Polytech-Sophia
Xidian University

lundi 25 octobre 2018
Durée : 2 mois

Projet C : Sudoku

Avant de commencer : la qualité des commentaires, avec notamment la présence des antécédents, des conséquents, des invariants de boucle, les rôles de chacune des fonctions, ainsi que les noms donnés aux variables, l'emploi à bon escient des majuscules et la bonne indentation rentreront pour une part importante dans l'appréciation du travail. Ce projet doit permettre de montrer votre autonomie et votre compréhension tant dans la conception du programme que dans sa réalisation. Enfin, si les codes de plusieurs projets se trouvent être identiques, ou être copiés depuis le web, tous les projets concernés seront immédiatement sanctionnés par un zéro.

1 Travail à réaliser

L'objectif de ce projet est de programmer le jeu bien connu du sudoku. Le sudoku se joue sur une grille 9×9, formée de 9 sous-grilles 3×3. Le but du jeu est de remplir la grille avec une série de chiffres de 1 à 9, tous différents, qui ne se trouvent jamais *plus d'une fois* sur une même ligne, dans une même colonne ou dans une même sous-grille.

Au départ, la grille dispose d'un certain nombre de chiffres, ce qui autorise une résolution progressive du problème complet. Selon le nombre de chiffres et leur disposition initiale, on définit des niveaux différents de complexité de résolution des grilles.

Une lecture des sites `fr.wikipedia.org/wiki/Sudoku` ou `fr.wikipedia.org/wiki/Mathematiques_du_Sudoku` vous sera utile.

2 Sujet

Votre programme de sudoku disposera d'un interface *textuelle*, c'est-à-dire que les données seront saisies sur l'entrée standard et les résultats seront affichés sur la sortie standard. Il devra offrir les fonctionnalités décrites ci-dessous.

2.1 Définition d'une grille

La grille de sudoku sera représentée par une matrice 9×9. Elle sera initialisée par des valeurs *fixées directement dans votre programme*. Vous pourrez récupérer facilement des exemples grilles de complexité variée sur le web.

2.2 Choix d'un chiffre

L'utilisateur propose un chiffre à insérer dans la grille sur l'entrée standard en indiquant les coordonnées de la case dans la grille (numéro de la ligne et de la colonne) suivies du chiffre à placer. Vous utiliserez la syntaxe *ligne,colonne :chiffre*. Par exemple,

```
3,2:9
```

indique que l'on veut place le chiffre 9 dans la case de la troisième ligne et de la deuxième colonne.

2.3 Contrôle de validité

L'utilisateur essaye de résoudre la grille de sudoku *progressivement* en insérant dans les cases de la grille les chiffres qu'il pense valides. Votre application proposera une vérification *au fur et à mesure* (*i.e.* après chaque coup) des chiffres proposés. Un message d'erreur signalera tout choix incorrect. Attention, le contrôle de validité vérifie seulement s'il est possible de placer le chiffre dans la case choisie, et non pas si le chiffre est à la bonne place (sa place finale).

2.4 Affichage

Votre programme permettra une visualisation de la grille après chaque choix *valide* d'un chiffre par l'utilisateur. Lorsque la grille est entièrement remplie, elle est donc valide, vous l'indiquerez par un message de félicitation.

2.5 Exemple

Au départ, votre programme pourra affichera la grille à trouver :

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

Ensuite, il lit les commandes de placements des chiffres. Ci-dessous, le joueur essaye de placer le chiffre 3 en $(7, 3)$, puis le chiffre 2 en $(1, 9)$. Le placement du premier chiffre est possible, et pas le second.

```
> 7,3:3
 3 . . . . 1 7 . .
 . . 4 . . . . . 6
 . . 2 . . . . 8 .
 . . . . . . . . .
 . . . 6 8 . 9 . 5
 2 9 . . . . . 4 .
 . 1 3 . . . . . .
 . . 8 . 6 . 3 . 2
 . . 6 . 1 3 . . .

> 1,9:2
placement invalide
 3 . . . . 1 7 . .
 . . 4 . . . . . 6
 . . 2 . . . . 8 .
 . . . . . . . . .
 . . . 6 8 . 9 . 5
 2 9 . . . . . 4 .
 . 1 3 . . . . . .
 . . 8 . 6 . 3 . 2
```

```
 . . 6 . 1 3 . . .

>
```

Votre programme doit déceler la fin de partie. Si tous les chiffres sont correctement placés, le sudoku est résolu et le programme écrit un message de félicitation.

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

> 1,3:9
 3 6 9 8 5 1 7 2 4
 8 5 4 3 2 7 1 9 6
 1 7 2 9 4 6 5 8 3
 6 8 5 7 9 4 2 3 1
 4 3 1 6 8 2 9 7 5
 2 9 7 1 3 5 6 4 8
 5 1 3 2 7 8 4 6 9
 7 4 8 5 6 9 3 . 2
 9 2 6 4 1 3 8 5 7

> 8,8:1
Bravo, sudolu résolu !
```

3 Remise du projet

Votre projet est à faire en *binôme*. Il est à rendre au plus tard :

Le 5 janvier 2019, minuit

1. vous enverrez à vos enseignants `dmei2006@xidian.edu.cn`. `jyliu@xidian.edu.cn` une archive `sudoku-n1-n2.tar.gz` avec `n1` et `n2` sont les deux derniers chiffres de numéro d'étudiant
2. vous enverrez à `vg@unice.fr` un rapport en français de 2 pages (pas moins, pas plus) qui décrit votre projet.

3.1 L'archive devra contenir :

— votre fichier source `.c` correctement documenté (chaque fonction doit avoir un commentaire, les invariants de boucle doivent être marqués), indenté, et codé (les noms de variables explicites, éviter les trop longues fonctions) ;
— un fichier `Documentation` au format *pdf* et décrivant le fonctionnement général du programme, les algorithmes, ainsi que les choix de programmation ;
— la compilation avec les options `-Wall -pedantic` ne doit pas donner de *warning*.

Bon travail et bon courage