Informatique en CPGE (2014-2015)
Exercices : le sudoku

Un tableau sera représenté par une liste de listes (les lignes du tableaux).

Par exemple, pour créer le tableau T = $\begin{array}{|c|c|c|}\hline 3&1&5\\\hline 1&2&0\\\hline 2&6&4\\\hline\end{array}$ (c'est un tableau 3x3),

on écrira : T=[[3,1,5],[1,2,0],[2,6,4]]

Pour jouer au **Sudoku**, on prend un tableau 9x9, partagé en 9 sous-tableaux 3x3, dont les cases ne contiennent que les chiffres de 0 à 9. On doit remplacer les 0 par des chiffres de 1 à 9 de manière à ce que :

- chaque chiffre de 1 à 9 n'apparait qu'une fois sur chaque ligne ;
- chaque chiffre de 1 à 9 n'apparait qu'une fois sur chaque colonne ;
- chaque chiffre de 1 à 9 n'apparait qu'une fois dans chacun des neuf sous-tableaux 3x3.

On ne modifie pas les chiffres non nuls du tableau donné.

Voici un exemple de jeu :

0	0	3	0	2	0	6	0	0
9	0	0	3	0	5	0	0	1
0	0	1	8	0	6	4	0	0
0	0	8	1	0	2	9	0	0
7	0	0	0	0	0	0	0	8
0	0	6	7	0	8	2	0	0
0	0	2	6	0	9	5	0	0
8	0	0	2	0	3	0	0	9
0	0	5	0	1	0	3	0	0

L'objectif est d'écrire un programme qui trouve la solution.

1. On commence par définir une fonction qui détermine le premier terme nul du tableau s'il y en a.

   Ecrire une fonction **zero(T)** qui prend en argument un tableau T, le parcourt ligne par ligne de gauche à droite et renvoie le couple d'indices (i , j) du premier terme nul ou $(-1\ ,\ -1)$ s'il n'y a aucun terme nul.

2. On définit ensuite une fonction qui vérifie si on peut remplacer un 0 par un chiffre c.

   Ecrire une fonction **test(x,y,c,T)** qui prend en argument deux indices x et y, (les indices d'une case contenant un 0), un chiffre c et le tableau T. Cette fonction teste si le chiffre c se trouve sur la ligne x et renvoie False si c'est le cas ; ensuite la fonction teste si le chiffre c se trouve sur la colonne y et renvoie False si c'est le cas ; enfin la fonction teste si le chiffre c se trouve dans le sous-tableau 3x3 contenant la case d'indice (x,y) et renvoie False si c'est le cas, (c'est le plus difficile) ; si les trois tests ne sont pas False, la fonction renvoie True (donc on peut remplacer le 0 par le chiffre c).

3. Il s'agit maintenant d'écrire une fonction **sudoku(T)** qui va résoudre le problème. Cette fonction sera définie de manière récursive.

   (a) On récupère les indices x et y du premier 0 s'il existe.

   (b) Si $x = -1$, il n'y a pas ou plus de 0, c'est le test d'arrêt : on affiche le tableau.

(c) Sinon : on teste si on peut remplacer le 0 par un des chiffres de 1 à 9 ; si c'est le cas on le fait et on appelle alors la fonction **sudoku** avec le nouveau tableau ; si ce n'est pas le cas, c'est qu'il faut revenir en arrière et on réaffecte la valeur 0 à la case d'indices (x,y).

4. Tester le programme avec le tableau donné ci-dessus.
5. On souhaite maintenant récupérer la solution dans une variable. Faire des essais en plaçant une instruction **return** dans la fonction **sudoku**.

   Conclusion ?
6. Si on y regarde de plus près, on peut constater que le programme continue à fonctionner après avoir afficher la solution. Une manière de récupérer la solution est de créer une variable **solution** (une liste vide), de la déclarer **global** au début de la fonction **sudoku**, puis dans le test d'arrêt, (donc lorsque T est égal à la solution), de copier T dans la variable **solution**. Attention faire une "vraie" copie, par exemple terme à terme.