SUDOKU

1 Le jeu

Le sudoku (du japonais *sû*, chiffre, et de *doku*, unique) est un jeu en forme de grille carrée de taille $n \times n$ avec $n \geq 2$. En général $n = 3$ et la grille est composée de 9 lignes, 9 colonnes et subdivisée en 9 régions ou sous-grilles de taille 3×3 (que l'on peut visualiser en figure 1 ci-dessous). Le but du jeu est de compléter une grille partiellement remplie avec des chiffres allant de 1 à 9, de sorte que chaque ligne, chaque colonne et chaque région, contienne une et une seule fois chacun des 9 chiffres. En figure 1, on donne un exemple de grille à résoudre (à gauche) et sa solution (à droite).

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

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

FIGURE 1 – Une grille de sudoku et sa solution

Dans la presse, on trouve des grilles classées selon leur degré de difficulté et admettant une solution unique. On ne considère ici que les sudoku dits faciles : il s'agit de sudoku que l'on peut résoudre en appliquant une règle de remplissage qui sera détaillée au paragraphe 3. Le travail demandé est de programmer la méthode de résolution utilisant cette règle.

2 Définitions

2.1 Cases voisines

Deux cases de la grille sont dites *voisines* si elles se trouvent toutes deux sur une même ligne, sur une même colonne ou dans une même région. La contrainte du jeu stipule que deux cases voisines doivent contenir des chiffres différents.

2.2 Cases ouvertes / cases fermées

Un case de la grille qui contient un chiffre est dite *fermée*, dans le cas contraire, elle est dite *ouverte*. Le but du jeu est donc de fermer toutes les cases ouvertes d'une grille en respectant les contraintes de remplissage.

2.3 Notion de candidat

Un chiffre k est dit *candidat* d'une case c si c est ouverte et si aucune voisine fermée de c ne contient le chiffre k. En figure 2 ci-dessous, on donne les candidats de chaque case ouverte de la grille de la figure 1. On peut constater que la case $(0, 0)$ admet les candidats 5, 6 et 9, alors que la case $(0, 1)$ n'admet qu'un seul candidat, le chiffre 9.

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

FIGURE 2 – Grille des candidats de chaque case ouverte

3 Méthode de résolution de grilles faciles

3.1 Candidat unique

Considérons une case ouverte (i, j) où i est le numéro de ligne de la grille et j le numéro de colonne. Un candidat k d'une case ouverte (i, j) est dit *unique* si et seulement si une au moins des quatre conditions suivantes est vérifiée :

(i) Le chiffre k est l'unique candidat de la case (i, j).

(ii) Le chiffre k n'est pas un candidat d'une autre case ouverte de la ligne i.

(iii) Le chiffre k n'est pas un candidat d'une autre case ouverte de la colonne j.

(iv) Le chiffre k n'est pas un candidat d'une autre case ouverte de la région contenant la case (i, j).

Regardons la grille des candidats de la figure 2 :

- Les cases $(0,1)$, $(1,3)$ et $(2,8)$ possèdent chacune un unique candidat. Le chiffre 9 est un candidat unique pour chacune de ces cases (condition (i)).
- Le chiffre 6 est un candidat unique de la case $(1,6)$ car 6 n'est candidat d'aucune autre case de la ligne 1 (condition (ii)).
- Le chiffre 5 est un candidat unique de la case $(0,0)$ car 5 n'est candidat d'aucune autre case de la colonne 0 (la condition (iii)).
- Le chiffre 4 est un candidat unique de la case $(0,6)$ car 4 n'est candidat d'aucune autre case de la région II (condition (iv)).
- Le chiffre 1 est un candidat unique de la case $(1,1)$, il vérifie les conditions (ii), (iii) et (iv).

3.2 Méthode des candidats uniques

Si une case ouverte (i,j) admet un candidat unique k alors on peut la remplir avec le chiffre k. On doit ensuite supprimer k de l'ensemble des candidats de chaque voisine ouverte de la case (i,j).

Considérons, par exemple, le candidat unique 5 de la case $(0,0)$. Si on remplit cette case avec le chiffre 5, le candidat 5 devra ensuite être éliminé de l'ensemble des candidats de chaque case ouverte de la ligne 0, de chaque case ouverte de la colonne 0 et de chaque case ouverte de la région 0. Cette élimination fait alors apparaître deux nouveaux candidats uniques : le 6 en $(0,2)$ et le 6 en $(8,0)$.

La méthode de résolution consiste donc à fermer, l'une après l'autre, toutes les cases ouvertes admettant un candidat unique. Les grilles les plus simples peuvent être résolues de cette façon, c'est le cas de la grille de la figure 2 sur laquelle on pourra s'exercer.

Si cette méthode ne permet pas de résoudre une grille, on dira que celle-ci est difficile. Dans ce cas, la méthode permet seulement d'éliminer tous les candidats uniques et de diminuer ainsi le nombre de cases ouvertes.

4 Programmation

4.1 Idée générale

Le programme à réaliser devra permettre de résoudre les grilles dites faciles et d'afficher leur solution. Si une grille est difficile, le programme affichera les candidats des cases ouvertes restantes. Il n'est pas demandé de réaliser d'interface graphique (mais vous êtes libres de le faire si vous le souhaitez et en avez le temps). Les affichages se feront en mode texte dans une fenêtre terminal.

4.2 Les grilles

Chaque grille à résoudre est enregistrée dans un fichier texte selon le format suivant : chaque ligne contient trois entiers, le premier est un numéro de ligne i (compris entre 0 et 8), le deuxième est un numéro de colonne j (compris entre 0 et 8) et le troisième est le chiffre k que contient la case (i,j). Les 3 premières lignes du fichier contenant la grille de la figure 2 sont les suivantes :

```
0 5 1
0 7 8
1 0 2
etc.
```

4.3 Les types à utiliser impérativement

▷ `Case`, une structure à 2 champs : le numéro de ligne (`x`) et le numéro de colonne (`y`) d'une case.

```
typedef struct {
     int x; // numéro de ligne
     int y; // numéro de colonne
} Case;
```

▷ `Cand`, une structure à 2 champs : un entier (`nbc`) et une table d'entiers (`tab`) à gérer de façon dynamique. Cette structure sera utilisée pour mémoriser les candidats d'une case. Si une case est ouverte, le champ `nbc` sera le nombre de candidats et ceux-ci seront stockés dans la table `tab`. Si une case est fermée alors `nbc` sera mis à `0` et `tab` sera mis à `NULL`.

```
typedef struct {
     int nbc;   // nombre de candidats
     int * tab; // table des candidats
} Cand;
```

4.4 Les principales variables à manipuler

▷ `int G[9][9]` : un tableau à 2 dimensions pour mémoriser une grille. Un élément (i, j) de ce tableau contiendra soit un chiffre si la case est fermée, soit `0` si la case est ouverte.

▷ `Cand C[9][9]` : un tableau à 2 dimensions permettant de mémoriser les candidats de chaque case de la grille. Pour une case ouverte `(i,j)`, on aura `C[i][j].nbc==0` et `C[i][j].tab==NULL`.

▷ `int NBO` : le nombre de cases ouvertes.

▷ `Case O[81]` : un tableau à une dimension pour mémoriser les cases ouvertes.

4.5 Les principales fonctions

Le problème devra être décomposé en tâches et chaque tâche donnera lieu à une fonction. Parmi les fonctions à écrire on devra trouver :

▷ `lireGrille` : fonction qui lit une grille dans un fichier texte et initialise le tableau `G`.

▷ `ecrireGrille` : une fonction qui affiche une grille `G` à l'écran.

▷ `estCandidat` : une fonction qui indique si un chiffre donné est candidat pour une case donnée.

▷ `initTab` : une fonction qui permet d'initialiser les tableaux `C` et `O`.

▷ `ecrireCand` : une fonction qui affiche les candidats des cases ouvertes à l'écran.

▷ `admetUnique` : une fonction qui retourne le candidat unique d'une case donnée si il en existe un ou la valeur 0 sinon.

▷ `fermerCase` : une fonction qui remplit une case de la grille avec son candidat unique puis qui élimine ce candidat des listes de candidats des cases voisines.

▷ `fermerGrille` : une fonction qui réalise le remplissage d'une grille par la méthode des candidats uniques.

5 Modalités

Le projet est à faire en trinôme (3 étudiants d'un même groupe de TD).

L'évaluation se fera en salle TP sur les ordinateurs du département. Lors de la soutenance, d'une durée de 30 minutes, la présence du trinôme est obligatoire. À l'issue de cette soutenance, vos fichiers sources devront être envoyés par mail à l'enseignant qui vous évalue.

Date de soutenance : jeudi 19 janvier 2017.

Un planning des créneaux horaires sera affiché le lundi 16 janvier.

N'oubliez pas de vous inscrire !

Des grilles pour tester votre projet

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

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

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

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

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

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