Sudoku

Objectif du projet (6h TD, 4h TDAO, 14h TP)

L'objectif du projet est de mettre en œuvre les notions et outils de la programmation orientée objet (classes, algorithmes, tableaux, listes, fichiers, interfaces graphiques) sur un problème concret et original donnant lieu à la réalisation d'un petit logiciel.

Les parties du projet abordées en TD constituent le socle *minimum* du projet. Il vous appartient ensuite de poursuivre votre projet par du travail personnel dans la direction de votre choix. Quelques pistes d'extensions sont données à titre d'exemple à la fin du sujet.

I Cadrage du projet

1 Contraintes et attendus

Les contraintes suivantes sont imposées :

1. le projet doit être réalisé en binôme (ou seul avec l'accord de votre enseignant de TP) ;
2. le projet doit être réalisé en Java avec Netbeans ;
3. le projet doit utiliser les notions et outils abordés dans le module ;
4. le projet doit comporter une interface permettant de tester rapidement les principales fonctionnalités du logiciel réalisé. Toute fonctionnalité non accessible par l'interface ne sera pas prise en compte.

Le projet doit être rendu sous forme d'une archive compressée **zip** contenant :

- tous les fichiers du projet `NetBeans` (répertoire complet du projet `NetBeans`) ;
- un rapport en pdf (voir plus loin).

Le fichier **zip** doit s'intituler `NOM1-NOM2.zip` et doit être déposé sur **moodle**. La date butoir pour rendre le projet est indiquée sur **moodle**.

2 Évaluation

L'évaluation du projet tient compte des éléments suivants :

1. qualité du rapport écrit ;
2. qualité de la présentation orale ;
3. comportement en séance de TP (préparation, implication, sérieux, etc.) ;

4. qualité de la programmation : lisibilité du programme (utilisation de noms pertinents pour les classes, les attributs, les méthodes et les variables), structure du programme (découpage en différentes classes, répartitions des traitements dans les différentes classes, utilisation et ré-utilisation des méthodes), présence de tests ;

5. utilisation adaptée des notions abordées dans le module : structures de données (tableaux, listes, arbres, récursivité, etc.), fichiers et interfaces graphiques ;

6. résultats obtenus : étendue des fonctionnalités programmées, justesse des résultats, absence de bugs, etc.

3 Contenu du rapport

Un rapport d'une dizaine de pages doit accompagner le projet `NetBeans` et doit permettre à un informaticien ne connaissant pas votre sujet de comprendre le travail effectué. Par exemple, un autre groupe d'étudiant devrait être capable de poursuivre le projet.

La rapport doit présenter la démarche suivie pendant la réalisation de votre projet. Le plan pourra s'inspirer des grandes étapes de développement d'un logiciel :

1. **analyse des besoins et spécifications fonctionnelles :** cette phase a pour but de délimiter le périmètre du logiciel et de décrire les fonctionnalités qu'il doit proposer aux utilisateurs ; on pourra présenter l'ensemble des besoins utilisateurs sous forme de *user stories*, puis détailler une user story importante (décrire le scénario utilisateur sous forme d'un organigramme, proposer une maquette de l'interface graphique et apporter des précisions sur son déroulement et les éventuelles contraintes à prendre en compte) ;

2. **conception architecturale :** cette étape a pour objectif de concevoir l'organisation interne du logiciel. On pourra lister les différentes classes à réaliser puis les décrire ainsi que leurs relations sous la forme d'un diagramme de classes UML ;

3. **conception détaillée :** cette phase s'intéresse à la conception des classes principales du projet ; pour simplifier, on pourra se focaliser sur la présentation d'un algorithme participant à la réalisation d'une fonctionnalité importante du logiciel ;

4. **codage et tests :** il s'agit dans cette étape de coder toutes les classes du logiciel et de tester une à une les méthodes ; on pourra décrire les problèmes rencontrés durant cette phase et donner quelques exemples de tests réalisés.

5. **tests d'acceptation et recette :** cette dernière phase consiste à vérifier que le le logiciel remplit les fonctionnalités attendues et répond aux attentes des utilisateurs ; on pourra présenter d'un exemple d'utilisation du logiciel avec des captures d'écran.

Le rapport pourra également inclure une conclusion présentant un bilan et une analyse critique du travail effectué, les limites du logiciel, ainsi que les perspectives d'amélioration ou d'extension. Dans le bilan, il est possible d'inclure un aspect de « répartition des tâches » si le projet a été réalisé en binôme.

II Le Sudoku

1 Origines du Sudoku

Le nom Sudoku est né de l'abréviation de la règle du jeu japonais « Sūji wa dokushin ni kagiru », signifiant littéralement « Chiffre limité à un seul » (sous-entendu par case et par ligne). Cependant, le jeu n'est pas particulièrement d'origine japonaise.

On peut en effet faire remonter son histoire jusqu'à la Chine antique, vers −650, où le jeu du carré magique fut créé par des mathématiciens chinois. D'ordre 3, il pouvait alors être représenté par différents symboles et utilisé dans le feng shui. Ce type de concept a ensuite voyagé et s'est transformé, tout d'abord chez les Indiens, qui le limitèrent à des chiffres, puis chez les Arabes vers le VII^{e} siècle. Mais ce n'est qu'en 1979 que les règles modernes du Sudoku furent établies par l'américain Howard Garns, et reprisent largement dans la presse.

La grille de jeu, présentée ci-dessous à titre d'exemple, est un carré de 9 cases de côté, subdivisée en autant de sous-grilles carrées identiques, appelées « régions », « carrés » ou « blocs ».

La règle générique du jeu est simple : chaque ligne, colonne et région ne doit contenir qu'une seule fois tous les chiffres de 1 à 9. Formulé autrement, chacun de ces ensembles doit contenir tous les chiffres de 1 à 9.

Une règle non écrite mais communément admise veut également qu'une bonne grille de Sudoku, une grille valide, ne doit présenter qu'une et une seule solution. Ce n'est néanmoins pas toujours le cas dans la presse.

Les chiffres ne sont utilisés que par convention, les relations arithmétiques entre eux ne servant pas. N'importe quel ensemble de signes distincts — lettres, formes, couleurs, symboles — peut être utilisé sans changer les règles du jeu.

L'intérêt du jeu réside dans la simplicité de ses règles, et dans la complexité de ses solutions. Les grilles publiées ont souvent un niveau de difficulté indicatif. L'éditeur peut aussi indiquer un temps de résolution probable. Quoiqu'en général les grilles contenant le plus de chiffres pré-remplis soient les plus simples, l'inverse n'est pas systématiquement vrai. Le véritable enjeu du Sudoku réside plutôt dans la difficulté de trouver la suite exacte des chiffres à ajouter.

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

Figure 1.1 – Exemple de grille de Sudoku

2 Résolution de grilles

a Notion de candidats

Les candidats sont les chiffres possibles dans une case donnée. Ils sont souvent marqués par les joueurs en petits, sur les cotés de la case. Quand il ne reste plus qu'un candidat pour une case, ce chiffre peut être attribué à cette case. On parle de paire quand une case contient deux candidats, de triplet pour trois et de quatrain pour quatre.

L'utilisation explicite des candidats est souvent requise pour résoudre une grille, à moins qu'elle soit extrêmement facile (ou que le joueur soit extrêmement doué !). Les méthodes présentées ci-dessous permettront ainsi soit d'éliminer des candidats dans une case, soit de choisir quel est le bon candidat dans la case.

b Singleton nu

On peut directement valider un chiffre dans une case quand il est la seule possibilité, compte tenu de la colonne, la ligne et du carré qui contiennent la case en question (figure 1.2).

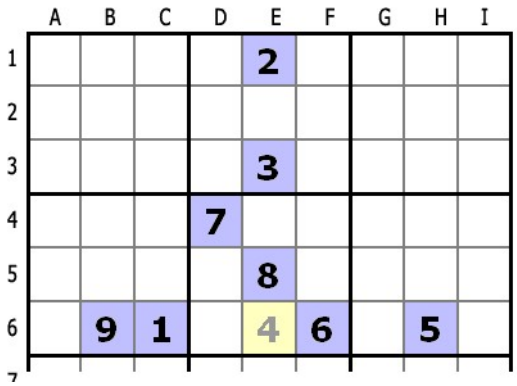

Figure 1.2 – En E6, seul le 4 est possible car les autres chiffres se trouvent déjà dans les groupes (ligne, colonne et carré) de cette case. 1, 5, 6, 9 peuvent être éliminés car ils sont déjà sur la ligne. 2, 3, 8 peuvent être éliminés car ils sont déjà sur la colonne. 7, 8, 6 peuvent être éliminés car ils sont déjà dans le carré. Il reste donc le 4 comme seul candidat.

c Singleton caché

Quand une case est la seule dans un groupe à pouvoir recevoir un chiffre, elle doit contenir ce chiffre, même si d'autres chiffres sont possibles par la méthode précédente, alors cette case contient ce chiffre (figure 1.3).

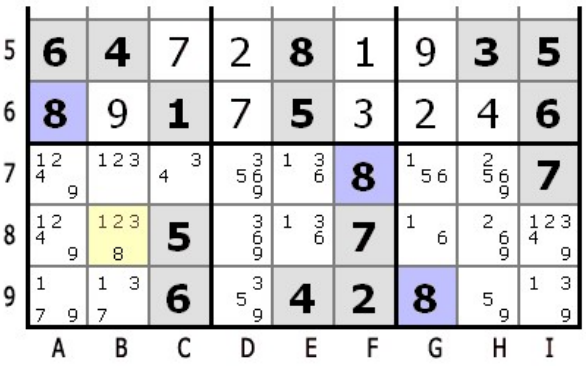

Figure 1.3 – Dans le 7ème carré (en bas à gauche), le 8 n'est possible que dans la case du centre, en B8. On peut donc éliminer les candidats 1, 2, 3 et placer 8 dans cette case.

d Candidat bloqué

Quand dans un carré, un chiffre n'est possible que sur un segment, alors le candidat peut être exclu de cette colonne/ligne dans les autres carrés (figure 1.4).

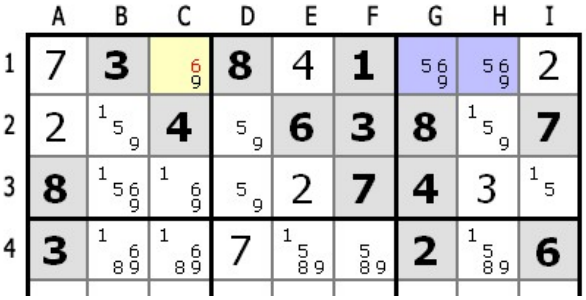

FIGURE 1.4 – Dans le 3ème carré (à droite), le 6 n'est possible que dans le segment du haut (GH-1). Il est donc forcément sur la ligne 1 dans ce carré. On peut le supprimer de C1.

Quand dans une ligne/colonne un seul carré peut contenir un chiffre, ce chiffre peut être exclu des autres cases de ce carré (figure 1.5).

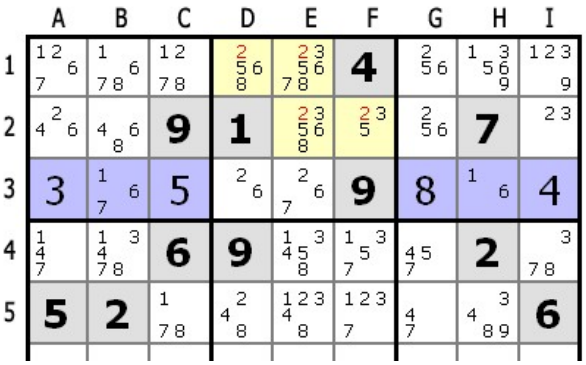

FIGURE 1.5 – Sur la 3ème ligne, seul le 2ème carré peut contenir un 2 (en D3 et E3). Le 2 ne peut donc pas se trouver dans ce carré sur les 1ère et 2ème lignes. On peut donc l'éliminer de D1, E1, E2 et F2.

e Ensembles

Paire nue : Quand un groupe contient deux cases avec une même paire de candidats (et eux seuls) alors ces candidats ne peuvent se trouver dans une autre case du groupe. Cela fonctionne pour une ligne, une colonne ou un carré (figure 1.6).

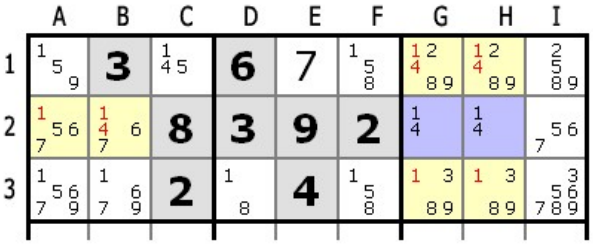

FIGURE 1.6 – Sur la 2ème ligne, on a la paire 1,4 en G2 et H2. On peut en déduire que si 1 est en G2, alors 4 est en H2 ou vice et versa. Mais ces deux cases contiendront forcément 1 et 4. On peut donc les supprimer du reste de la ligne. Comme ces paires appartiennent toutes les deux au 3ème carré à droite), alors on peut aussi éliminer 1 et 4 des autres cases de ce carré.

Triplet et quatrain nus : Quand trois cases d'un groupe ne contiennent pas d'autres chiffres que trois candidats, ces chiffres peuvent être exclus des autres cases du groupe. Attention ! Il n'y a pas besoin que ces trois cases contiennent tous les chiffres du triplet, il faut seulement que ces cases soient les seules à avoir les trois chiffres en commun. De même pour un quatrain, si quatre cases ne contiennent pas d'autres candidats qu'un quatrain déterminé, alors ces chiffres peuvent être exclus des autres cases du groupe.

Plus généralement, pour un nombre N de candidats d'un groupe, il faut trouver N cases qui contiennent ces candidats. Chaque case doit contenir un des candidat, donc pour trois cases contenant un triplet (même incomplet), les trois chiffres du triplet vont être répartis sur ces trois cases, et ne pourront donc pas être dans une autre case du groupe (figure 1.7).

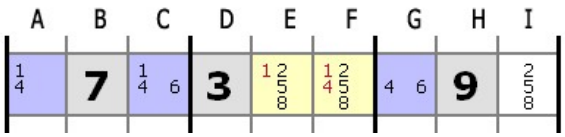

Figure 1.7 – Sur cette ligne, on a dans les cases A, C et G le triplet 1,4,6 ou deux candidats de ce triplet. Ces trois cases vont forcément contenir les trois chiffres de ce triplet, ils ne peuvent donc être ailleurs sur la ligne. On peut les supprimer des autres cases (ici E et F) avec certitude.

Ensemble caché : Pour les ensembles nus, les paires, triplets et quatrains permettent de supprimer des candidats dans le reste du groupe. Avec cette technique, les paires, triplets et quatrains permettent de supprimer les autres candidats des cases les contenant.

S'il y a N cases (2, 3 ou 4) contenant N chiffres en commun, alors tous les autres candidats de ces cases peuvent être exclus.

Comme pour les triplets et quatrains nus, les cases n'ont pas besoin de contenir tous les chiffres du triplet/quatrain (figure 1.8). Les triplets cachés sont très difficiles à repérer, ils sont heureusement rarement utiles pour résoudre un Sudoku. Les quatrains ne peuvent pratiquement pas être décelés !

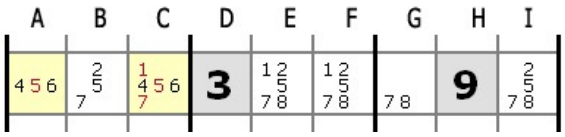

Figure 1.8 – Sur cette ligne, la paire 4,6 ne se trouve que dans les cases A et C. On peut donc éliminer les autres candidats pour ces deux cases car elles contiendront forcément soit 4, soit 6 et rien d'autre.

f Techniques complexes

Il existe de nombreuses autres méthodes de résolutions, plus complexes à mettre en œuvre car plus difficiles à repérer sans assistance numérique. Il s'agira typiquement d'associer des paires en différents endroits de la grille pour procéder à des éliminations, ou encore de réaliser des chainages logiques. Les détails de ces méthodes sont disponibles en ligne, par exemple sur le site dont sont extrait les exemples précédents : www.sudoku129.com/grilles/tips_intro.php .

g Résolution hasardeuse

Il peut arriver qu'on ne puisse pas avancer dans la résolution d'un Sudoku par une des méthodes logiques classiques. Il faut alors choisir arbitrairement un candidat d'une case et voir ce qui en résulte. Si on tombe sur une impasse ou une situation impossible, alors il faut remonter au moment où l'on a choisi arbitrairement et faire un autre choix.

Cette technique laisse une grande part au hasard et n'est normalement pas à utiliser pour résoudre un Sudoku à la main. Elle permet néanmoins de résoudre tout type de grille.

3 Création de grilles

a Nombre de grilles pleines possibles

En 2005, Bertram Felgenhauer et Frazer Jarvis ont prouvé que le nombre de grilles pleines possibles est de 6 670 903 752 021 072 936 960 $\approx 6,67 \times 10^{21}$

On peut néanmoins considérer que deux grilles complètes sont équivalentes si elles peuvent être transformées l'une en l'autre grâce à une combinaison quelconque des opérations suivantes : échange des lignes avec les colonnes (2 solutions) ; permutations des 9 nombres (9 ! solutions) ; permutation des trois lignes au sein d'un même bloc ($3!^3$ solutions) ou des trois colonnes ($3!^3$ solutions) ; permutation des trois blocs sur une ligne de blocs (3 ! solutions) ou sur une colonne de blocs (3 ! solutions).

Une grille complète permet ainsi de créer un total de $29!(3!)^8 = 1\,218\,998\,108\,160 \approx 1,22 \times 10^{12}$ grilles essentiellement équivalentes. Ainsi, il est donc possible de concevoir 5 472 730 538 $\approx 5,47 \times 10^9$ grilles pleines non-équivalentes.

b Grille minimale

A partir d'une grille pleine, il est possible de créer un grand nombre de grilles partiellement vides, mais aboutissant tout de même à un Sudoku ayant une solution unique. Parmi ces grilles, on s'intéresse le plus souvent à celles dites minimales.

Un Sudoku est dit minimal s'il a une solution unique et si, chaque fois qu'on essaie de lui ôter une case dévoilée, la grille obtenue (qui a toujours évidemment au moins une solution) se retrouve avoir plusieurs solutions.

La minimalité est une exigence annexe parfois attendue du créateur de Sudoku. Cependant elle peut être difficile à concilier avec d'autres exigences annexes, comme des exigences esthétiques, par exemple de symétrie, à savoir que les dévoilés se situent dans un ensemble de cellules présentant une certaine forme de symétrie.

Combien de cases initiales remplies sont nécessaires pour conduire à une solution unique ? C'est, à ce jour, une question ouverte. Le meilleur résultat, obtenu par une équipe de Japonais, est de 17 cases.

Remarque

Deux types d'approches sont à distinguer pour construire une grille de Sudoku :

- l'approche « bottom-up », qui part d'une grille vide et la remplit petit à petit ;
- l'approche « top-down », qui part d'une grille complète et élimine des indices un par un.

c Difficulté d'une grille

Plusieurs facteurs influent sur la difficulté des grilles : nombre de cellules à remplir, nombre d'étapes de résolution simples à effectuer, nombre d'hypothèses à faire en cas de blocage...

Cette question de la difficulté trouve souvent une réponse subjective, car elle est liée aux concepts et représentations visuelles que chacun est prêt à adopter. Mais elle peut être complètement élucidée si l'on hiérarchise, du simple au complexe, les techniques et procédés que l'on peut utiliser pour réussir une grille, et si l'on considère notre manière de jouer (observer certaines règles de handicap, comme la résolution intégrale par raisonnement mental uniquement, ou l'interdiction absolue de reproduire la grille-problème en plusieurs grilles en faisant des hypothèses, etc.).

On classe ainsi souvent les grilles en plusieurs types, suivant les techniques logiques devant être mises en oeuvre pour les résoudre.

III Contenu du projet

1 Fondamentaux

- Pouvoir créer une grille
  - → en rentrant les chiffres à la main dans certaines cases, en se basant sur une grille trouvée dans la presse par exemple.
  - → en chargeant la grille depuis un fichier texte.
  - → aléatoirement, avec néanmoins une solution unique et une base minimale.
- Pouvoir jouer une partie de Sudoku.
- Pouvoir sauvegarder une partie et pourvoir reprendre une partie interrompue.
- Pouvoir revenir en arrière sur ses coups joués.
- Pouvoir faire des hypothèses sur les candidats dans chaque case.
- Disposer d'une interface
  - → console, avec l'affichage de la grille et des interactions textuelles pour réaliser les actions.
  - → graphique, avec la possibilité de jouer directement dans les cases, et des boutons adaptés aux différentes options.
- Pouvoir résoudre automatiquement une grille existante
  - → avec des techniques logiques.
  - → avec une résolution hasardeuse, par récursivité via un arbre de résolution par exemple.
- Pouvoir vérifier si le chiffre affiché dans une case est le bon.
- Pouvoir demander de l'aide.
- Pouvoir accéder à un score (temps, nombre de coups joués, nombre d'aides obtenues, ou autre).

2 Aller plus loin

- Utiliser des techniques de résolution complexes.
- Proposer des outils de construction de grilles avec des particularités esthétiques, telle que la symétrie.
- Mettre en place une interface de création de grille (avec unicité, minimalité...) simplifiant le travail pour l'utilisateur, mais lui laissant la possibilité de faire des choix (méthode semi-automatisée).
- Mettre en place une méthode pour évaluer en profondeur la difficulté d'une grille (suivant des critères plus ou moins subjectifs, mais à définir).
- Adapter l'affichage, pour remplacer les chiffres par des lettres ou des dessins.
- Adapter le code pour des grilles de Sudoku 4x4, 16x16, 25x25 ou même de taille quelconque.
- Adapter le code pour des variantes du Sudoku : grilles 7x7 avec des hexaminos, grilles irrégulières, grilles se chevauchant...
- Mettre en place une façon de jouer en multijoueur (course contre la montre par exemple).