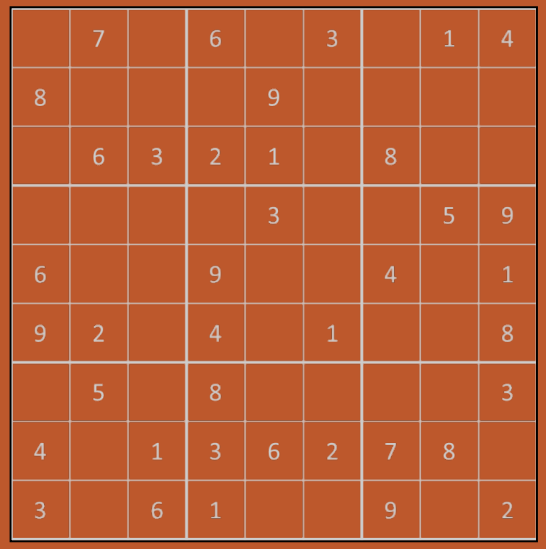
Résolution d’un Sudoku

Le jeu du Sodoku



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

But du jeu :

- remplir une **grille** de 9x9 **cases** avec une série de chiffres tous différents (les chiffres de 1 à 9 dans la version classique du jeu)
- qui ne se trouvent jamais plus d'une fois :
  - sur une même **ligne**
  - dans une même **colonne**
  - ou dans un même **carré 3x3**
- autrement appelés « **régions** »

Début de partie :

- certaines cases sont déjà connues
- ce qui autorise une résolution progressive du problème complet

Certaines règles simples peuvent permettre la résolution d'une grille de sudoku

Objectif du projet :

- implémenter un jeu de sudoku
- et sa **résolution** à l'aide de ces règles simples
- On pourra commencer par l'implémentation d'une seule règle.

Approche de modélisation du jeu du sudoku

Les Cases

Chaque case de la grille peut contenir les chiffres de 1 à 9.

A un instant *t* de la résolution, seules certaines valeurs sont envisageables pour chaque case

- Modélisation possible :
  - Associer à chaque case un domaine des possibles (liste de candidats) :

    ➔ ensemble des valeurs possibles pour cette case

Etat initial de la liste de candidats :

- Si la case est connue de valeur v

  ➔ la liste des candidats est réduite à la seule valeur v
- Si la case est non-connue (vide)

  ➔ la liste des candidats est composée des chiffres de 1 à 9

Résolution de la grille :

- Objectif : réduire le domaine des possibles à une seule valeur
- et donc rendre les cases vides connues

Règles de résolution

3 règles de filtrage « implémentables » identifiées

Première règle à programmer a minima

Chaque **règle de résolution :**

- est appliquée à une **région** de la grille de sudoku :
  ligne, colonne ou carré 3x3
- doit indiquer si l'application d'une règle a modifié la grille ou non

➔ Permet d'identifier si une grille est bloquée ou non

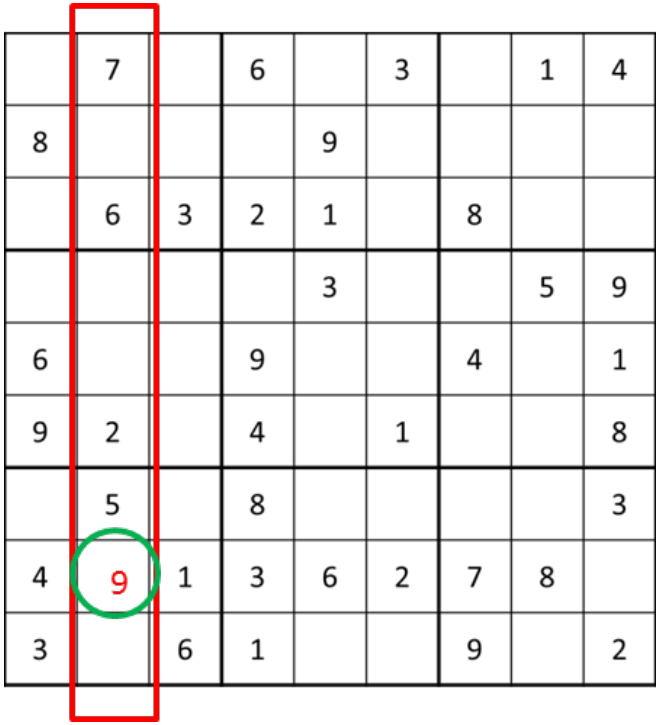
1ère règle de résolution

Chaque chiffre ne peut être attribué qu'une seule fois dans une région :

- ➔ si une valeur est déjà attribuée (case connues de la région)

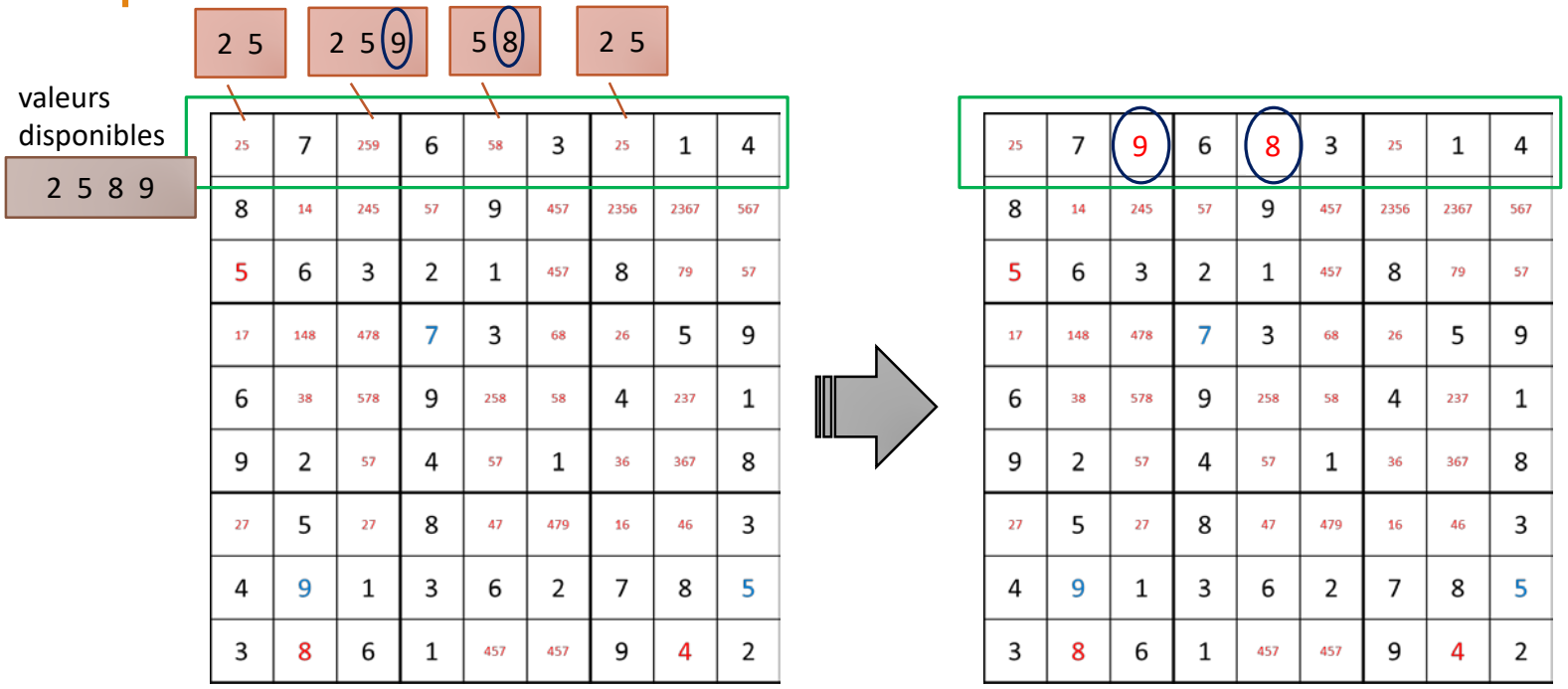
  elle ne peut donc être dans le domaine des possibles des autres cases
- ➔ Cette règle consiste donc :
  - à **exclure** du domaine des possibles d'une région donnée les valeurs déjà attribuées à une autre case dans cette région

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



2ème règle de résolution

Si un chiffre disponible n'est présent dans le domaine des possibles que d'une **unique** case de la région,
c'est qu'il **doit être attribué** à cette case.
Cette règle est parfois appelée règle du singleton caché.



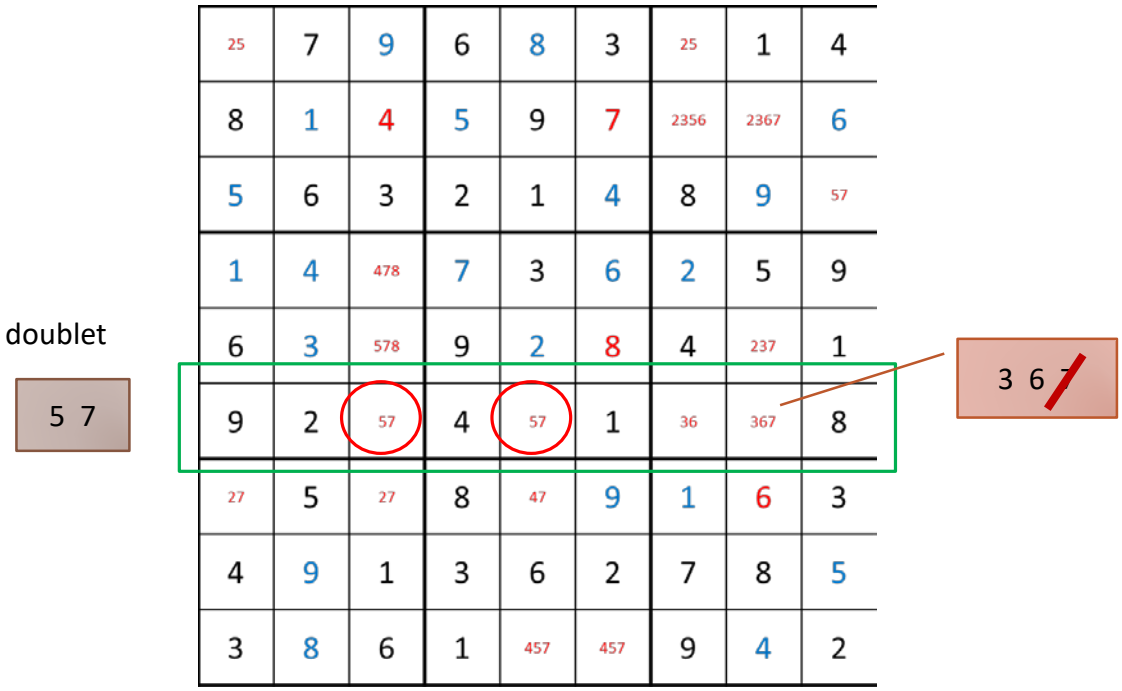
3ème règle de résolution

Si, sur une région, il existe **2 cases différentes** qui ont un **domaine des possibles identique** de **deux valeurs strictement**,

alors c'est deux valeurs sont nécessairement dans ces deux cases.

Elles doivent donc être retirées du domaines des possibles des autres cases.

Cette règle est parfois appelée **groupe nu** pour 2 cases.



25	7	9	6	8	3	25	1	4
8	1	4	5	9	7	2356	2367	6
5	6	3	2	1	4	8	9	57
1	4	478	7	3	6	2	5	9
6	3	578	9	2	8	4	237	1
9	2	57	4	57	1	36	367	8
27	5	27	8	47	9	1	6	3
4	9	1	3	6	2	7	8	5
3	8	6	1	457	457	9	4	2

Les Régions

Chaque règle s'applique de façon identique sur :

- les lignes
- les colonnes
- les carrés 3x3

➔ constituent des **Régions**

Chaque Région :

- est composée de 9 cases

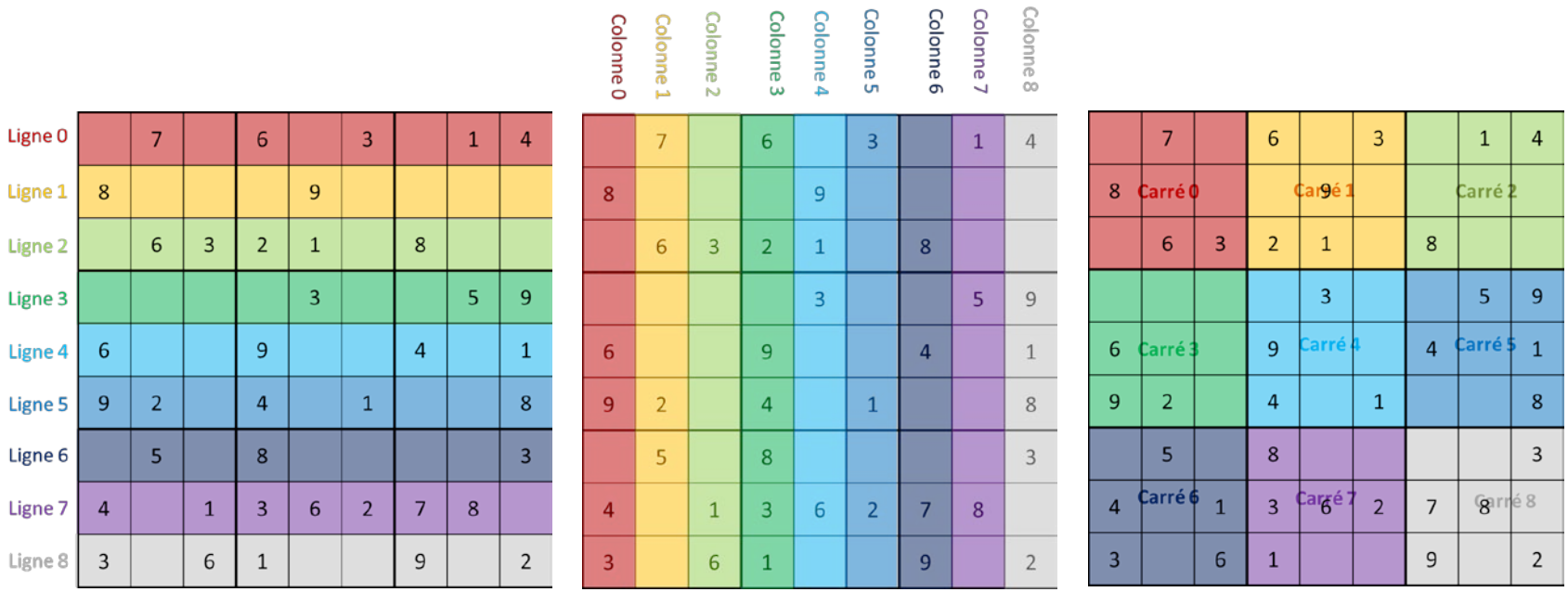
Les Régions

3 types de régions :

- les lignes
- les colonnes
- les carrés 3x3

Chaque Région :

- peut être créée à partir d'une « matrice » de cases
- Et d'une « position » dans la « matrice »



Implémentation de la modélisation choisie

La classe Case

Attributs

Le domaine des possibles
liste d'entiers de 1 à 9

Méthodes

Accesseurs pour chaque attribut

L'état de la case
booléen indiquant si la case est connue ou non

Affichage de la case

Retrait d'un ensemble de valeurs au domaine des possibles

Fixe la valeur d'une case

La classe Région

Attributs

Liste de 9 cases

Méthodes

Affichage de la région

Accesseurs de l'état de la région
(une région connue est une région dont toutes les cases sont connues)

Création de :

- la liste des **valeurs attribuées** (valeurs contenues dans les cases connues)
- la liste des **valeurs disponible**s (complémentaire de la précédente)
- et la liste des **cases vides**

Une **méthode pour chaque règle de résolution**

Pour chaque règle on retourne un marqueur qui précise si on a appliqué la règle

Les sous-classes de la classe Region

Sous-classes

- Ligne
- Colonne
- Carre

La classe Grille

Attributs

Nom de la grille

Liste des régions

crée à partir d'une matrice de cases obtenue par lecture du fichier

Une **case est partagée** par une ligne, une colonne et un carré

Une **case modifiée** par une des règles l'est dans **toutes les régions** qui la contiennent

Méthodes

Accesseurs de l'état de la grille

Resoudre : application des règles à l'ensemble des régions

- un filtrage par règle
- Indique si la grille est résolue, bloquée ou en cours

on applique les règles jusqu'à que la grille soit résolue ou bloquée (plus de modification possible par les règles)

Sauvegarde : Affichage du résultat dans un fichier de sortie

Activités de la séance

Ecrire les méthodes en rouge :

- De la classe *Grille*
  - lecture du fichier dans le constructeur de la classe
- De la classe *Case*
  - Aide : La méthode *remove* permet d'enlever un élément d'une liste
  - Les tester avec le main fourni
- De la classe *Region*
  - Dans l'ordre
  - Les tester au fur et à mesure avec le main fourni