Réalisez un projet en Python : Sudoku*

Jilles S. Dibangoye
jilles-steeve.dibangoye@insa-lyon.fr

Préambule: Ceci est un projet individuel. Le format du fichier attendu est le suivant : "sudoku_nom_prenom.py". Le fichier de votre projet (.py) est à déposer sur Moodle avant le 04 octobre à 23h59. Merci de testez le dépôt pour vous assurez qu'il fonctionne.

Compétences ciblées

Être capable de :

1. écrire une classe en Python
2. manipulez des fichiers en Python
3. écrire une programme modulaire en Python

1 Programmation : résolution de Sudoku

Une grille de Sudoku est une grille de taille 9×9, subdivisée en 9 carrés 3×3 appelés *régions*. Les cases de cette grille peuvent recevoir des chiffres. Le jeu de Sudoku consiste à completer une grille partiellement remplie par des chiffres en respectant la condition suivante : chaque ligne, colonne et region ne doit contenir qu'une seule fois tous les chiffres de un à neuf.

Un exemple de grille est représenté sur la gauche de la Figure 1, la grille complétée est représentée sur la droite de la même Figure. On peut vérifier que sur la grille de droite, chaque ligne contient tous les chiffres une et une seule fois, il en est de même pour chaque colonne et chaque région.

1.1 Créez une classe Sudoku

Question 1. Écrire une classe Python, nommée `Sudoku`, qui permet d'encoder une grille Sudoku (complète ou incomplète).

Indication : Vous pourrez utiliser un tableau bi-dimensionnel d'entiers de taille 9×9 pour stocker une grille de Sudoku.

*crédit: Tanguy Risset – *cf.* TP AGP

1.2 Renseignez une instance de Sudoku via un fichier

Afin de tester vos méthodes, nous vous fournissons un fichier `sudoku.txt` [1] contenant des *grilles* de Sudoku. Chaque ligne du fichier correspond à une grille de Sudoku. Les grilles sont simplement codées à l'aide de 81 caractères qui correspondent aux valeurs initiales des cases. Les premiers 9 caractères correspondent à la première ligne de la grille, les 9 suivants à la deuxième ligne etc. Par exemple, la ligne du fichier

```
790304108000006000000080059030000497000000000217000030470010000000700000302609071
```

correspond à la grille de Sudoku :

7	9		3		4	1		8
					6			
				8			5	9
		3				4	9	7
2	1	7					3	
4	7			1				
			7					
3		2	6		9		7	1

Question 2. — Écrivez une fonction `lire_fichier_sudoku(fichier, ligne)` qui permet de lire une grille de Sudoku à partir d'un fichier.

1.3 Renseignez une instance de Sudoku de façon interactive

Question 3. — Écrivez une fonction `saisir_sudoku()` qui permet à l'utilisateur de renseigner une grille de Sudoku en saisissant 81 caractères. Gérez les exceptions.

1.4 Affichez une instance de Sudoku

Question 4. — Écrivez une fonction `afficher_sudoku()` afin d'afficher une grille de Sudoku.

Indication : Vous pourrez vous inspirer de l'affichage ci-dessous

1. Téléchargez le fichier sudoku.txt sur moodle.

```
 -------------
|790|304|108|
|000|006|000|
|000|080|059|
-------------
|030|000|497|
|000|000|000|
|217|000|030|
-------------
|470|010|000|
|000|700|000|
|302|609|071|
-------------
```

1.5 Vérifiez une solution

On dira qu'un Sudoku est *valide* s'il est complété et satisfait l'ensemble des contraintes décrites plus tôt.

Question 5. — Écrire une fonction Python, nommée `est_valide()`, dans la classe `Sudoku`, qui vérifie qu'une grille complètement remplie est un Sudoku valide. La fonction renvoie 1 si le Sudoku est valide, et 0 sinon.

Indication : Considérez trois fonctions de test de validité d'une grille Sudoku selon : la région, la ligne, et la colonne.

	2		5		1		9	
8			2		3			6
	3			6			7	
		1				6		
5	4						1	9
		2				7		
	9			3			8	
2			8		4			7
	1		9		7		6	

Sudoku à compléter

4	2	*6*	5	*7*	1	*3*	9	*8*
8	*5*	*7*	2	*9*	3	*1*	*4*	6
1	3	*9*	*4*	6	*8*	*2*	7	*5*
9	*7*	1	*3*	*8*	*5*	6	*2*	*4*
5	4	*3*	*7*	*2*	*6*	*8*	1	9
6	*8*	2	*1*	*4*	*9*	7	*5*	*3*
7	9	*4*	*6*	3	*2*	*5*	8	*1*
2	*6*	*5*	8	*1*	4	*9*	*3*	7
3	1	*8*	9	*5*	7	*4*	6	*2*

Sudoku complété

FIGURE 1 – Un exemple de grille Sudoku avec sa solution.

1.6 Résolvez des Sudoku simples

Pour cette partie, nous allons supposer que nous n'avons que des *grilles simples* de Sudoku à résoudre—la définition d'une grille de Sudoku simple sera donnée plus loin. On se propose d'implémenter une méthode Python pour résoudre les Sudoku simples.

		4		3	6			5
		5	1				8	
1	3	7	8		4	6		2
	5		4				3	6
4	9			7	3	2		8
		6		8		9	7	
8		9	7	6	5			1
6		2	3				4	
			9			8		7

FIGURE 2 – Exemple d'une grille Sudoku simple.

1. Pour chaque chiffre `i` présent dans une case `c`, on mémorise le fait que ce chiffre `i` ne peut pas être mis dans une case présente sur la même ligne que celle de `c` ou sur la même colonne que celle de `c` ou sur la même région que celle de `c`.
2. Compte-tenu de ces instructions, on cherche une case vide pour laquelle il existe une *unique* valeur permise.
3. Si on ne trouve pas une telle case l'algorithme échoue (le Sudoku n'est pas *simple*).
4. Si on trouve une telle case : on la remplit, on mémorise les nouvelles interdictions introduites par le chiffre qu'on a rajouté et on repart en 2.

Un `Sudoku` sera donc dit *simple* si cette méthode permet de le résoudre, c'est à dire qu'à chaque étape, il existe une case pour laquelle une seule valeur est possible. La grille de la Figure 2 est une `Sudoku` simple. La Figure 3 montre les trois premières étapes possibles pour commencer le remplissage de la grille :

1. la case de coordonnées $(8,5)$ ne peut contenir que la valeur 1,
2. ensuite, la case de coordonnées $(8,7)$ ne peut contenir que la valeur 5,
3. ce qui entraîne que la case de coordonnées $(7,7)$ ne peut contenir que la valeur 3,
4. etc.

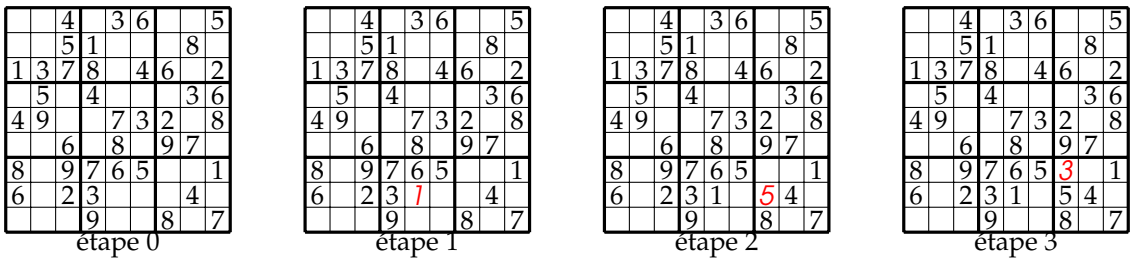

FIGURE 3 – Trois première cases de la grille Sudoku *simple*, remplies par notre méthode. Notez qu'il y'a d'autres choix possibles à chaque étape.

Question 6. — Écrire une méthode de la classe `Sudoku`, qui prend en entrée une grille partiellement remplie, et la remplit complètement avec cette méthode (ou échoue si le Sudoku n'est pas simple).

Indication : Pour stocker les interdictions, on pourra par exemple utiliser un tableau `M` d'entiers de dimension 3, chaque dimension étant de taille 9. La case `M`$[i][j][k]$ vaudra 1 si le chiffre $k+1$ ne peut pas être ecrit dans la case de coordonnées $(i+1, j+1)$ et 0 sinon (les tableaux en Python commençant à 0, il faut ajouter 1 pour avoir des valeurs entre 1 et 9). Il n'y a pas obligation d'utiliser cette structure de données, vous pouvez proposer une autre structure de données pour écrire cette fonction.

2 Concevez un algorithme de résolution de Sudoku

Question 7. — Implémentez en Python une fonction `resoudre()` de résolution d'une grille de Sudoku quelconque (par forcément simple).

Question 8. — Écrire un programme interactif principal en Python permettant à l'utilisateur de :

1. résoudre une grille Sudoku tirée aléatoirement parmi celles à disposition dans le fichier des grilles Sudoku.
2. proposez une solution partielle de la grille Sudoku, si l'utilisateur le souhaite.
3. proposez une solution complète de la grille Sudoku, si l'utilisateur le souhaite.