0 et 2:? 2
Numéro de la colonne:Entrez s.v.p. un nombre entre 0 et 2:? 1
+--0--1--2--+
0| X | O | X |
+---+---+---+
1| X | X | O |
+---+---+---+
2| O | X | O |
+---+---+---+
-----
Partie terminée! Aucun joueur n'a gagné!
Parties gagnées par Mondher : 0
Parties gagnées par Siwar : 0
Parties nulles: 1
Voulez-vous recommencer (O,N)?
Ln: 117 Col: 31
```

Éléments fournis

La solution est modélisée en 4 classes : une classe **Case** représentant une case du jeu Tic-Tac-Toe, une classe **Plateau** représentant le plateau du jeu, une classe **Joueur** permettant de modéliser les deux joueurs (personne ou ordinateur) et une classe **Partie** modélisant une partie du jeu entre ces deux joueurs.

Nous vous fournissons avec cet énoncé quatre fichiers Python (un fichier par classe est fourni). Les fichiers **case.py** et **joueur.py** contiennent respectivement le code des classes **Case** et **Joueur**. Vous ne devez pas modifier ces fichiers sauf si vous voulez ajouter d'autres méthodes.

Les fichiers **plateau.py** et **partie.py** contiennent respectivement le code des classes **Plateau** et **Partie**. La majorité des méthodes de ces deux classes sont à compléter. Il faut donc remplacer '*pass*' par votre code. La modélisation étant déjà faite pour vous, nous avons inclus en commentaires des informations expliquant ce qu'il y a à faire pour chaque méthode. Il est à noter que vous devez compléter les commentaires (auteurs, date et description) au début de chacun de ces fichiers.

Ce que vous devez rendre

Vous devez envoyer un email en attachant un fichier .zip comportant uniquement les fichiers suivants :

- Les fichiers **case.py**, **joueur.py**, **plateau.py** et **partie.py** avec les modifications demandées.
- Ne remettez aucun autre fichier ou dossier que les 4 fichiers mentionnés ci-haut.

Le nom du .zip doit respecter le format suivant : **mini-projet-CIN-Gr.zip** où CIN est le numéro de la carte d'identité nationale de la personne qui va envoyer le travail par email (**un seul travail par équipe doit être envoyé**, sinon vous risquez d'avoir une pénalité) et Gr est le numéro de votre groupe (exp : Gr-01-02). De plus, il est de votre responsabilité de vérifier que vous nous avez envoyé les bons fichiers (**pas vides et non corrompus**), sinon vous pouvez avoir un zéro. Voici l'email qu'il faut utiliser : parad.proga1@gmail.com

Important : Le code doit être interprété sans erreurs et sans *warnings*, sinon vous pouvez être fortement pénalisé. Le plagiat est interdit. Nous avons d'ailleurs des outils permettant de détecter la ressemblance de code de deux équipes différentes.

Bon travail !