TD1 – Sujet n°3

Sudoku

Dossier de conception et code

SAE 1.2 – Comparaison d'approches algorithmiques

Yannis DUVIGNAU – TD1 / TP1

Mattin GUIHENEUF – TD1 / TP2

Année universitaire 2022-2023

Table des matières

1. Principe du jeu

Le joueur doit compléter une grille (9 x 9) de valeurs, de sorte que :

- Chaque ligne de la grille,
- Chaque colonne de la grille,
- Et chaque zone (3 x 3) de la grille contienne les valeurs 1, 2, 3, 4, 5, 6, 7, 8 et 9.

Une grille incomplète est proposée au joueur en début de partie. A chaque tour le joueur propose une valeur à placer sur la grille.

Le joueur gagne lorsqu'il complète la grille correctement.

Le joueur perd lorsqu'il a consommé un nombre de droits à l'erreur prédéfini.

2. Spécifications du programme

2.1 Spécifications initiales

2.1.1 Spécifications du besoin

- Les règles du jeu sont toujours visibles
- Le nombre d'erreurs autorisées (> 3) est saisi par le joueur
- La grille de départ à compléter est fournie
- Les valeurs de la grille de départ sont d'une couleur différente des valeurs positionnées par l'utilisateur

A chaque tour :

- Le numéro de tour est affiché
- Le nombre d'erreurs réalisées et le nombre d'erreurs autorisés sont affichés
- Le joueur saisit les coordonnées et la valeur voulus : x, y, i ϵ { 1..9 } x { 1..9 } x { 1..9 }
- Lorsque la saisie est compatible avec la grille, le jeu le fait savoir et modifie la grille
- Lorsque la saisie est incompatible avec la grille, le jeu le fait savoir, compte une erreur de plus et liste les valeurs possibles au regard des valeurs présentes dans la zone (3 x 3), la ligne et la colonne.
- Lorsque la saisie modifie une ancienne valeur, le jeu le fait savoir et ne compte pas d'erreur
- Lorsque la saisie concerne une valeur de la grille de départ, le jeu le fait savoir et compte une erreur
- A tout instant, le joueur peur abandonner.

2.1.2 Situations à satisfaire

Situation 1 « Saisie compatible avec la grille »

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

Tour 1, Erreur 0/3, Abandon
Proposition (cf. x y i) ? 1 3 1
O U I !
Appuyez une touche pour continuer...
```

Situation 2 « Saisie incompatible avec la grille »

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

Tour 2, Erreur 0/3
Proposition (cf. x y i) ? 8 5 4
Erreur # valeur incompatible #
Valeurs possibles : 2, 5, 8
Appuyez une touche pour continuer...
```

Situation 3 « Saisie sur une valeur ancienne »

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

Tour 3, Erreur 1/3
Proposition (cf. x y i) ? 1 3 2
O U I !     valeur 1 modifiée en 2
Appuyez une touche pour continuer...
```

Situation 4 « Saisie sur grille de départ »

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

Tour 4, Erreur 1/3
Proposition (cf. x y i) ? 1 1 6
Erreur # saisie sur grille de départ #
Appuyez une touche pour continuer...
```

Situation 5 « Erreur de saisie »

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

Tour 5, Erreur 2/3
Proposition (cf. x y i) ? 1 e a
E R R E U R   D E   S A I S I E   ! ! !
Appuyez une touche pour continuer...
```

Situation 6 « Abandon »

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

Tour 5, Erreur 2/3
Proposition (cf. x y i) ? 0 0 0
A B A N D O N ! !
Appuyez une touche pour continuer...
```

Situation 7 « Dépasse le nombre d'erreurs »

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

Tour 8, Erreur 3/3
Proposition (cf. x y i) ? 8 2 7
Erreur # valeur incompatible #
Valeurs possibles : 2, 5, 8
P E R D U ! ! ! plus de 3 erreurs
Appuyez une touche pour continuer...
```

Situation 8 « Grille complétée »

```
    1 2 3   4 5 6   7 8 9
  -------------------------
1 | 5 3 4   6 7 8   9 1 2 | 1
2 | 6 7 2   1 9 5   3 4 8 | 2
3 | 1 9 8   3 4 2   5 6 7 | 3
  -------------------------
4 | 8 5 9   7 6 1   4 2 3 | 4
5 | 4 2 6   8 5 3   7 9 1 | 5
6 | 7 1 3   9 2 4   8 5 6 | 6
  -------------------------
7 | 9 6 1   5 3 7   2 8 4 | 7
8 | 2 8 7   4 1 9   6 3 5 | 8
9 | 3 4 5   2 8 6   1 7 9 | 9
  -------------------------
    1 2 3   4 5 6   7 8 9

Tour 61, Erreur 2/3
B R A V O ! ! ! !
Appuyez une touche pour continuer...
```

2.2 Spécifications complémentaires / Clarifications

Nous souhaitions clarifier la situation 5 “Erreur de Saisie”.

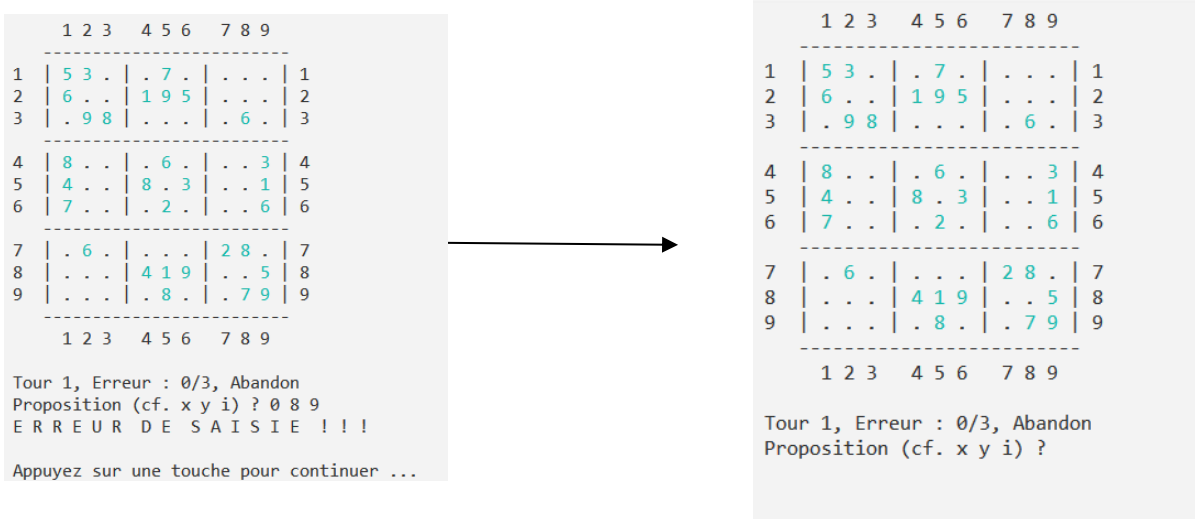
Le joueur saisit des valeurs numériques impossibles dans le contexte du Sudoku c’est-à-dire des valeurs numériques <= 0 ou > 9.

Comme une erreur de saisie dans la situation 5, elle ne sera pas comptabilisée comme une erreur dans le jeu et le compteur d’erreurs n’augmentera pas.

Par exemple :

- **Proposition (cf. x y i) ? 0 8 9**

  E R R E U R D E S A I S I E ! ! !



- **Proposition (cf. x y i) ? 10 10 3**

  E R R E U R D E S A I S I E ! ! !

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 10 10 3
E R R E U R  D E  S A I S I E  ! ! !

Appuyez sur une touche pour continuer ...
```



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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

Attention : Les 0 peuvent être acceptés dans la proposition que s'il y en a 3 :

Proposition (cf. x f i) ? 0 0 0 => Situation 6 "Abandon"

3. Jeux d'essais pour les situations initiales

3.1 Situation 1

Jeux d'essais	Données								
	propJoueur			charAbscisse	charOrdonnee	charElement	nbTour	grilleJeu [NB_CASES] [NB_CASES]	etatActuelProp
Situation 1 : Saisie compatible avec la grille	propJoueur.abscisse = 3	propJoueur.ordonnee = 1	propJoueur.element = 1	charAbscisse = '3'	charOrdonnee = '1'	charElement = '1'	nbTour = 1	{{'5', '3', '.', '.', '7', '.', '.', '.', '.'}, {'6', '.', '.', '1', '9', '5', '.', '.', '.'}, {'.', '9', '8', '.', '.', '.', '.', '6', '.'}, {'8', '.', '.', '.', '6', '.', '.', '.', '3'}, {'4', '.', '.', '8', '.', '3', '.', '.', '1'}, {'7', '.', '.', '.', '2', '.', '.', '.', '6'}, {'.', '6', '.', '.', '.', '.', '2', '8', '.'}, {'.', '.', '.', '4', '1', '9', '.', '.', '5'}, {'.', '.', '.', '.', '8', '.', '.', '7', '9'}}	etatActuelProp = enTrait
	propJoueur.abscisse = 5	propJoueur.ordonnee = 5	propJoueur.element = 5	charAbscisse = '5'	charOrdonnee = '5'	charElement = '5'	nbTour = 1	{{'5', '3', '.', '.', '7', '.', '.', '.', '.'}, {'6', '.', '.', '1', '9', '5', '.', '.', '.'}, {'.', '9', '8', '.', '.', '.', '.', '6', '.'}, {'8', '.', '.', '.', '6', '.', '.', '.', '3'}, {'4', '.', '.', '8', '.', '3', '.', '.', '1'}, {'7', '.', '.', '.', '2', '.', '.', '.', '6'}, {'.', '6', '.', '.', '.', '.', '2', '8', '.'}, {'.', '.', '.', '4', '1', '9', '.', '.', '5'}, {'.', '.', '.', '.', '8', '.', '.', '7', '9'}}	etatActuelProp = enTrait
	propJoueur.abscisse = 1	propJoueur.ordonnee = 6	propJoueur.element = 4	charAbscisse = '1'	charOrdonnee = '6'	charElement = '4'	nbTour = 1	{{'5', '3', '.', '.', '7', '.', '.', '.', '.'}, {'6', '.', '.', '1', '9', '5', '.', '.', '.'}, {'.', '9', '8', '.', '.', '.', '.', '6', '.'}, {'8', '.', '.', '.', '6', '.', '.', '.', '3'}, {'4', '.', '.', '8', '.', '3', '.', '.', '1'}, {'7', '.', '.', '.', '2', '.', '.', '.', '6'}, {'.', '6', '.', '.', '.', '.', '2', '8', '.'}, {'.', '.', '.', '4', '1', '9', '.', '.', '5'}, {'.', '.', '.', '.', '8', '.', '.', '7', '9'}}	etatActuelProp = enTrait

Résultats attendus								Afficher à l'écran
verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	verifErreurGrilleDep(grilleBase, propJoueur)	verifLaComp(grilleJeu, propJoueur)	nbTour	grilleJeu [NB_CASES] [NB_CASES]	etatActuelProp	grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1]	
FAUX	FAUX	FAUX	FAUX	nbTour = 2	{{'5', '3', ' ', ' ', ' ', '7', ' ', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', ' ', '1', '9', '5', ' ', ' ', ' ', ' '}, {'1', '9', '8', ' ', ' ', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', ' ', '6', ' ', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', ' ', '8', ' ', '3', ' ', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', ' ', '2', ' ', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', ' ', '2', '8', ' ', ' '}, {' ', ' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5', ' '}, {' ', ' ', ' ', ' ', ' ', '8', ' ', ' ', '7', '9', ' '}}	etatActuelProp = valComp	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] == ' '	cf Annexe_1, maquette_écran_1
FAUX	FAUX	FAUX	FAUX	nbTour = 5	{{'5', '3', ' ', ' ', ' ', '7', ' ', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', ' ', '1', '9', '5', ' ', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', ' ', '8', '5', '3', ' ', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', ' ', '2', ' ', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', ' ', '2', '8', ' ', ' '}, {' ', ' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5', ' '}, {' ', ' ', ' ', ' ', ' ', '8', ' ', ' ', '7', '9', ' '}}	etatActuelProp = valComp	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] == ' '	cf Annexe_1, maquette_écran_2
FAUX	FAUX	FAUX	FAUX	nbTour = 1	{{'5', '3', ' ', ' ', ' ', '7', '4', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', ' ', '1', '9', '5', ' ', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', ' ', '8', ' ', '3', ' ', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', ' ', '2', ' ', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', ' ', '2', '8', ' ', ' '}, {' ', ' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5', ' '}, {' ', ' ', ' ', ' ', ' ', '8', ' ', ' ', '7', '9', ' '}}	etatActuelProp = valComp	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] == ' '	cf Annexe_1, maquette_écran_3

3.2 Situation 2

Jeux d'essais	Données									
	propJoueur			charAbscisse	charOrdonnee	charElement	nbTour	grilleJeu [NB_CASES] [NB_CASES]	etatActuelProp	nbErreur
Situation 2 : Saisie incompatible avec la grille	propJoueur.abscisse = 5	propJoueur.ordonnee =	propJoueur.element = 4	charAbscisse = '5'	charOrdonnee = '8'	charElement = '4'	nbTour = 2	{{'5','3',' ',' ','7',' ',' ',' ',' '}, {'6',' ',' ','1','9','5',' ',' ',' '}, {'1','9','8',' ',' ',' ',' ','6',' '}, {'8',' ',' ',' ','6',' ',' ',' ','3'}, {'4',' ',' ','8',' ','3',' ',' ','1'}, {'7',' ',' ',' ','2',' ',' ',' ','6'}, {' ','6',' ',' ',' ',' ','2','8',' '}, {' ',' ',' ','4','1','9',' ',' ','5'}, {' ',' ',' ',' ','8',' ',' ','7','9'}}	etatActuelProp = enTrait	nbErreur = 0
	propJoueur.abscisse = 2	propJoueur.ordonnee =	propJoueur.element = 1	charAbscisse = '2'	charOrdonnee = '2'	charElement = '1'	nbTour = 2	{{'5','3',' ',' ','7',' ',' ',' ',' '}, {'6',' ',' ','1','9','5',' ',' ',' '}, {'1','9','8',' ',' ',' ',' ','6',' '}, {'8',' ',' ',' ','6',' ',' ',' ','3'}, {'4',' ',' ','8',' ','3',' ',' ','1'}, {'7',' ',' ',' ','2',' ',' ',' ','6'}, {' ','6',' ',' ',' ',' ','2','8',' '}, {' ',' ',' ','4','1','9',' ',' ','5'}, {' ',' ',' ',' ','8',' ',' ','7','9'}}	etatActuelProp = enTrait	nbErreur = 1
	propJoueur.abscisse = 9	propJoueur.ordonnee =	propJoueur.element = 6	charAbscisse = '5'	charOrdonnee = '3'	charElement = '7'	nbTour = 2	{{'5','3',' ',' ','7',' ',' ',' ',' '}, {'6',' ',' ','1','9','5',' ',' ',' '}, {'1','9','8',' ',' ',' ',' ','6',' '}, {'8',' ',' ',' ','6',' ',' ',' ','3'}, {'4',' ',' ','8',' ','3',' ',' ','1'}, {'7',' ',' ',' ','2',' ',' ',' ','6'}, {' ','6',' ',' ',' ',' ','2','8',' '}, {' ',' ',' ','4','1','9',' ',' ','5'}, {' ',' ',' ',' ','8',' ',' ','7','9'}}	etatActuelProp = enTrait	nbErreur = 0

Résultats attendus								Afficher à l'écran
nbErreur	etatActuelProp	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	nbTour	verifErreurGrilleDep(grilleBase, propJoueur)	verifLaComp(grilleJeu, propJoueur)	afficherValPossibles(grilleJeu, propJoueur, tabValIncomp, tabValComp, tabBoolVerif)	
nbErreur = 1	etatActuelProp = valIncomp	FAUX	FAUX	nbTour = 3	FAUX	VRAI	affichage de 2, 5 et 8	cf Annexe_1, maquette_écran_4
nbErreur = 0	etatActuelProp = valIncomp	FAUX	FAUX	nbTour = 3	FAUX	VRAI	affichage de 2, 4 et 7	cf Annexe_1, maquette_écran_5
nbErreur = 1	etatActuelProp = valIncomp	FAUX	FAUX	nbTour = 3	FAUX	VRAI	affichage de 1, 2, 3, 4 et 5	cf Annexe_1, maquette_écran_6

3.3 Situation 3

Jeux d'essais	Données								
	propJoueur			charAbscisse	charOrdonnee	charElement	nbTour	grilleJeu [NB_CASES] [NB_CASES]	etatActuelProp
Situation 3 : Saisie sur une valeur ancienne	propJoueur.abscisse = 3	propJoueur.ordonnee =	propJoueur.element = 2	charAbscisse = '3'	charOrdonnee = '1'	charElement = '2'	nbTour = 3	{{'5', '3', '1', ' ', ' ', '7', ' ', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', ' ', '1', '9', '5', ' ', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', ' ', '6', ' ', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', ' ', '8', ' ', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', ' ', '2', ' ', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', ' ', '4', '1', '9', ' ', ' ', ' ', '5'}, {' ', ' ', ' ', ' ', ' ', '8', ' ', ' ', ' ', '7', '9'}}	etatActuelProp = enTrait
	propJoueur.abscisse = 3	propJoueur.ordonnee =	propJoueur.abscisse = 4	charAbscisse = '3'	charOrdonnee = '9'	charElement = '4'	nbTour = 3	{{'5', '3', ' ', ' ', '7', ' ', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', ' ', '6', '2'}, {'8', ' ', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', ' ', '5'}, {' ', ' ', ' ', ' ', '8', ' ', ' ', ' ', '7', '9'}}	etatActuelProp = enTrait
	propJoueur.abscisse = 1	propJoueur.ordonnee =	propJoueur.element = 2	charAbscisse = '1'	charOrdonnee = '6'	charElement = '2'	nbTour = 3	{{'5', '3', ' ', ' ', '7', '4', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', ' ', '5'}, {' ', ' ', ' ', ' ', '8', ' ', ' ', ' ', '7', '9'}}	etatActuelProp = enTrait

Résultats attendus								Afficher à l'écran
etatActuelProp	nbTour	grilleJeu [NB_CASES] [NB_CASES]	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	verifErreurGrilleDep(grilleBase, propJoueur)	verifLaComp(grilleJeu, propJoueur)	grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1]	
etatActuelProp = valComp	nbTour = 2	{{'5', '3', ' ', ' ', '7', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' '}, {'2', '9', '8', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5'}, {' ', ' ', ' ', ' ', '8', ' ', ' ', '7', '9'}}	FAUX	FAUX	FAUX	FAUX	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] != ' '	cf Annexe_1, maquette_écran_7
etatActuelProp = valComp	nbTour = 2	{{'5', '3', ' ', ' ', '7', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', '6', '4'}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5'}, {' ', ' ', ' ', ' ', '8', ' ', ' ', '7', '9'}}	FAUX	FAUX	FAUX	FAUX	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] != ' '	cf Annexe_1, maquette_écran_8
etatActuelProp = valComp	nbTour = 2	{{'5', '3', ' ', ' ', '7', '2', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5'}, {' ', ' ', ' ', ' ', '8', ' ', ' ', '7', '9'}}	FAUX	FAUX	FAUX	FAUX	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] != ' '	cf Annexe_1, maquette_écran_9

3.4 Situation 4

Jeux d'essais	Données									
	propJoueur			charAbscisse	charOrdonnee	charElement	nbTour	grilleJeu [NB_CASES] [NB_CASES]	nbErreur	etatActuelProp
Situation 4 : Saisie sur grille de départ	propJoueur.abscisse = 1	propJoueur.ordonnee =	propJoueur.element = 6	charAbscisse = '1'	charOrdonnee = '1'	charElement = '6'	nbTour = 4	{{'5','3',' ',' ',' ','7',' ',' ',' ',' '}, {'6',' ',' ',' ','1','9','5',' ',' ',' '}, {' ','9','8',' ',' ',' ',' ',' ','6',' '}, {'8',' ',' ',' ',' ','6',' ',' ',' ','3'}, {'4',' ',' ','8',' ','3',' ',' ','1'}, {'7',' ',' ',' ',' ','2',' ',' ',' ','6'}, {' ','6',' ',' ',' ',' ',' ','2','8',' '}, {' ',' ',' ','4','1','9',' ',' ','5'}, {' ',' ',' ',' ',' ','8',' ',' ','7','9'}}	nbErreur = 1	etatActuelProp = enTrait
	propJoueur.abscisse = 3	propJoueur.ordonnee =	propJoueur.element = 1	charAbscisse = '3'	charOrdonnee = '3'	charElement = '1'	nbTour = 4	{{'5','3',' ',' ',' ','7',' ',' ',' ',' '}, {'6',' ',' ',' ','1','9','5',' ',' ',' '}, {' ','9','8',' ',' ',' ',' ',' ','6','4'}, {'8',' ',' ',' ',' ','6',' ',' ',' ','3'}, {'4',' ',' ','8',' ','3',' ',' ','1'}, {'7',' ',' ',' ',' ','2',' ',' ',' ','6'}, {' ','6',' ',' ',' ',' ',' ','2','8',' '}, {' ',' ',' ','4','1','9',' ',' ','5'}, {' ',' ',' ',' ',' ','8',' ',' ','7','9'}}	nbErreur = 0	etatActuelProp = enTrait
	propJoueur.abscisse = 9	propJoueur.ordonnee =	propJoueur.element = 7	charAbscisse = '9'	charOrdonnee = '5'	charElement = '7'	nbTour = 4	{{'5','3',' ',' ',' ','7','2',' ',' ',' '}, {'6',' ',' ',' ','1','9','5',' ',' ',' '}, {' ','9','8',' ',' ',' ',' ',' ','6',' '}, {'8',' ',' ',' ',' ','6',' ',' ',' ','3'}, {'4',' ',' ','8',' ','3',' ',' ','1'}, {'7',' ',' ',' ',' ','2',' ',' ',' ','6'}, {' ','6',' ',' ',' ',' ',' ','2','8',' '}, {' ',' ',' ','4','1','9',' ',' ','5'}, {' ',' ',' ',' ',' ','8',' ',' ','7','9'}}	nbErreur = 2	etatActuelProp = enTrait

Résultats attendus						Afficher à l'écran
etatActuelProp	nbTour	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	verifErreurGrilleDep(grilleBase, propJoueur)	nbErreur	
etatActuelProp = erreurGrilleDep	nbTour = 5	FAUX	FAUX	VRAI	nbErreur = 2	cf Annexe_1, maquette_écran_10
etatActuelProp = erreurGrilleDep	nbTour = 5	FAUX	FAUX	VRAI	nbErreur = 1	cf Annexe_1, maquette_écran_11
etatActuelProp = erreurGrilleDep	nbTour = 5	FAUX	FAUX	VRAI	nbErreur = 3	cf Annexe_1, maquette_écran_12

3.5 Situation 5

Jeux d'essais	Données					
	charAbscisse	charOrdonnee	charElement	nbTour	etatActuelProp	nbErreur
Situation 5 : Erreur de saisie	charAbscisse = 'l'	charOrdonnee = 'e'	charElement = 'a'	nbTour = 5	etatActuelProp = enTrait	nbErreur = 2
	charAbscisse = 'a'	charOrdonnee = '2'	charElement = '3'	nbTour = 5	etatActuelProp = enTrait	nbErreur = 0
	charAbscisse = 'l'	charOrdonnee = '5'	charElement = 'c'	nbTour = 5	etatActuelProp = enTrait	nbErreur = 1

Résultats attendus					Afficher à l'écran
etatActuelProp	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	nbTour	nbErreur	
etatActuelProp = erreurSaisie	FAUX	VRAI	nbTour = 6	nbErreur = 3	cf Annexe_1, maquette_écran_13
etatActuelProp = erreurSaisie	FAUX	VRAI	nbTour = 6	nbErreur = 1	cf Annexe_1, maquette_écran_14
etatActuelProp = erreurSaisie	FAUX	VRAI	nbTour = 6	nbErreur = 2	cf Annexe_1, maquette_écran_15

3.6 Situation 6

Jeux d'essais	Données									
	propJoueur			charAbscisse	charOrdonnee	charElement	nbTour	nbErreur	stadeActuelPartie	etatActuelProp
Situation 6 : Abandon	propJoueur.abscisse = 0	propJoueur.ordonnee = 0	propJoueur.element = 0	charAbscisse = '0'	charOrdonnee = '0'	charElement = '0'	nbTour = 5	nbErreur = 2	stadeActuelPartie = enCours	etatActuelProp = enTrait
	propJoueur.abscisse = 0	propJoueur.ordonnee = 0	propJoueur.element = 0	charAbscisse = '0'	charOrdonnee = '0'	charElement = '0'	nbTour = 4	nbErreur = 0	stadeActuelPartie = enCours	etatActuelProp = enTrait

Résultats attendus					Afficher à l'écran
etatActuelProp	stadeActuelPartie	nbTour	verifAbandon(propJoueur)	nbErreur	
etatActuelProp = abandon	stadeActuelPartie = choixAbandon	nbTour = 6	VRAI	nbErreur = 3	cf Annexe_1, maquette_écran_16
etatActuelProp = abandon	stadeActuelPartie = choixAbandon	nbTour = 5	VRAI	nbErreur = 1	cf Annexe_1, maquette_écran_17

3.7 Situation 7

Jeux d'essais	Données											
	propJoueur			charAbscisse	charOrdonnee	charElement	nbTour	grilleJeu [NB_CASES] [NB_CASES]	nbErreurMax	nbErreur	etatActuelProp	stadeActuelPartie
Situation 7 : Dépasse le nombre d'erreurs	propJoueur.abscisse = 2	propJoueur.ordonnee = 8	propJoueur.element = 7	charAbscisse = '2'	charOrdonnee = '8'	charElement = '7'	nbTour = 8	{{'5', '3', ' ', ' ', '7', ' ', ' ', '9', '1', '2'}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', '8'}, {'2', '9', '8', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5'}, {' ', ' ', ' ', '8', ' ', ' ', '7', '9'}}	nbErreurMax = 3	nbErreur = 2	etatActuelProp = enTrait	stadeActuelPartie = enCours
	Pas de donnée			charAbscisse = 'l'	charOrdonnee = 'e'	charElement = 'a'	nbTour = 4	{{'5', '3', ' ', ' ', '7', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5'}, {' ', ' ', ' ', '8', ' ', ' ', '7', '9'}}	nbErreurMax = 4	nbErreur = 3	etatActuelProp = enTrait	stadeActuelPartie = enCours
	propJoueur.abscisse = 1	propJoueur.ordonnee = 1	propJoueur.element = 6	charAbscisse = '1'	charOrdonnee = '1'	charElement = '6'	nbTour = 4	{{'5', '3', ' ', ' ', '7', ' ', ' ', ' ', ' '}, {'6', ' ', ' ', '1', '9', '5', ' ', ' ', ' '}, {' ', '9', '8', ' ', ' ', ' ', ' ', '6', ' '}, {'8', ' ', ' ', ' ', '6', ' ', ' ', ' ', '3'}, {'4', ' ', ' ', '8', ' ', '3', ' ', ' ', '1'}, {'7', ' ', ' ', ' ', '2', ' ', ' ', ' ', '6'}, {' ', '6', ' ', ' ', ' ', ' ', '2', '8', ' '}, {' ', ' ', ' ', '4', '1', '9', ' ', ' ', '5'}, {' ', ' ', ' ', '8', ' ', ' ', '7', '9'}}	nbErreurMax = 18	nbErreur = 17	etatActuelProp = enTrait	stadeActuelPartie = enCours

Résultats attendus									Afficher à l'écran
stadeActuelPartie	etatActuelProp	nbTour	nbErreur	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	verifErreurGrilleDep(grilleBase, propJoueur)	verifLaComp(grilleJeu, propJoueur)	afficherValPossibles(grilleJeu, propJoueur, tabValIncomp, tabValComp, tabBoolVerif)	
stadeActuelPartie = erreurMax	etatActuelProp = valIncomp	nbTour = 9	nbErreur = 3	FAUX	FAUX	FAUX	VRAI	affichage de 2, 5 et 8	cf Annexe_1, maquette_écran_18
stadeActuelPartie = erreurMax	etatActuelProp = erreurSaisie	nbTour = 5	nbErreur = 4	FAUX	VRAI	Aucun résultat			cf Annexe_1, maquette_écran_19
stadeActuelPartie = erreurMax	etatActuelProp = erreurGrilleDep	nbTour = 5	nbErreur = 18	FAUX	FAUX	VRAI	Aucun résultat		cf Annexe_1, maquette_écran_20

3.8 Situation 8

Jeux d'essais	Données								
	propJoueur			charAbscisse	charOrdonnee	charElement	grilleJeu [NB_CASES] [NB_CASES]	nbTour	stadeActuelPartie
Situation 8 : Grille complétée	propJoueur.abscisse = 9	propJoueur.ordonnee = 1	propJoueur.element = 3	charAbscisse = '9'	charOrdonnee = '1'	charElement = '3'	{{'5', '3', '4', '6', '7', '8', '9', '1', '2'}, {'6', '7', '2', '1', '9', '5', '3', '4', '8'}, {'1', '9', '8', '3', '4', '2', '5', '6', '7'}, {'8', '5', '9', '7', '6', '1', '4', '2', '3'}, {'4', '2', '6', '8', '5', '3', '7', '9', '1'}, {'7', '1', '3', '9', '2', '4', '8', '5', '6'}, {'9', '6', '1', '5', '3', '7', '2', '8', '4'}, {'2', '8', '7', '4', '1', '9', '6', '3', '5'}, {'.', '4', '5', '2', '8', '6', '1', '7', '9'}}	nbTour = 61	stadeActuelPartie = enCours
	propJoueur.abscisse = 9	propJoueur.ordonnee = 1	propJoueur.element = 3	charAbscisse = '9'	charOrdonnee = '1'	charElement = '3'	{{'5', '3', '4', '6', '7', '8', '9', '1', '2'}, {'6', '7', '2', '1', '9', '5', '3', '4', '8'}, {'1', '9', '8', '3', '4', '2', '5', '6', '7'}, {'8', '5', '9', '7', '6', '1', '4', '2', '3'}, {'4', '2', '6', '8', '5', '3', '7', '9', '1'}, {'7', '1', '3', '9', '2', '4', '8', '5', '6'}, {'9', '6', '1', '5', '3', '7', '2', '8', '4'}, {'2', '8', '7', '4', '1', '9', '6', '3', '5'}, {'.', '4', '5', '2', '8', '6', '1', '7', '9'}}	nbTour = 90	stadeActuelPartie = enCours

Résultats attendus								Afficher à l'écran
stadeActuelPartie	recherchePointGrille(grilleJeu)	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	verifErreurGrilleDep(grilleBase, propJoueur)	verifLaComp(grilleJeu, propJoueur)	grilleJeu [NB_CASES] [NB_CASES]	grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1]	
stadeActuelPartie = victoire	VRAI	FAUX	FAUX	FAUX	FAUX	{{'5', '3', '4', '6', '7', '8', '9', '1', '2'}, {'6', '7', '2', '1', '9', '5', '3', '4', '8'}, {'1', '9', '8', '3', '4', '2', '5', '6', '7'}, {'8', '5', '9', '7', '6', '1', '4', '2', '3'}, {'4', '2', '6', '8', '5', '3', '7', '9', '1'}, {'7', '1', '3', '9', '2', '4', '8', '5', '6'}, {'9', '6', '1', '5', '3', '7', '2', '8', '4'}, {'2', '8', '7', '4', '1', '9', '6', '3', '5'}, {'3', '4', '5', '2', '8', '6', '1', '7', '9'}}	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] == '.'	cf Annexe_1, maquette_écran_21
stadeActuelPartie = victoire	VRAI	FAUX	FAUX	FAUX	FAUX	{{'5', '3', '4', '6', '7', '8', '9', '1', '2'}, {'6', '7', '2', '1', '9', '5', '3', '4', '8'}, {'1', '9', '8', '3', '4', '2', '5', '6', '7'}, {'8', '5', '9', '7', '6', '1', '4', '2', '3'}, {'4', '2', '6', '8', '5', '3', '7', '9', '1'}, {'7', '1', '3', '9', '2', '4', '8', '5', '6'}, {'9', '6', '1', '5', '3', '7', '2', '8', '4'}, {'2', '8', '7', '4', '1', '9', '6', '3', '5'}, {'3', '4', '5', '2', '8', '6', '1', '7', '9'}}	grilleJeu[propJoueur.abscisse - 1] [propJoueur.ordonnee - 1] == '.'	cf Annexe_1, maquette_écran_22

4 Jeux d'essais pour les situations complémentaires

4.1 Situation 1

Jeux d'essais	Données					
	charAbscisse	charOrdonnee	charElement	nbTour	etatActuelProp	nbErreur
Situation 1 : Erreur de saisie (contenant un ou deux zéro(s) dans la proposition)	charAbscisse = '0'	charOrdonnee = '0'	charElement = '1'	nbTour = 5	etatActuelProp = enTrait	nbErreur = 2
	charAbscisse = '0'	charOrdonnee = '1'	charElement = '1'	nbTour = 5	etatActuelProp = enTrait	nbErreur = 0

Résultats attendus					Afficher à l'écran
etatActuelProp	verifAbandon(propJoueur)	verifErreurSaisie(propJoueur)	nbTour	nbErreur	
etatActuelProp = erreurSaisie	FAUX	VRAI	nbTour = 6	nbErreur = 3	cf Annexe_1, maquette_écran_23
etatActuelProp = erreurSaisie	FAUX	VRAI	nbTour = 6	nbErreur = 1	cf Annexe_1, maquette_écran_24

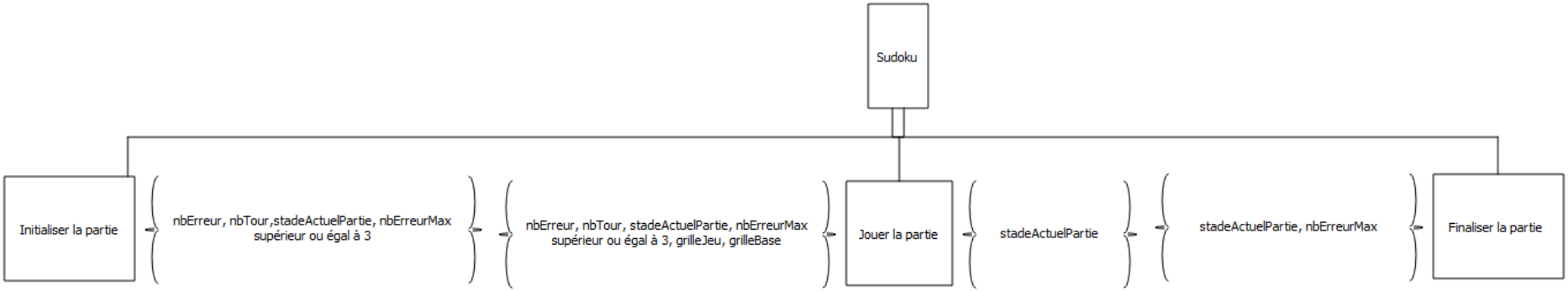
4.2Situation 2

Jeux d'essais	Données	Résultats attendus	Afficher à l'écran
	stringNbErreurMax	nbErreurMax	
Situation 2 : Erreur de saisie (sur le nombre d'erreur maximum)	stringNbErreurMax = "54"	nbErreurMax = 54	cf Annexe_1, maquette_écran_25
	stringNbErreurMax = "-8"	nbErreurMax = 3	cf Annexe_1, maquette_écran_26
	stringNbErreurMax = "14,5"	nbErreurMax = 3	cf Annexe_1, maquette_écran_27
	stringNbErreurMax = "14.5"	nbErreurMax = 3	cf Annexe_1, maquette_écran_28
	stringNbErreurMax = "a"	nbErreurMax = 3	cf Annexe_1, maquette_écran_29
	stringNbErreurMax = "abc"	nbErreurMax = 3	cf Annexe_1, maquette_écran_30
	stringNbErreurMax = "£_8)"	nbErreurMax = 3	cf Annexe_1, maquette_écran_31

Algorithme principal (programme principal)

Sudoku

Algorithme



Initialiser la partie : Dans cette étape, il s'agit d'initialiser les éléments permettant de commencer la partie. Dans notre cas, il s'agit d'initialiser les paramètres permettant d'entamer une partie.

Jouer la partie : Dans cette étape, il s'agit de compléter le sudoku proposé. L'utilisateur rentre des points de coordonnées (x et y) ainsi qu'une valeur à ajouter (i). Le joueur peut faire des erreurs jusqu'à que le nombres d'erreur dépasse le nombre d'erreur maximum (saisie dans la partie « initialiser la partie »).

Finaliser la partie : Dans cette dernière étape, il s'agit de conclure la partie en présentant au joueur la raison pour laquelle la partie a pris fin (abandon, victoire, nombre d'erreur dépassés)

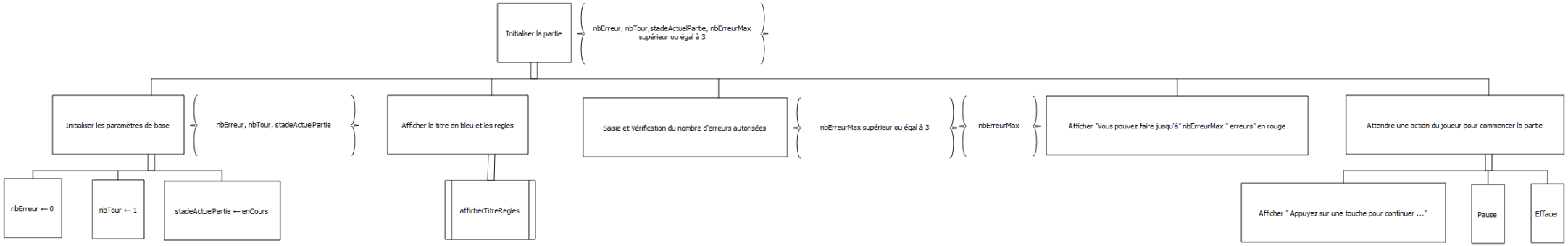
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.
NB_TOUR_MIN_FIN	Constant entier court non signé	Le nombre de tours minimum pour pouvoir compléter la grille. Il a été calculer au préalable pour optimiser le jeu et éviter de faire des vérifications inutiles : Il y a 81 cases et 30 chiffres pré-rentrées donc le joueur peut finir le jeu en 51 coups minimum soit 51 tours.
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.

Initialiser la partie

Cette étape consiste à mettre en place les paramètres nécessaires pour pouvoir commencer la partie.

Algorithme



Description des sous-problèmes

L'initialisation de la partie consiste se décompose en cinq étapes :

- L'affichage du titre et des règles du jeu du sudoku
- L'initialiser des paramètres de base du jeu. Il s'agit d'initialiser le nombre de tour pour et le nombre d'erreur et le stade actuel de la partie, utilisé pour la condition de sortie de jouer la partie.
- La saisie et la vérification du nombre d'erreur (maximum) autorisées. Ce sous-problème consiste à saisir le nombre d'erreur maximum puis vérifier que celui-ci ne soit pas inférieur à 3.
- L'affichage du nombre d'erreur maximum que l'utilisateur peut faire.
- L'attente d'une action du joueur pour commencer la partie. Pour continuer l'utilisateur doit appuyer sur une touche (quelconque).

- Effacer le terminal précédent.

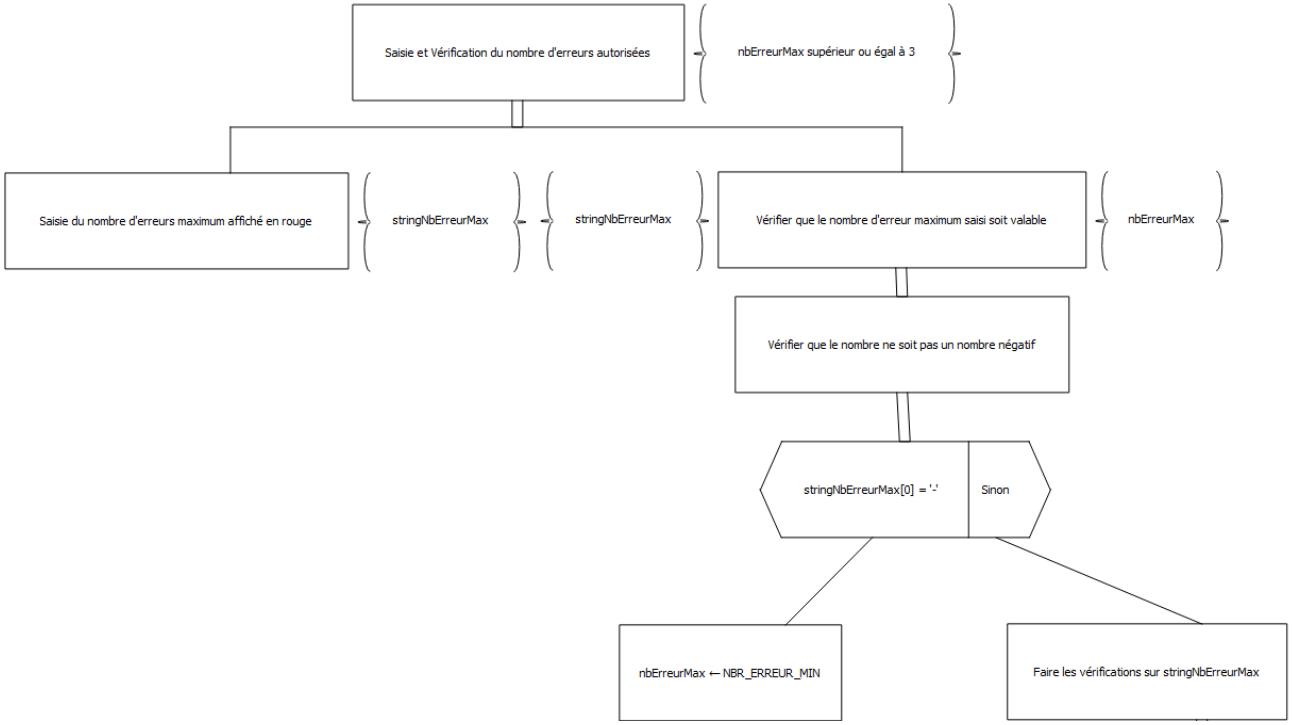
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.

Saisie-Verif du nombre d’erreurs autorisées

Le sous-problème Saisie-Verif du nombre d’erreurs autorisées gère la saisie et la vérification de la saisie (conforme aux contraintes des scénarios et des règles du jeu).

Algorithme



Description des sous-problèmes

Le sous-problème saisie-verif du nombre d'erreurs autorisées consiste à répéter les 2 étapes :

- **Saisie du nombre maximum d'erreur** : Ce sous-problème consiste demander à l'utilisateur de saisir un nombre d'erreur qu'il ne pourra pas dépasser durant toute la partie de sudoku.

- **Vérifier que le nombre d'erreur maximum autorisé**: L'objectif de cette partie est de vérifier que le nombre d'erreur saisie précédemment ne soit ni une lettre ni un chiffre inférieur à 3. Ce traitement vérifie d'abord que la saisie ne soit pas une lettre, si c'est le cas la demande de saisie est répétée jusqu'à que l'utilisateur saisisse un chiffre. Si ce chiffre saisit est inférieur à 3, alors le nombre d'erreur maximum est (par défaut) égal à 3. Dans le cas contraire, le nombre d'erreur maximum est retenu.

Le résultat de la saisie est attribué à une variable stringNbErreurMax de type chaine de caractère et non pas entier car elle sera utilisée par la suite pour vérifier que la saisie ne soit pas une lettre (boucle infinie si affectation d'un caractère à un entier)
Une conversion est faite pour une raison de cohérence (un chiffre est un entier et non un caractère) et est nécessaire pour passer la saisie de caractère à entier.

Le résultat de ce sous-problème est donc une valeur du nombre d'erreur maximum correcte et supérieur ou égale à 3.

Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
stringNbErreurMax	Caractère	Correspond à la saisie son forme d'un chaine de caractère qui servira par la suite à faire les vérification de validation du nombre d'erreurs maximum.
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.
NBR_ERREUR_MIN	Constant entier court non signé	Le nombre d'erreurs minimum autorisé que le joueur peut saisir.

Jouer la partie

Cette étape consiste à jouer la partie de sudoku.

Algorithme

Description des sous-problèmes

Cette étape consiste à répéter 4 étapes marquant le déroulement d'une partie.
Ces 4 étapes sont les suivantes :

- Initialiser le paramètre de gestion d'un tour. L'état actuel de la proposition est initialisé pour pouvoir suivre le dérouler d'un tour.
- Jouer un tour consiste à rentrer une proposition et faire la vérification de celle-ci.
- Effacer le terminal pour un côté esthétique et éviter de surcharger le terminal de commande lors d'une partie
- Condition d'arrêt d'une partie. L'algorithme vérifie l'utilisateur ne sois pas en situation d'abandon ou du nombre d'erreur maximum dépassé

Stratégie de l'algorithme mise en œuvre et justification

Dictionnaire des variables

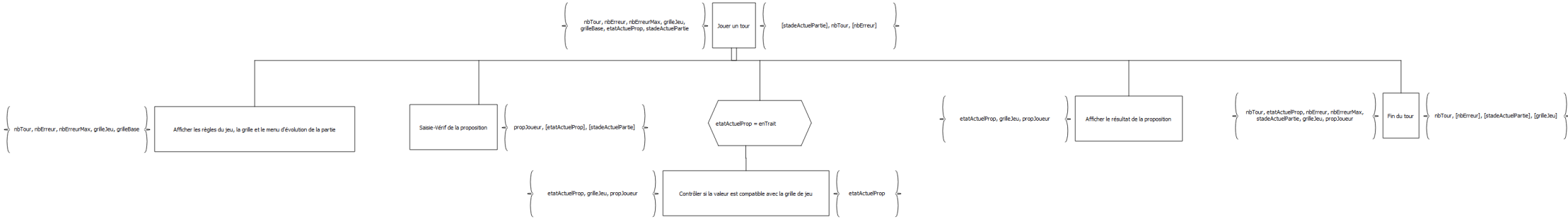
NOM	TYPE	SIGNIFICATION
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.

grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.
NB_TOUR_MIN_FIN	Constant entier court non signé	Le nombre de tours minimum pour pouvoir compléter la grille. Il a été calculer au préalable pour optimiser le jeu et éviter de faire des vérifications inutiles : Il y a 81 cases et 30 chiffres pré-rentrées donc le joueur peut finir le jeu en 51 coups minimum soit 51 tours.
tabValIncomp[NB_CASES]	Tableau de caractères	Le tableau dans lequel on va ajouter (sans doublon) toutes les valeurs trouvées sur ligne, colonne et carré pour des coordonnées [x, y] rentrées. Ce tableau est utilisé pour afficher les valeurs possibles lorsque l'utilisateur se trompe (scénario 2)
tabValComp[NB_CASES]	Tableau de caractères	Tableau de 1 à 9 utilisé pour comparer les valeurs lors de l'affichage des valeurs possibles (scénario 2)
tabBoolVerif[NB_CASES]	Tableau de booléen	Le tableau dans lequel les valeurs incompatibles présentent dans le tableau tabValIncomp seront égale à faux, vrai sinon. Utiliser pour trouver les valeurs possible (scénario 2)
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.

Jouer un tour

Cette étape consiste à jouer un tour.

Algorithme



Description des sous-problèmes

Le sous-problème « Afficher les règles du jeu, la grille et l'évolution de la partie » se découpe en 3 étapes :

- L'affichage des règles du jeu.
- L'affichage de la grille (procédure).
- L'affichage du menu d'évolution de la partie (affichage du nombre de tour réaliser par le joueur, affichage du nombre d'erreur et du nombre d'erreur maximum et l'affichage de la possibilité d'abandon).

Dictionnaire des variables

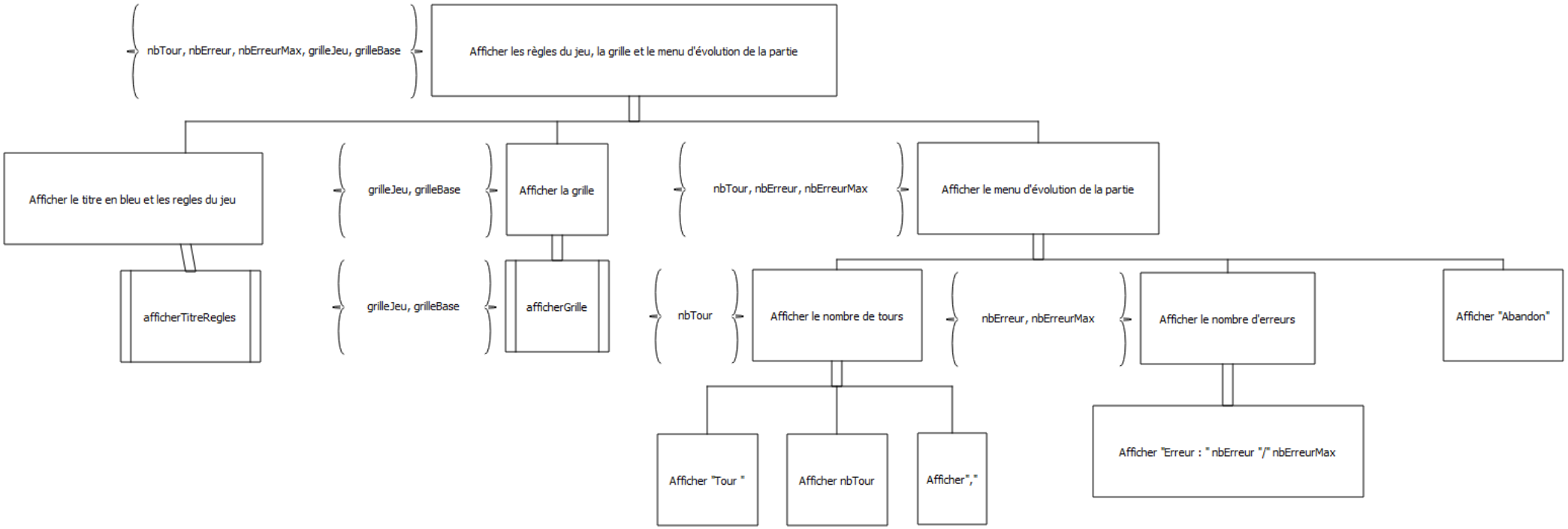
NOM	TYPE	SIGNIFICATION
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.
NB_TOUR_MIN_FIN	Constant entier court non signé	Le nombre de tours minimum pour pouvoir compléter la grille. Il a été calculer au préalable pour optimiser le jeu et éviter de faire des vérifications inutiles : Il y a 81 cases et 30 chiffres pré-rentrées donc le joueur peut finir le jeu en 51 coups minimum soit 51 tours.
charAbscisse	Caractère	L'abscisse de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charOrdonnee	Caractère	L'ordonnée de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charElement	Caractère	L'élément de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
tabValIncomp[NB_CASES]	Tableau de caractères	Le tableau dans lequel on va ajouter (sans doublon) toutes les valeurs trouvées sur ligne, colonne et carré pour des coordonnées [x, y] rentrées. Ce tableau est utilisé pour afficher les valeurs possibles lorsque l'utilisateur se trompe (scénario 2)
tabValComp[NB_CASES]	Tableau de caractères	Tableau de 1 à 9 utilisé pour comparer les valeurs lors de l'affichage des valeurs possibles (scénario 2)

etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.
tabBoolVerif[NB_CASES]	Tableau de booléen	Le tableau dans lequel les valeurs incompatibles présentent dans le tableau tabValIncomp seront égale à faux, vrai sinon. Utiliser pour trouver les valeurs possible (scénario 2)

Afficher les règles du jeu, la grille et le menu d’évolution de la partie

Cette étape consiste à saisir une proposition et faire une première vérification dessus.

Algorithme



Description des sous-problèmes

Cette partie d'algorithme est divisé en deux étapes :

- La saisie de la proposition
- La première vérification de la proposition. La vérification est faite pour que l'utilisateur ne rentre pas une proposition où le x ou le y ou le i n'appartiennent pas à l'intervalle [1…9] ou bien le joueur souhaite faire une proposition qui remplacerai un chiffre de la grille de départ.

VOIR afficherGrille ! (lien cliquable)

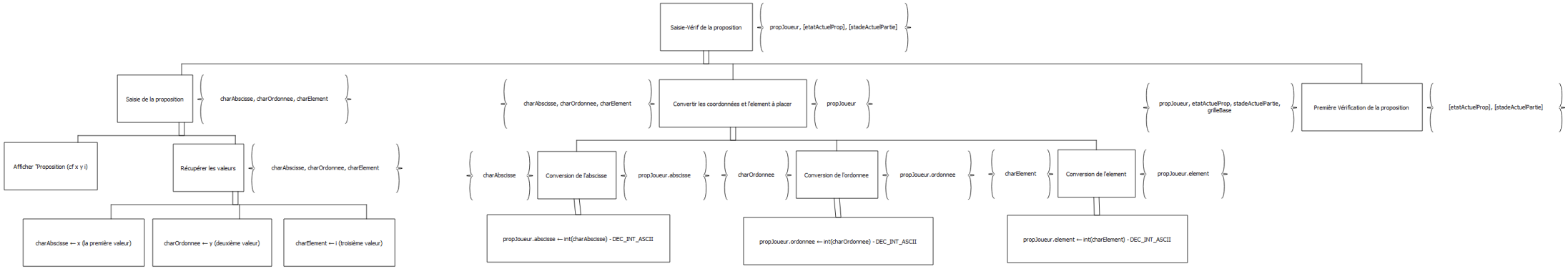
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.

Saisie-Verif de la proposition

Cette étape consiste à saisir et vérifier la proposition

Algorithme



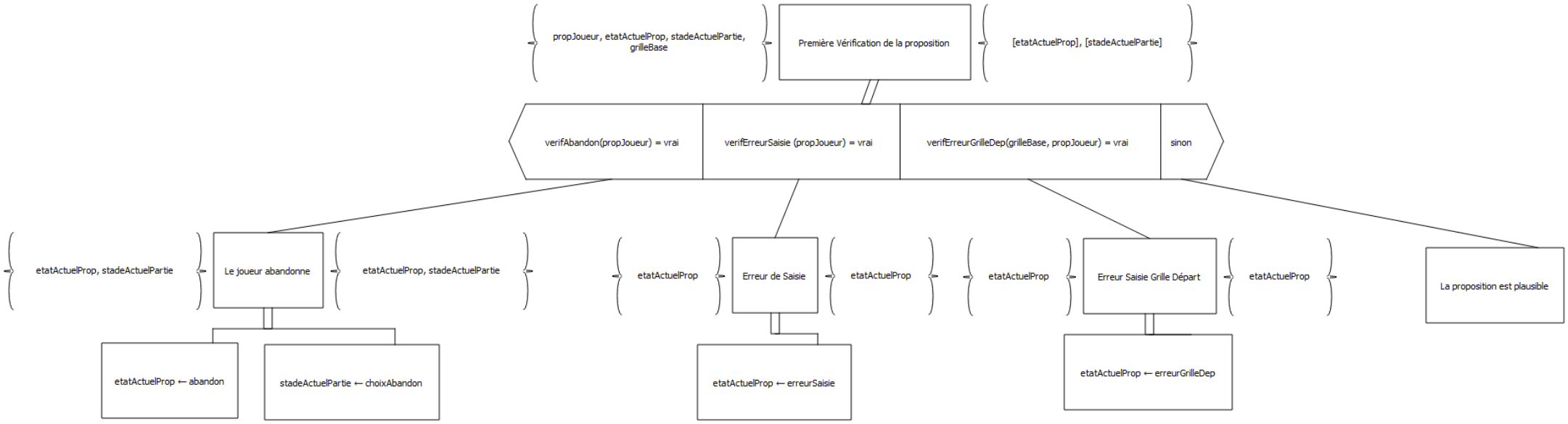
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
charAbscisse	Caractère	L'abscisse de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charOrdonnee	Caractère	L'ordonnée de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charElement	Caractère	L'élément de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.

Première vérification de la proposition

Cette étape consiste à effectuer une première vérification sur la saisie/proposition du joueur

Algorithme



VOIR verifAbandon ! (lien cliquable)
VOIR verifErreurSaisie ! (lien cliquable)
VOIR verifErreurGrilleDep ! (lien cliquable)

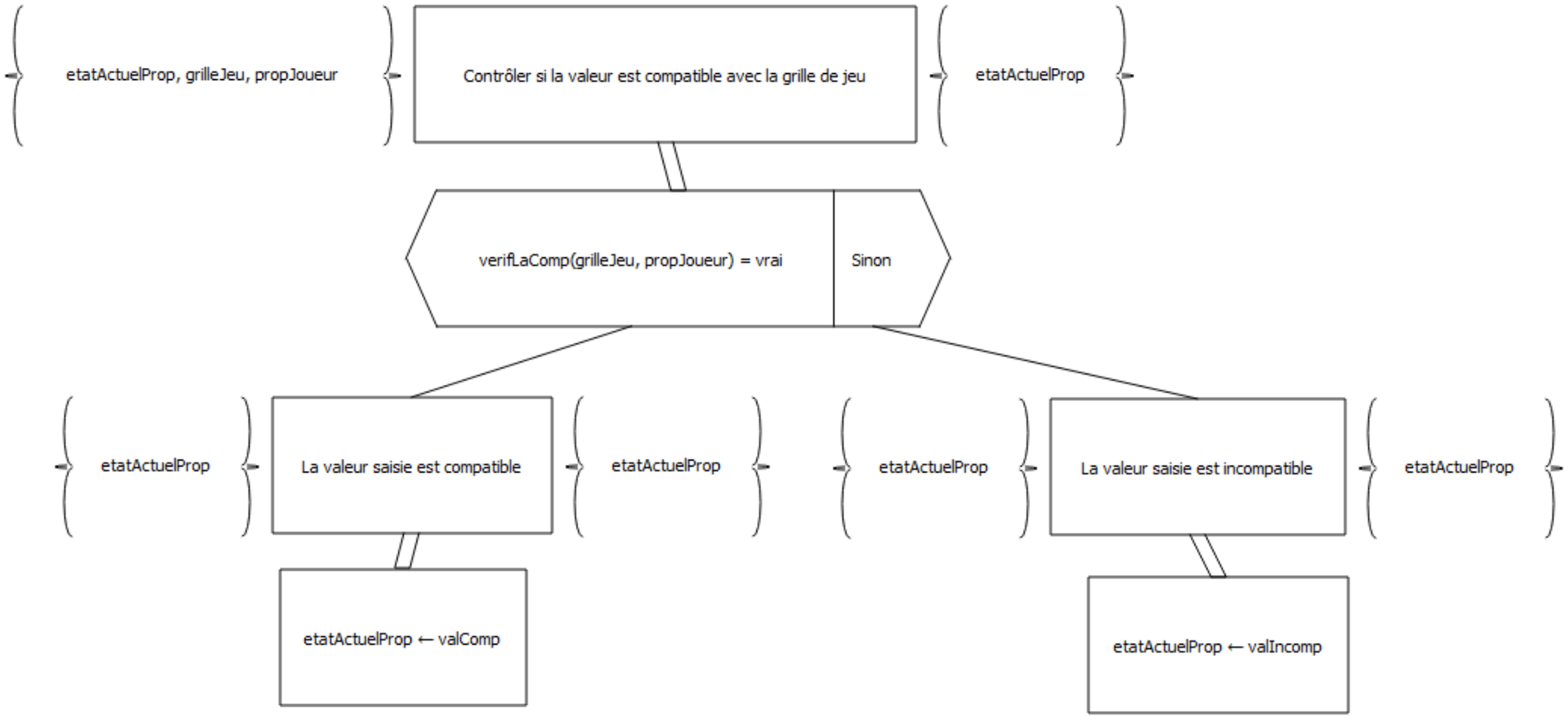
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.

Contrôler si la valeur est compatible avec la grille de jeu

Cette étape consiste à contrôler la compatibilité de la proposition saisie.

Algorithme



VOIR verifLaComp ! (lien cliquable)

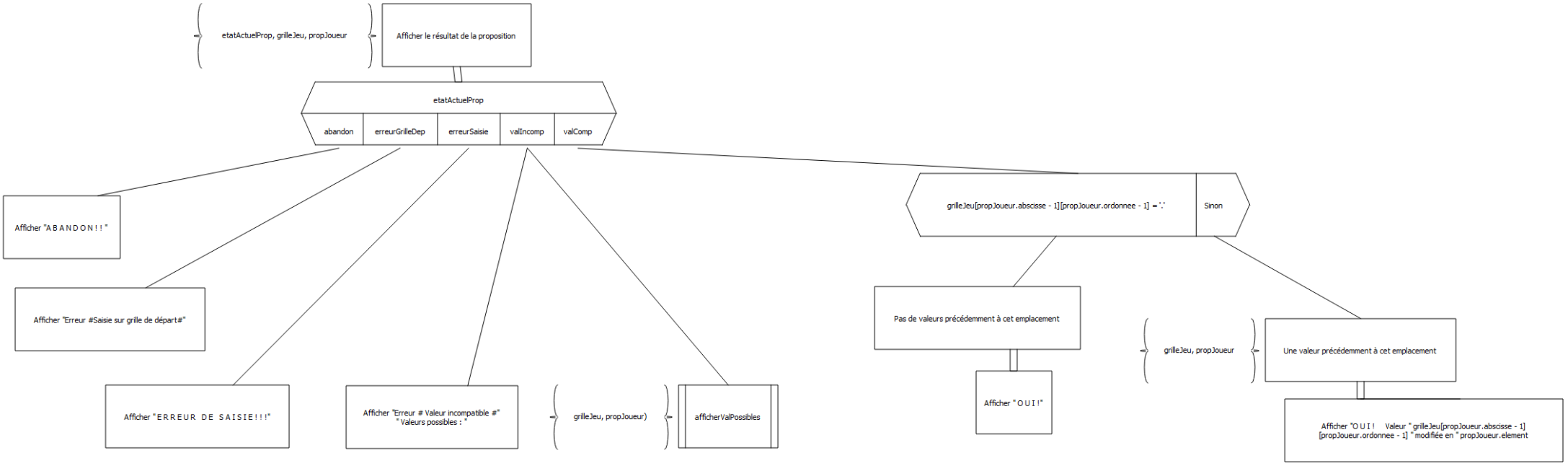
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.

Afficher le résultat de la proposition

Cette étape consiste à afficher les différents résultats de la proposition

Algorithme



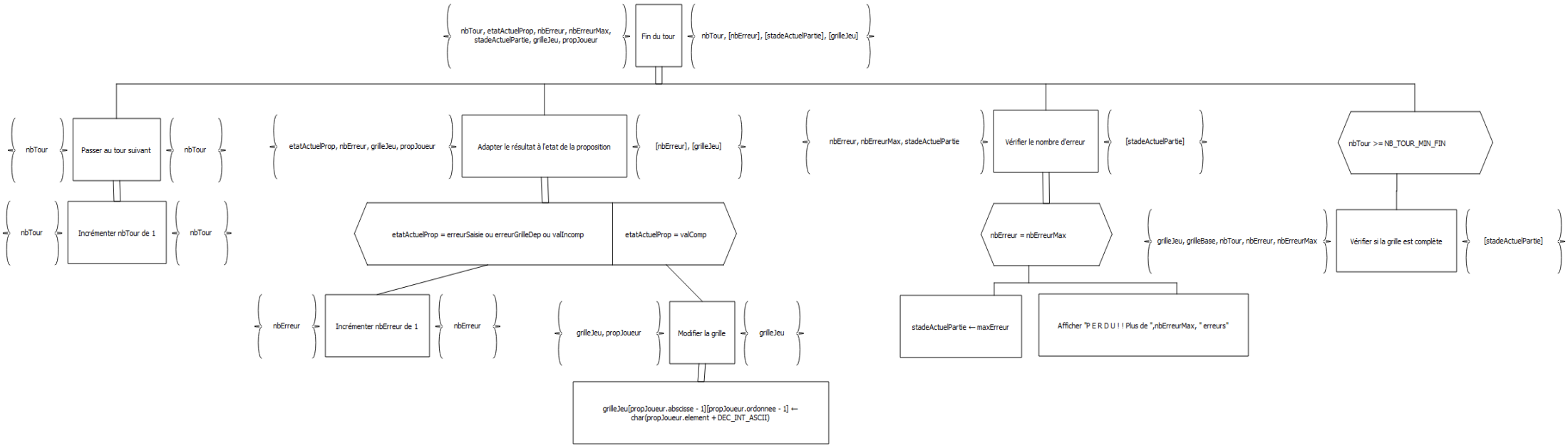
VOIR afficherValPossibles ! (lien cliquable)

Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
tabValIncomp[NB_CASES]	Tableau de caractères	Le tableau dans lequel on va ajouter (sans doublon) toutes les valeurs trouvées sur ligne, colonne et carré pour des coordonnées [x, y] rentrées. Ce tableau est utilisé pour afficher les valeurs possibles lorsque l'utilisateur se trompe (scénario 2)
tabValComp[NB_CASES]	Tableau de caractères	Tableau de 1 à 9 utilisé pour comparer les valeurs lors de l'affichage des valeurs possibles (scénario 2)
tabBoolVerif[NB_CASES]	Tableau de booléen	Le tableau dans lequel les valeurs incompatibles présentent dans le tableau tabValIncomp seront égale à faux, vrai sinon. Utiliser pour trouver les valeurs possible (scénario 2)
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.

Fin du tour

Cette étape consiste à finir un tour de jeu


nbTour, etatActuelProp, nbErreur, nbErreurMax, stadeActuelPartie, grilleJeu, propJoueur
Fin du tour
nbTour, [nbErreur], [stadeActuelPartie], [grilleJeu]
nbTour
Passer au tour suivant
nbTour
nbTour
Incrémenter nbTour de 1
nbTour
etatActuelProp, nbErreur, grilleJeu, propJoueur
Adapter le résultat à l'etat de la proposition
[nbErreur], [grilleJeu]
etatActuelProp = erreurSaisie ou erreurGrilleDep ou valIncomp
etatActuelProp = valComp
nbErreur
Incrémenter nbErreur de 1
nbErreur
grilleJeu, propJoueur
Modifier la grille
grilleJeu
grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1] ← char(propJoueur.element + DEC_INT_ASCII)
nbErreur, nbErreurMax, stadeActuelPartie
Vérifier le nombre d'erreur
[stadeActuelPartie]
nbErreur = nbErreurMax
stadeActuelPartie ← maxErreur
Afficher "P E R D U ! ! Plus de ",nbErreurMax, " erreurs"
nbTour >= NB_TOUR_MIN_FIN
grilleJeu, grilleBase, nbTour, nbErreur, nbErreurMax
Vérifier si la grille est complète
[stadeActuelPartie]

Algorithme

Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine sur il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.

Vérifier si la grille est complète

Cette étape consiste à vérifier si la grille est complété. Pour permettre ou non la finalisation du jeu

Algorithme


grilleJeu, grilleBase, nbTour, nbErreur, nbErreurMax
Vérifier si la grille est complète
[stadeActuelPartie]
recherchePointGrille(grilleJeu) = faux
Effacer
nbTour, nbErreur, nbErreurMax, grilleJeu, grilleBase
Afficher les règles du jeu, la grille et le menu d'évolution de la partie
Afficher "B R A V O ! ! ! !"
stadeActuelPartie ← victoire
Afficher le titre en bleu et les regles du jeu
grilleJeu, grilleBase
Afficher la grille
nbTour, nbErreur, nbErreurMax
Afficher le menu d'évolution de la partie
afficherTitreRegles
grilleJeu, grilleBase
afficherGrille
nbTour
Afficher le nombre de tours
nbErreur, nbErreurMax
Afficher le nombre d'erreurs
Afficher "Tour "
Afficher nbTour
Afficher","
Afficher "Erreur : " nbErreur "/" nbErreurMax

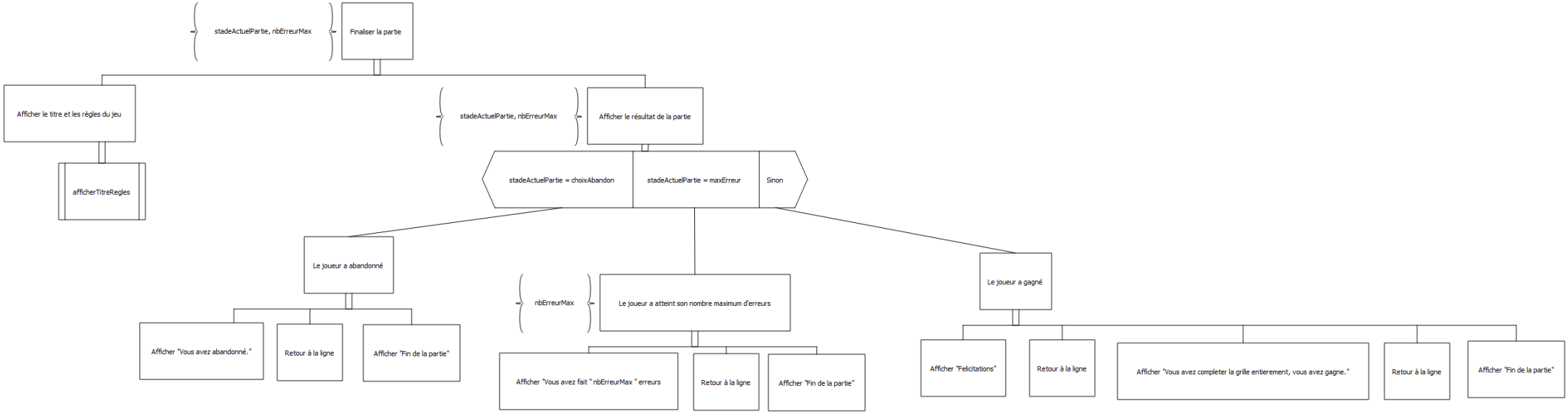
Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.

Finaliser la partie

Cette étape consiste à finaliser la partie, afficher suivant la condition de sortie de la partie, le résultat correspondant.

Algorithme

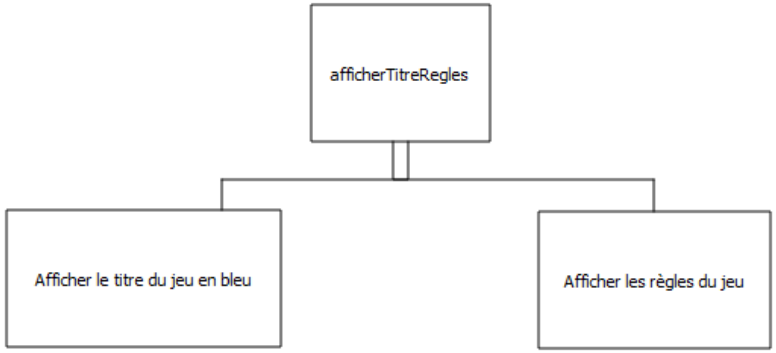

stadeActuelPartie, nbErreurMax
Finaliser la partie
Afficher le titre et les règles du jeu
afficherTitreRegles
stadeActuelPartie, nbErreurMax
Afficher le résultat de la partie
stadeActuelPartie = choixAbandon
stadeActuelPartie = maxErreur
Sinon
Le joueur a abandonné
Afficher "Vous avez abandonné."
Retour à la ligne
Afficher "Fin de la partie"
nbErreurMax
Le joueur a atteint son nombre maximum d'erreurs
Afficher "Vous avez fait " nbErreurMax " erreurs
Retour à la ligne
Afficher "Fin de la partie"
Le joueur a gagné
Afficher "Felicitations"
Retour à la ligne
Afficher "Vous avez completer la grille entierement, vous avez gagne."
Retour à la ligne
Afficher "Fin de la partie"

Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, abandon, erreurMax ou victoire
nbErreurMax	Entier court non signé	Le nombre d'erreur maximum que le joueur s'autorise.

Algorithmes secondaire (modules)

AfficherTitreRegles

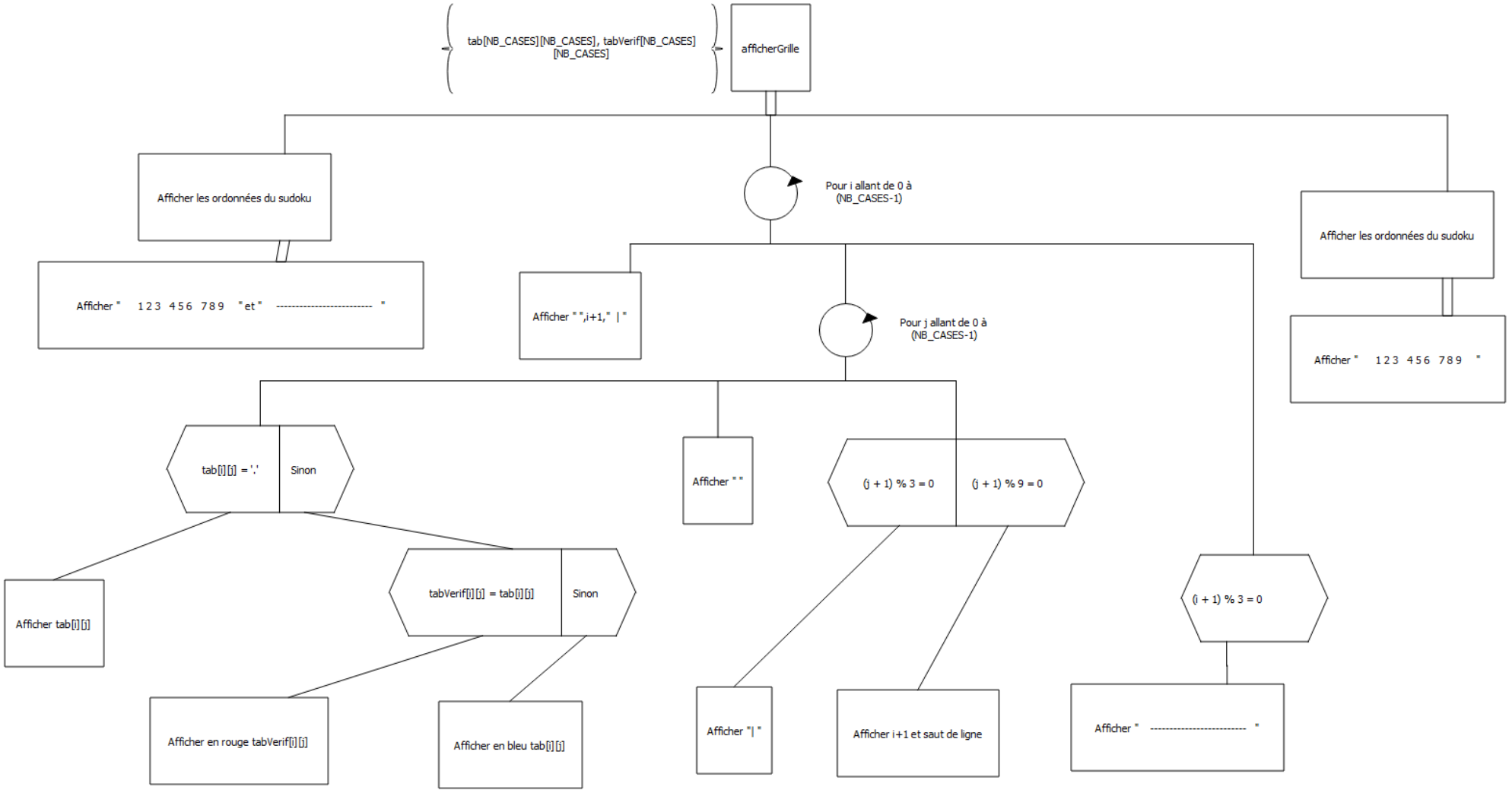


AfficherGrille

Ce sous-programme consiste afficher la grille dans lequel se déroule jeu.

RETOURNER sur Afficher les règles du jeu, la grille et le menu d’évolution de la partie ! (lien cliquable)

Algorithme


tab[NB_CASES][NB_CASES], tabVerif[NB_CASES][NB_CASES]
afficherGrille
Afficher les ordonnées du sudoku
Afficher " 1 2 3 4 5 6 7 8 9 "et " ---------- "
Pour i allant de 0 à (NB_CASES-1)
Afficher " ",i+1," | "
Pour j allant de 0 à (NB_CASES-1)
tab[i][j] = '.'
Sinon
Afficher tab[i][j]
tabVerif[i][j] = tab[i][j]
Sinon
Afficher en rouge tabVerif[i][j]
Afficher en bleu tab[i][j]
Afficher " "
(j + 1) % 3 = 0
(j + 1) % 9 = 0
Afficher "| "
Afficher i+1 et saut de ligne
Afficher les ordonnées du sudoku
Afficher " 1 2 3 4 5 6 7 8 9 "
(i + 1) % 3 = 0
Afficher " ---------- "

Dictionnaire des variables

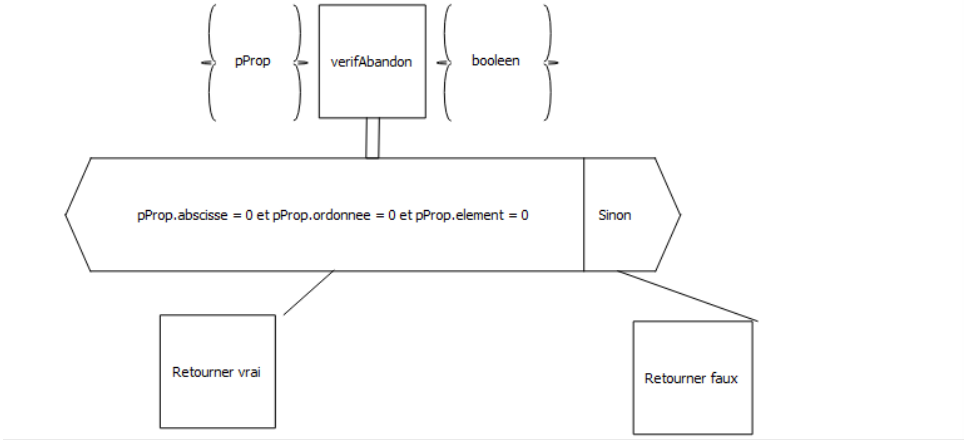
NOM	TYPE	SIGNIFICATION
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.
NB_CASES	Constant entier court non signé	Le nombre de cases / la taille des tableaux employés. Le nombre de cases de nos tableaux sont de taille de 9.

verifAbandon

Ce sous-programme consiste vérifier si il y a une condition d'abandon.

RETOURNER sur Première vérification de la proposition ! (lien cliquable)

Algorithme



Dictionnaire des variables

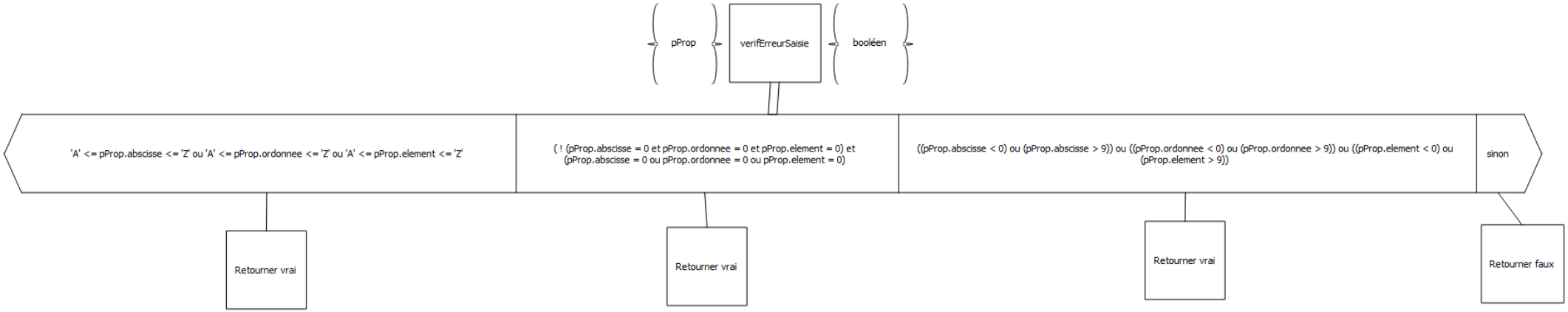
NOM	TYPE	SIGNIFICATION
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.

verifErreurSaisie

Ce sous-programme consiste vérifier si il y a une erreur de saisie.

RETOURNER sur Première vérification de la proposition ! (lien cliquable)

Algorithme



Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
charAbscisse	Caractère	L'abscisse de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)

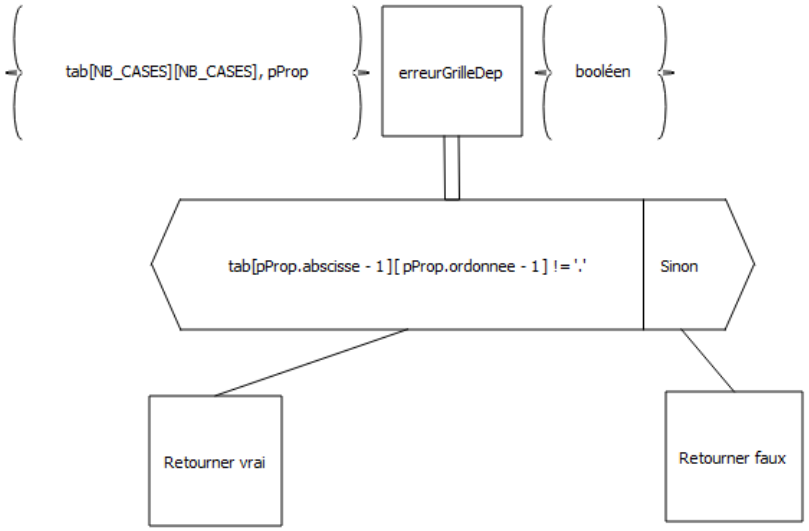
charOrdonnee	Caractère	L'ordonnée de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charElement	Caractère	L'élément de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
DEC_INT_ASCII	Constant entier court non signé	Le décalage, par rapport à la table Ascii, pour convertir un chiffre (en caractère) en chiffre (en entier). Le décalage est de 48.

verifErreurGrilleDep

Ce sous-programme consiste vérifier si la proposition se situe sur a grille de départ

RETOURNER sur Première vérification de la proposition ! (lien cliquable)

Algorithme



Dictionnaire des variables

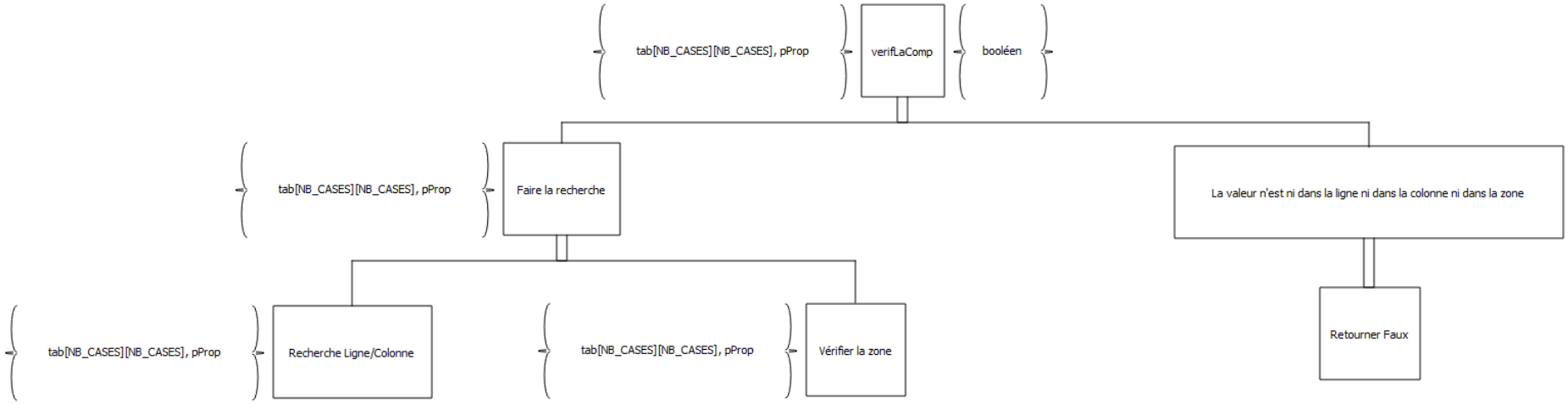
NOM	TYPE	SIGNIFICATION
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.

verifLaComp

Ce sous-programme consiste vérifier la compatibilité de la proposition.

RETOURNER sur Contrôler si la valeur est compatible avec la grille de jeu ! (lien cliquable)

Algorithme

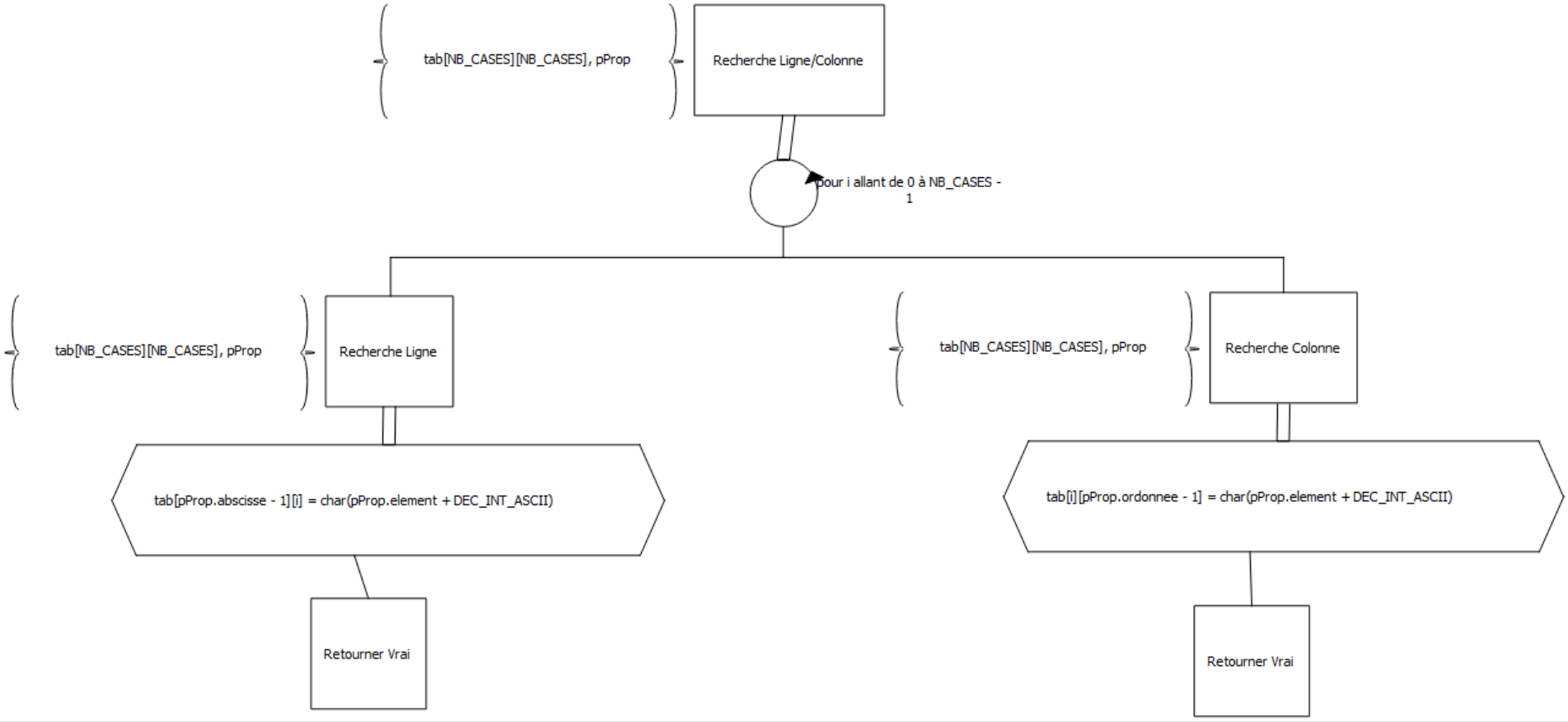


Dictionnaire des variables

NOM	TYPE	SIGNIFICATION
propJoueur	Enregistrement	Proposition du joueur qui consiste à saisir une abscisse, un ordonnée et un élément.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.

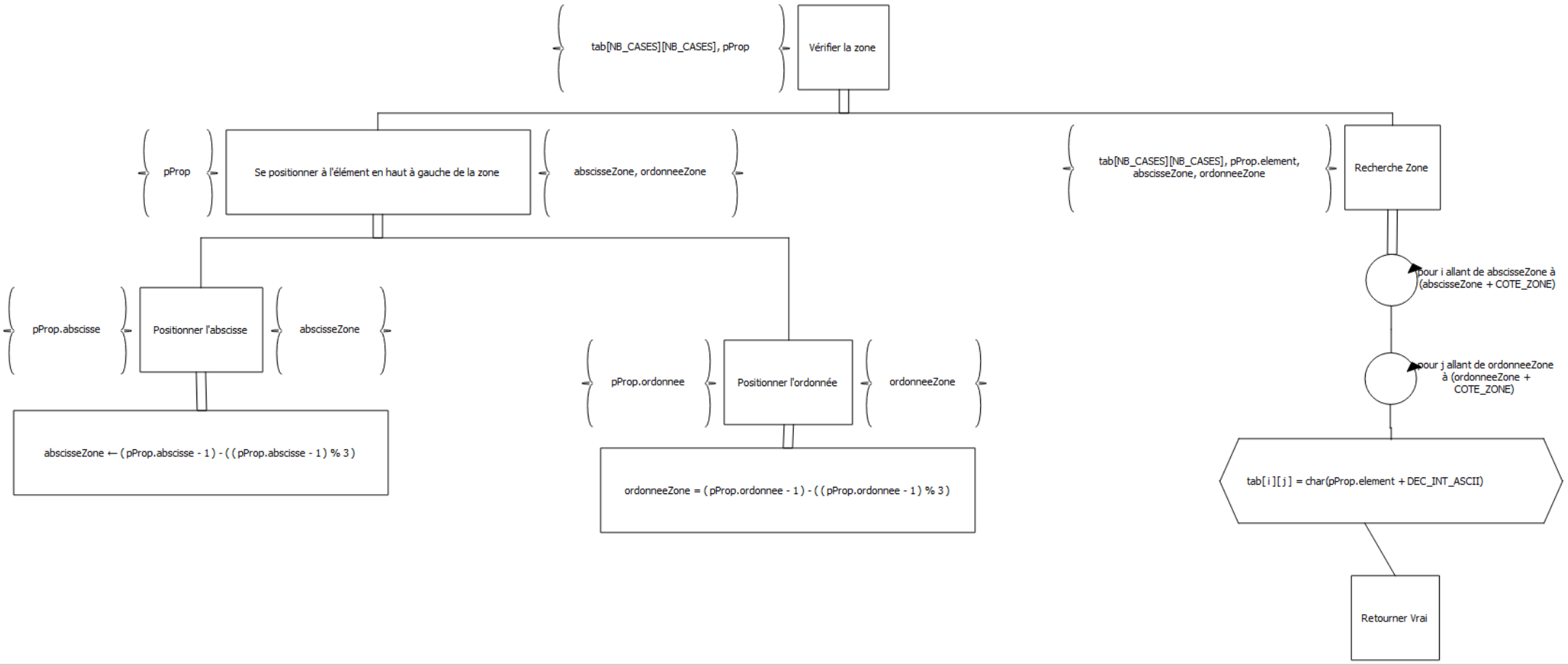
-- verifLaComp – Ligne/Colonne

Algorithme



-- verifLaComp -- Vérifier la zone

Algorithme

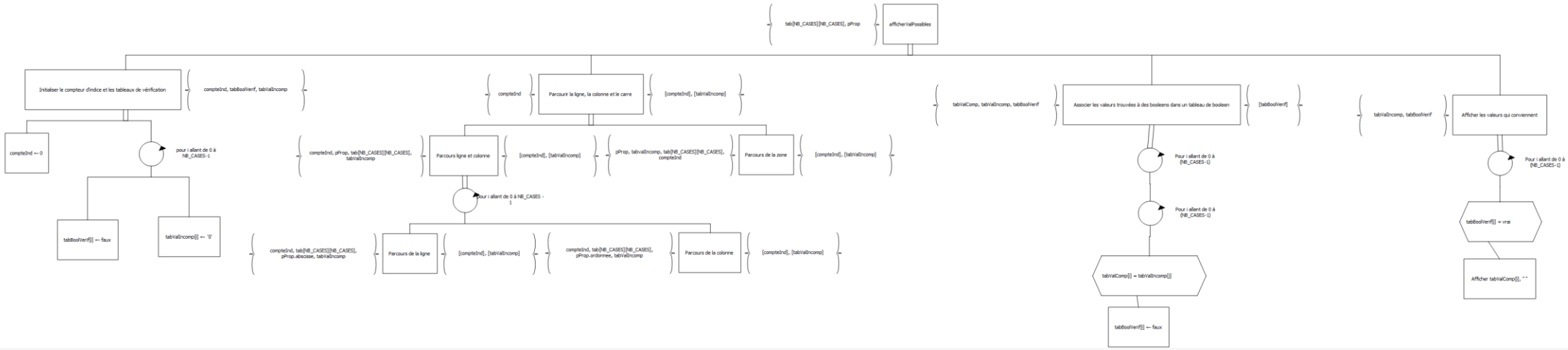


afficherValPossibles

Ce sous-programme consiste afficher les valeurs possibles

RETOURNER sur Afficher l'état de la proposition ! (lien cliquable)

Algorithme



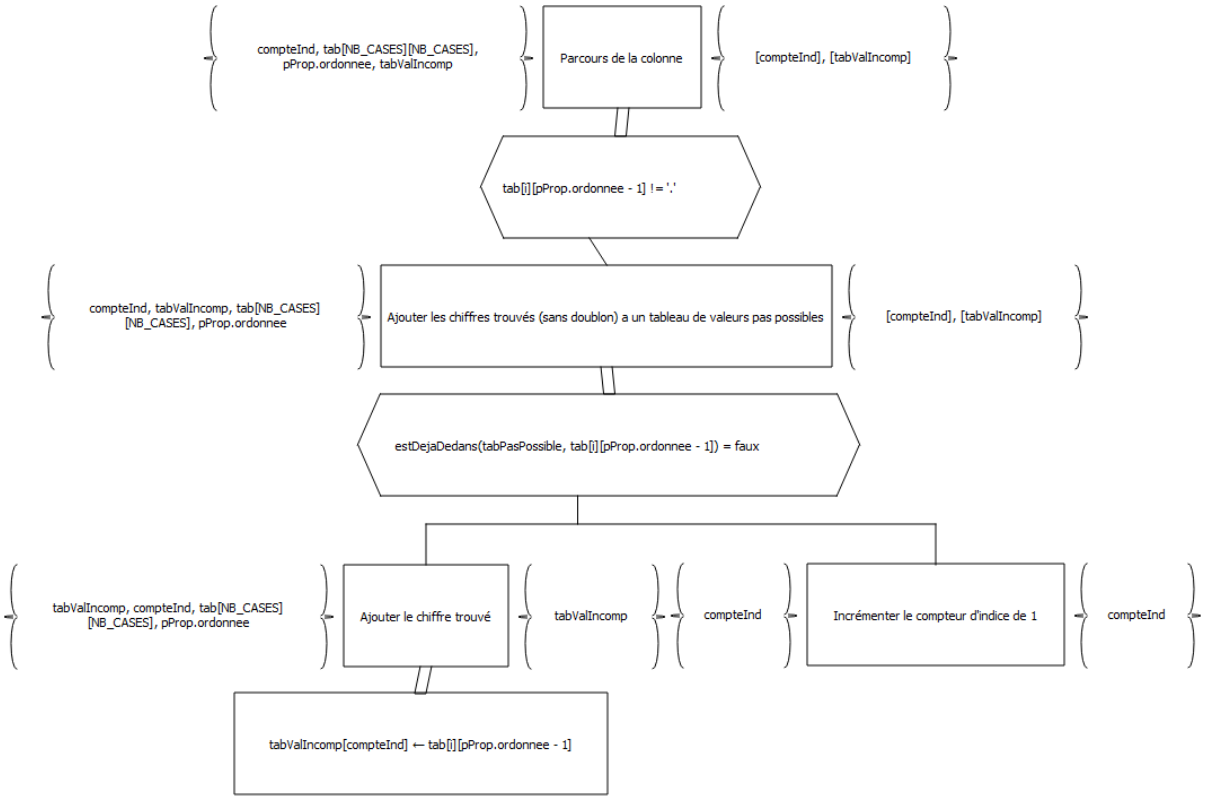
-- afficherValPossibles -- Parcourir la ligne

Algorithme

-- afficherValPossibles -- Parcours la colonne

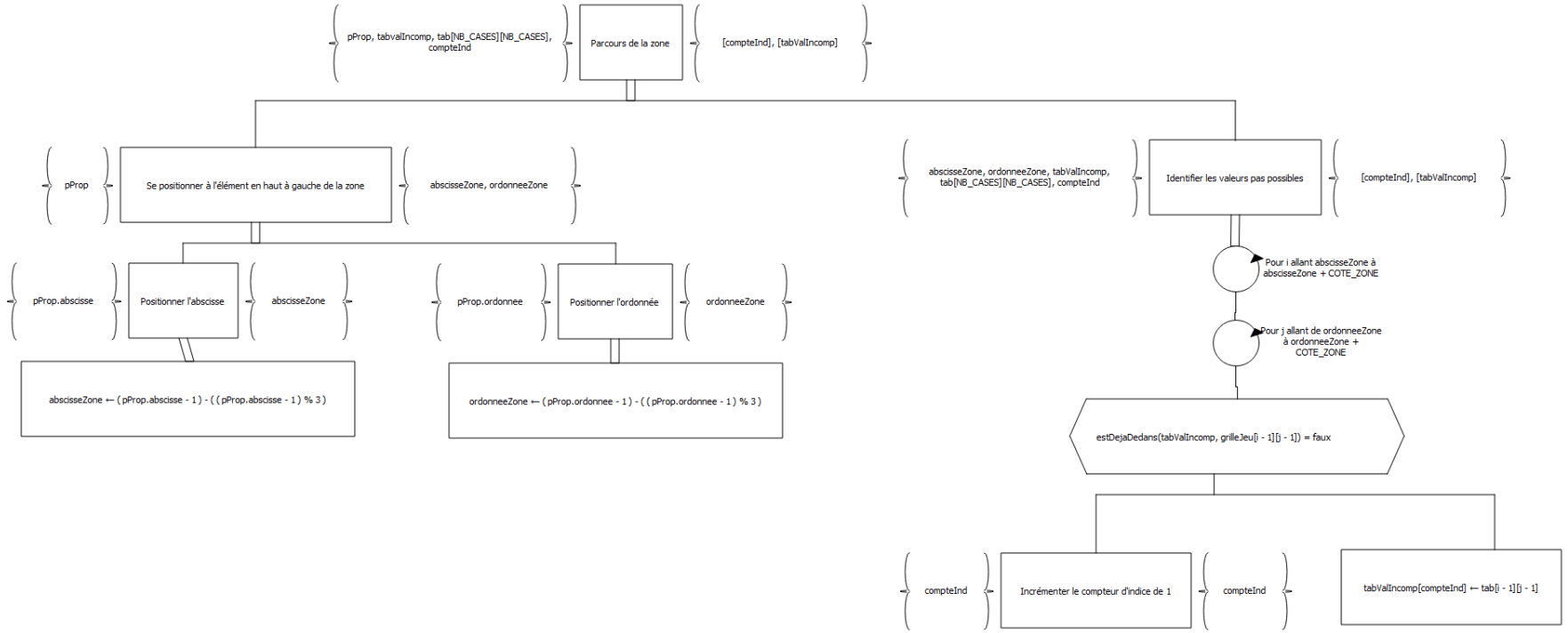
VOIR estDejaDedans ! (lien cliquable)

Algorithme



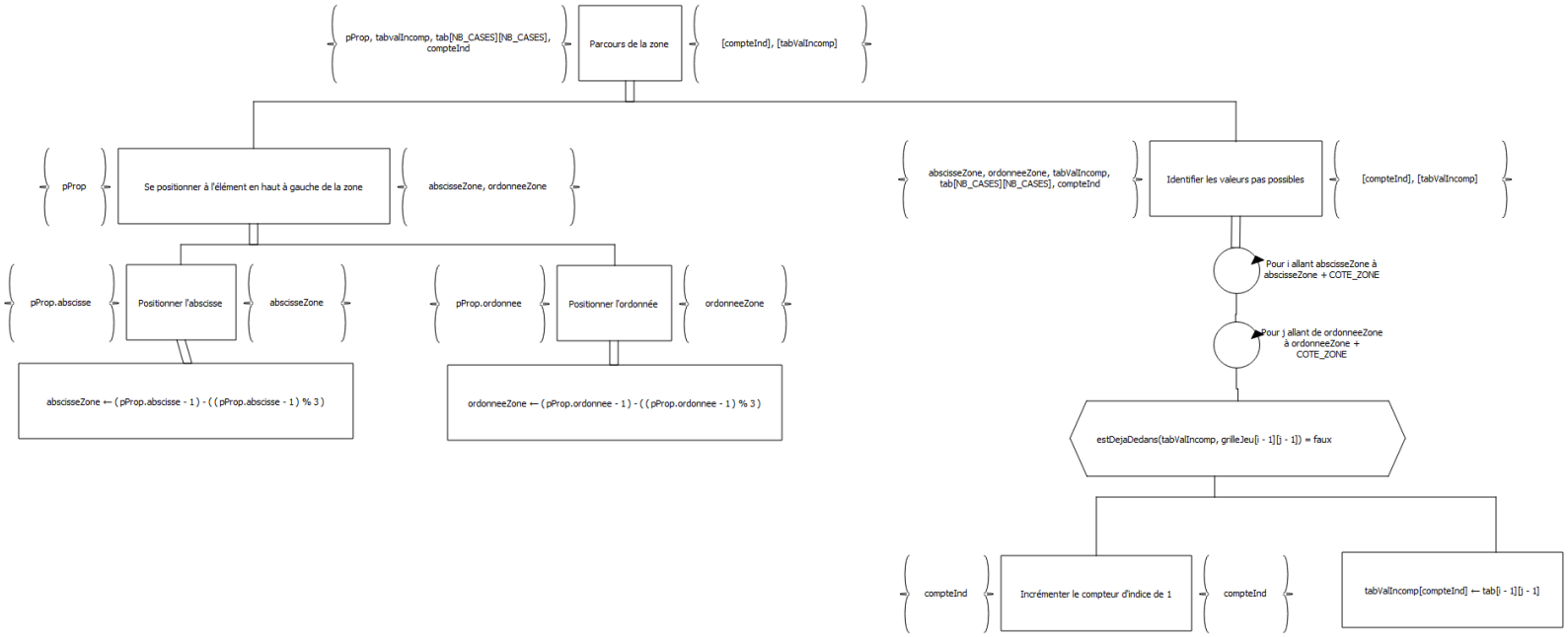
-- afficherValPossibles -- Parcours de la zone

Algorithme



-- afficherValPossibles -- Parcours de la zone

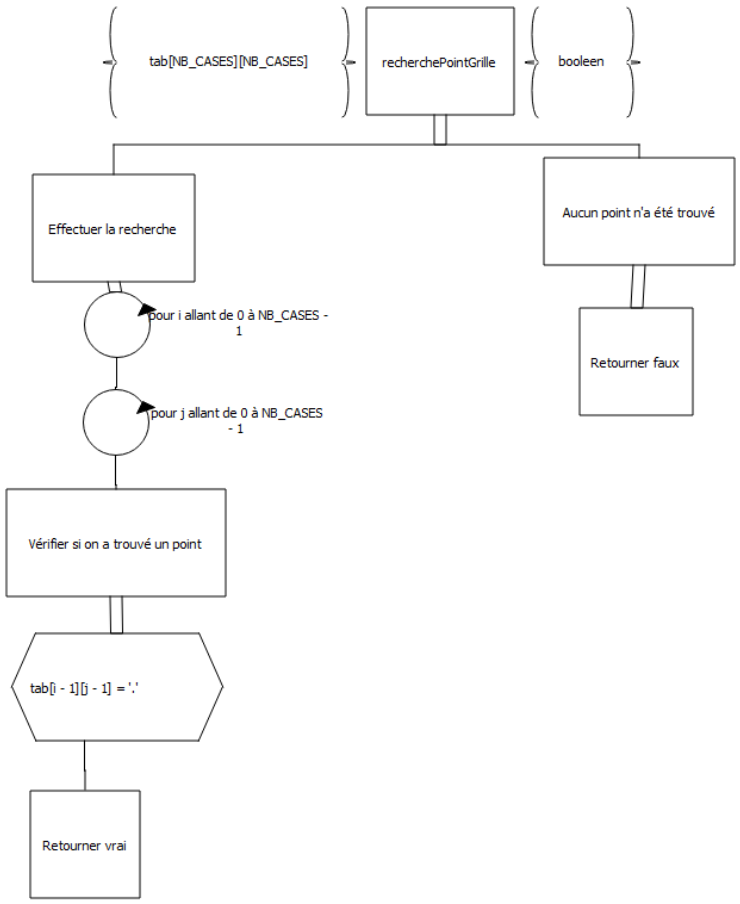
Algorithme



rechercherPointGrille

Ce sous-programme consiste à vérifier la condition de fin de jeu. On recherche un oint restant dans la grille.

Algorithme

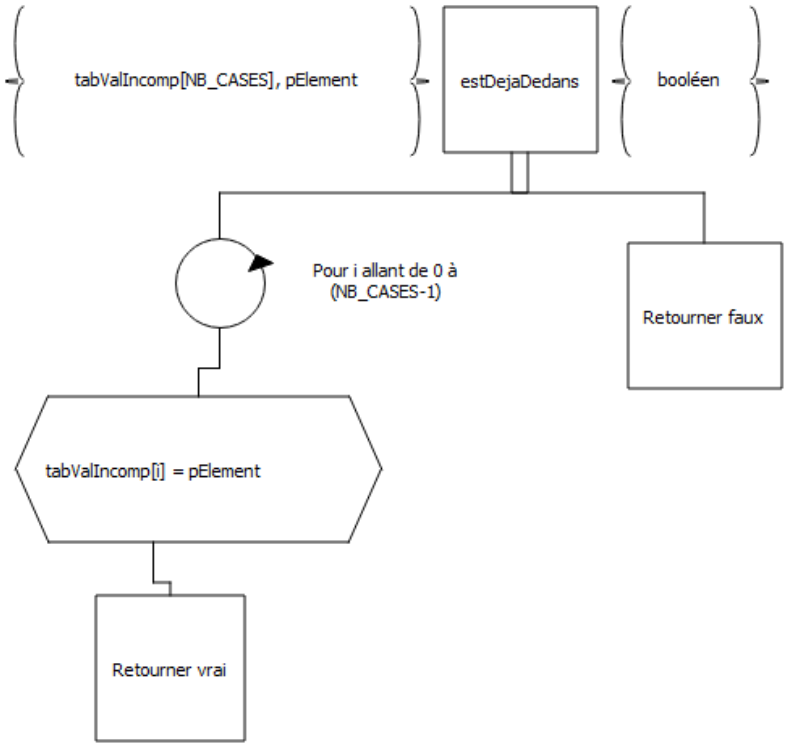

tab[NB_CASES][NB_CASES]
recherchePointGrille
booleen
Effectuer la recherche
Aucun point n'a été trouvé
pour i allant de 0 à NB_CASES - 1
Retourner faux
pour j allant de 0 à NB_CASES - 1
Vérifier si on a trouvé un point
tab[i - 1][j - 1] = '.'
Retourner vrai

estDejaDedans

Ce sous-programme consiste vérifier si un élément est déjà dans le tableau.

RETOURNER sur -- afficherValPossibles -- Parcours de la ligne ! (lien cliquable)

Algorithme



Etat de finalisation

SCENARIO	PRIS EN COMPTE
Saisie compatible avec la grille	Oui
Saisie incompatible avec la grille	Oui
Saisie sur une valeur ancienne	Oui
Saisie sur grille de départ	Oui
Erreur de saisie	Oui
Abandon	Oui
Dépasse le nombre d'erreurs	Oui
Grille complétée	Oui

Présentation globale / justification du découpage du code : fichiers, modules, sous-programmes...

Nous avons utilisé un module pour nos sous-programmes afin d'alléger le code principal (main.cpp). De plus nous avons essayé de regrouper des sous-programmes qui peuvent être réutilisés pour autres cas de Sudoku avec des jeux de données différents par exemple.

Ressentis personnels

GUIHENEUF Mattin

J'ai apprécié travailler sur ce projet, le travail de groupe était bien réparti et nous avons putous les deux coder et participer à la réflexion algorithmique du jeu. Le travail répond à nos attentes.

DUVIGNAU Yannis

Un projet (SAE) selon moi complet en termes de compétences : projet de groupe, conception et réflexion algorithmique, codage en C++ et enfin création d'un rapport final complet. J'ai bien apprécié le projet, il nous a permis de mettre en œuvre/pratique tout ce qu'on as pu voir en cours de développement (R1.01) depuis de début de la formation du BUT informatique. J'étais également avec un bon binôme : bonne répartition des tâches, bonne entente, bonne dynamique et chacun de nous est intervenu dans tous les aspects du projet (SAE) pour mettre à profit nos compétences respectives et personnelles.
Je pense que le travail rendu correspond à nos attentes à tous les deux même s'il y a tout de même des imperfections dans le travail rendu.
Je suis content et fier du rendu final !

Remarques

Informations que les étudiants souhaitent communiquer aux enseignants au sujet de cette SAE :

- Choix réalisés : Nous nous sommes d'abord posés sur les différentes situations initiales et avons déterminé les situations complémentaires qui nous paraissaient nécessaire pour le bon déroulement du jeu. Nous avons, par la suite, commencé par faire un algorithme du jeu en schématisant. Nous nous sommes ensuite attaqués à la décomposition des sous-problèmes petit à petit. Nous avons ensuite peaufiné notre algorithme dans les détails. Après cela, nous avons débuter le code en suivant notre solution algorithmique. Nous avons dû adapter notre algorithme car certaines choses n'étaient pas cohérentes. Après avoir finaliser le tout nous avons commencé le rapport par les situations initiales et complémentaires. Nous avons mis ensuite les algorithmes décomposés avec leur dictionnaire des données respectifs et leur description. Nous avons fini par les jeux d'essais.
- Difficultés : Nous avons confondu par étourderie l'abscisse et l'ordonnée et nous nous en sommes rendu compte après avoir tout terminé. Par manque de temps, nous n'avons pas corriger cette étourderie.
- Temps total passé **hors séances prévues à l'emploi du temps** : beaucoup trop ☹️ ! Nous avons passé beaucoup de nos soirées à travailler sur cette SAE. Nous avons même parfois fait passer le travail avant le sport.

Code C++

Choix d'organisation des fichiers composant le code source

On a utilisé un module pour nos sous-programmes afin d'alléger le code principal (main.cpp). De plus nous avons essayé de regrouper des sous-programmes qui peut être réutiliser dans d'autres jeux de donnée.

Liste des fichiers et contenu (en intention = quels types de sous-programmes il contient ; et/ou extension = liste des sous-programmes qu'il contient).

Code source

Nous certifions que le code source (sudoku.cpp, sudoku.h et le main.cpp) en C++ présenté ici est entièrement original et n'a pas été copié ou reproduit à partir de sources externes. Tous les droits d'auteur et les propriétés intellectuelles sont détenus par nous, les développeurs de ce produit. Ce code a été développé par nous (GUIHENEUF Mattin et DUVIGNAU Yannis).

```
/* Certification :
   Nous certifions que le code source main.cpp en C++ présenté ici est entièrement original et n'a pas été copié ou reproduit à partir de sources externes.
   Tous les droits d'auteur et les propriétés intellectuelles sont détenus par nous, les développeurs de ce produit.
   Ce code s'appuie uniquement sur les notions de programmation vues dans le cadre du module R1.01 - Initiation au développement (Partie 1 et Partie 2).
*/
```

```
/*
 * Fichier : main.cpp
 * But : Permet d'éxécuter le jeu du Sudoku
 * Auteurs : Yannis Duvignau et Mattin Guiheneuf
 * Date Dernière Modif : 18/01/2023
 */

#include <iostream>
#include "sudoku.h"
#include "game-tools.h"
using namespace std;

int main(void)
{
   // VARIABLES
   unsigned short int nbErreur; // Le nombre d'erreurs du joueur au cours du jeu.
   unsigned short int nbTour;   // Le nombre de tours accomplis par le joueur.

   enum stadePartie
   {
      enCours,
      choixAbandon,
      maxErreur,
      victoire
   };
   stadePartie stadeActuelPartie; /* Le stade actuel auquel appartient le joueur durant une partie.
                                     Il peut prendre les valeurs suivantes : enCours, choixAbandon, erreurMax ou victoire.*/

   string stringNbErreurMax; //

   int nbErreurMax;                    // Le nombre d'erreur maximum que le joueur s'autorise.
   const unsigned short int NBR_ERREUR_MIN = 3; // Le nombre d'erreurs minimum autorisé que le joueur peut saisir.
```

```
char charAbscisse; /* L'abscisse de la proposition récupérée après sa saisie.
                      Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères). */
char charOrdonnee; /* L'ordonnée de la proposition récupérée après sa saisie.
                      Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)*/
char charElement;  /* L'élément de la proposition récupérée après sa saisie.
                      Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)*/

const unsigned short int NB_TOUR_MIN_FIN = 51; /* Le nombre de tours minimum pour pouvoir compléter la grille.
                                                  Il a été calculé au préalable pour optimiser le jeu et éviter de faire des vérifications inutiles :
                                                  Il y a 81 cases et 30 chiffres pré-rentrées donc le joueur peut finir le jeu en 51 coups minimum soit 51 tours.*/

enum etatProp
{
   enTrait,
   abandon,
   erreurSaisie,
   erreurGrilleDep,
   valComp,
   valIncomp
};
etatProp etatActuelProp; /* L'état de la proposition actuel.
                            Détermine s'il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ,
                            la valeur est incompatible avec la grille (l'élément ne peut pas être placé),
                            la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement).
                            Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp. */

Prop propJoueur; // Proposition du joueur qui consiste à récupérer la saisie d'une abscisse, d'une ordonnée et d'un élément.

char grilleJeu[NB_CASES][NB_CASES] = {{'5', '3', '.', '.', '7', '.', '.', '.', '.'},
                                      {'6', '.', '.', '1', '9', '5', '.', '.', '.'},
                                      {'.', '9', '8', '.', '.', '.', '.', '6', '.'},
                                      {'8', '.', '.', '.', '6', '.', '.', '.', '3'},
```

```
                        {'4', '.', '.', '8', '.', '3', '.', '.', '1'},
                        {'7', '.', '.', '.', '2', '.', '.', '.', '6'},
                        {'.', '6', '.', '.', '.', '.', '2', '8', '.'},
                        {'.', '.', '.', '4', '1', '9', '.', '.', '5'},
                        {'.', '.', '.', '.', '8', '.', '.', '7', '9'}}; // Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
  char grilleBase[NB_CASES][NB_CASES] = {{'5', '3', '.', '.', '7', '.', '.', '.', '.'},
                        {'6', '.', '.', '1', '9', '5', '.', '.', '.'},
                        {'.', '9', '8', '.', '.', '.', '.', '6', '.'},
                        {'8', '.', '.', '.', '6', '.', '.', '.', '3'},
                        {'4', '.', '.', '8', '.', '3', '.', '.', '1'},
                        {'7', '.', '.', '.', '2', '.', '.', '.', '6'},
                        {'.', '6', '.', '.', '.', '.', '2', '8', '.'},
                        {'.', '.', '.', '4', '1', '9', '.', '.', '5'},
                        {'.', '.', '.', '.', '8', '.', '.', '7', '9'}}; // Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.
  char tabValComp[NB_CASES] = {'1', '2', '3', '4', '5', '6', '7', '8', '9'};          // Tableau de 1 à 9 utilisé pour comparer les valeurs lors de l'affichage des valeurs possibles (scénario 2)

  char tabValIncomp[NB_CASES]; /* Le tableau dans lequel on va ajouter (sans doublon) toutes les valeurs trouvées sur ligne, colonne et carré pour des coordonnées [x, y] rentrées.
                        Ce tableau est utilisé pour afficher les valeurs possibles lorsque l'utilisateur se trompe (scénario 2)*/
  bool tabBoolVerif[NB_CASES]; /* Le tableau dans lequel les valeurs incompatibles présentent dans le tableau tabValIncomp seront égale à faux, vrai sinon. Utiliser pour trouver les
valeurs possible (scénario 2)*/

  // TRAITEMENTS

  // Initialiser la partie >> nbErreur, nbTour, stadeActuelPartie, nbErreurMax supérieur ou égal à 3

  // Initialiser les paramètres de base >> nbErreur, nbTour, stadeActuelPartie

  nbErreur = 0;
  nbTour = 1;
  stadeActuelPartie = enCours;

  // Afficher le titre en bleu et les règles
```

```
afficherTitreRegles();

// Saisie et Vérification du nombre d'erreurs autorisées >> nbErreurMax supérieur ou égal à 3

// Saisie du nombre d'erreurs maximum affiché en rouge >> stringNbErreurMax
cout << endl
    << endl
    << "Le nombre d'erreurs maximum autorisees doit etre superieur ou egale a 3. " << endl
    << "Si vous entrez autre chose qu'une valeur entiere superieure a 3, le nombre d'erreurs maximum autorisees sera initialise par defaut a 3" << endl
    << endl;
afficherTexteEnCouleur(" Choix du nombre d'erreurs autorisees (>= 3) : ", rouge, false);
cin >> stringNbErreurMax;

// stringNbErreurMax >> Vérifier que le nombre d'erreurs maximum saisi soit valable >> nbErreurMax
// Vérifier que le nombre ne soit pas un nombre négatif
if (stringNbErreurMax[0] == '-')
{
   // nbErreurMax n'est pas validé
   nbErreurMax = NBR_ERREUR_MIN;
}
else
{
   // Faire les vérifications sur stringNbErreurMax
   for (unsigned int i = 0; i < stringNbErreurMax.size(); i++)
   {
      // Vérifier que le nombre d'erreurs maximum, saisi par le joueur, soit bien un nombre entier
      if (stringNbErreurMax[i] >= 'a' && stringNbErreurMax[i] <= 'z')
      {
         // nbErreurMax n'est pas validé
         nbErreurMax = NBR_ERREUR_MIN;
      }
      else if (stringNbErreurMax[i] == '.' || stringNbErreurMax[i] == ',')
      {
```

```
            // nbErreurMax n'est pas validé
            nbErreurMax = NBR_ERREUR_MIN;
            break;
        }
        else
        {
            nbErreurMax = nbErreurMax + (int(stringNbErreurMax[i]) - static_cast<int>(DEC_INT_ASCII)) * (carre(10, static_cast<int>(stringNbErreurMax.size()) - static_cast<int>(i) - 1));
        }
        // Vérif nbErreurMax
        if (nbErreurMax < NBR_ERREUR_MIN)
        {
            nbErreurMax = NBR_ERREUR_MIN;
        }
    }
}

afficherTexteEnCouleur("                    Vous pouvez faire jusqu'a ", rouge, false);
afficherNombreEnCouleur(nbErreurMax, rouge, false);
afficherTexteEnCouleur(" erreurs", rouge, true);

// Attendre une action du joueur pour commencer la partie

cout << endl
     << "Appuyer sur une touche pour continuer ...";
pause();
effacer();

/* nbErreur, nbTour, stadeActuelPartie, nbErreurMax supérieur ou égal à 3, NB_TOUR_MIN_FIN,  tabValIncomp, tabValComp, tabBoolVerif,
   grilleJeu, grilleBase, NB_CASES >> Jouer la Partie >> stadeActuelPartie */

do
{
    // Initialiser les paramètres de gestion d'un tour
```

```
etatActuelProp = enTrait;

/* nbTour, nbErreur, nbErreurMax, grilleJeu, grilleBase, etatActuelProp, stadeActuelPartie,
  tabValIncomp, tabValComp, tabBoolVerif >> Jouer un tour */

// nbTour, nbErreur, nbErreurMax, grilleJeu, grilleBase >> Afficher les règles du jeu, la grille et le menu d'évolution de la partie

// Afficher le titre en bleu et les règles du jeu
afficherTitreRegles();

// grilleJeu, grilleBase >> Afficher la grille
afficherGrille(grilleJeu, grilleBase);

// nbTour, nbErreur, nbErreurMax >> Afficher le menu d'évolution de la partie

// nbTour >> Afficher le nombre de tours
cout << " Tour " << nbTour << ",";

// nbErreur, nbErreurMax >> Afficher le nombre d'erreurs
cout << " Erreur : " << nbErreur << "/" << nbErreurMax << ", ";

cout << "Abandon" << endl;

// Saisie-Vérif de la proposition >> propJoueur, [etatActuelProp], [stadeActuelPartie]

// Saisie de la proposition >> charAbscisse, charOrdonnee, charElement
cout << " Proposition (cf. x y i) ? ";
cin >> charAbscisse;
cin >> charOrdonnee;
cin >> charElement;

// charAbscisse, charOrdonnee, charElement >> Conversion des coordonnées et de l'élément à placer >> propJoueur
```

```
// charAbscisse >> Conversion de l'abscisse >> propJoueur.abscisse
propJoueur.abscisse = static_cast<unsigned short int>(int(charAbscisse) - static_cast<int>(DEC_INT_ASCII));

// charOrdonnee >> Conversion de l'ordonnée >> propJoueur.ordonnee
propJoueur.ordonnee = static_cast<unsigned short int>(int(charOrdonnee) - static_cast<int>(DEC_INT_ASCII));

// charElement >> Conversion de l'élément >> propJoueur.element
propJoueur.element = static_cast<unsigned short int>(int(charElement) - static_cast<int>(DEC_INT_ASCII));

// propJoueur, etatActuelProp, stadeActuelPartie, grilleBase >> Vérification de la proposition >> [etatActuelProp], [stadeActuelPartie]
if (verifAbandon(propJoueur))
{
   // etatActuelProp, stadeActuelPartie >> Le joueur abandonne >> etatActuelProp, stadeActuelPartie
   etatActuelProp = abandon;
   stadeActuelPartie = choixAbandon;
}
else if (verifErreurSaisie(propJoueur))
{
   // etatActuelProp >> Erreur de Saisie >> etatActuelProp
   etatActuelProp = erreurSaisie;
}
else if (verifErreurGrilleDep(grilleBase, propJoueur))
{
   // etatActuelProp >> Erreur Saisie Grille Départ >> etatActuelProp
   etatActuelProp = erreurGrilleDep;
}
else
{
   // La proposition est plausible
}

if (etatActuelProp == enTrait)
{
```

```
   // etatActuelProp, grilleJeu, propJoueur >> Contrôler si la valeur est compatible avec la grille du jeu >> etatActuelProp
   if (verifLaComp(grilleJeu, propJoueur))
   {
      // etatActuelProp >> La valeur saisie est compatible >> etatActuelProp
      etatActuelProp = valComp;
   }
   else
   {
      // etatActuelProp >> La valeur saisie est incompatible >> etatActuelProp
      etatActuelProp = valIncomp;
   }
}

// etatActuelProp, grilleJeu, tabValComp, tabValIncomp, tabBoolVerif, propJoueur >> Afficher le résultat de la proposition

switch (etatActuelProp)
{
case abandon:
   cout << " A B A N D O N ! !\n";
   break;

case erreurGrilleDep:
   cout << " Erreur # Saisie sur grille de depart #\n";
   break;

case erreurSaisie:
   cout << " E R R E U R  D E  S A I S I E  ! ! !\n";
   break;

case valIncomp:
   cout << " Erreur # Valeur incompatible #" << endl
        << " Valeurs possibles : ";
   afficherValPossibles(grilleJeu, propJoueur);
```

```
    break;

case valComp:
    if (grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1] == '.')
    {
        // Pas de valeurs précedemment à cet emplacement
        cout << " O U I !";
    }
    else
    {
        // grilleJeu, propJoueur >> Une valeur précedemment à cet emplacement
        cout << " O U I !    Valeur " << grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1] << " modifiee en " << propJoueur.element;
    }
default:
    break;
}

// nbTour, etatActuelProp, nbErreur, nbErreurMax, stadeActuelPartie, grilleJeu, propJoueur >> Fin du tour

// nbTour >> Passer au tour suivant >> nbTour
nbTour = static_cast<unsigned short int>(nbTour + 1);

// etatActuelProp, nbErreur, grilleJeu, propJoueur >> Adapter le résultat à l'état de la proposition >> [nbErreur], [grilleJeu]

if (etatActuelProp == erreurSaisie || etatActuelProp == erreurGrilleDep || etatActuelProp == valIncomp)
{
    // nbErreur >> Incrémenter nbErreur de 1 >> nbErreur
    nbErreur = static_cast<unsigned short int>(nbErreur + 1);
}
else if (etatActuelProp == valComp)
{
    // grilleJeu >> Modifier la grille >> grilleJeu
    grilleJeu[propJoueur.abscisse - 1][propJoueur.ordonnee - 1] = char(propJoueur.element + DEC_INT_ASCII);
```

```
}

// nbErreur, nbErreurMax, stadeActuelPartie >> Vérifier le nombre d'erreurs >> [stadeActuelPartie]
if (nbErreur == nbErreurMax)
{
   stadeActuelPartie = maxErreur;
   cout << "P E R D U  ! ! ! plus de " << nbErreurMax << " erreurs" << endl;
}

if (nbTour > NB_TOUR_MIN_FIN)
{
   // grilleJeu >> Vérifier si la grille est complète >> [stadeActuelPartie]
   if (!rechercheDansGrille(grilleJeu))
   {
      effacer();
      // nbTour, nbErreur, nbErreurMax, grilleJeu, grilleBase >> Afficher les règles du jeu, la grille et le menu d'évolution de la partie

      // Afficher le titre en bleu et les règles du jeu
      afficherTitreRegles();

      // grilleJeu, grilleBase >> Afficher la grille
      afficherGrille(grilleJeu, grilleBase);

      // nbTour, nbErreur, nbErreurMax >> Afficher le menu d'évolution de la partie

      // nbTour >> Afficher le nombre de tours
      cout << " Tour " << nbTour << ",";

      // nbErreur, nbErreurMax >> Afficher le nombre d'erreurs
      cout << " Erreur : " << nbErreur << "/" << nbErreurMax << endl;

      stadeActuelPartie = victoire;
```

```
            cout << " B R A V O  ! ! ! !";
        }
    }

    // Attendre que le joueur appuie sur une touche
    cout << "\n Appuyez sur une touche pour continuer ...";
    pause();
    effacer();

} while (stadeActuelPartie == enCours);

// Finaliser la partie

// Afficher le titre et les règles du jeu
afficherTitreRegles();

// Afficher le résultat de la partie
if (stadeActuelPartie == choixAbandon)
{
    // Le joueur a abandonné
    cout << "Vous avez abandonne." << endl
         << "Fin de la partie\n";
}
else if (stadeActuelPartie == maxErreur)
{
    // Le joueur a atteint son nombre maximum d'erreurs
    cout << "Vous avez fait " << nbErreurMax << " erreurs, vous avez perdu." << endl
         << "Fin de la partie\n";
}
else
{
    // Le joueur a gagné
    cout << "Felicitations" << endl
```

```
            << "Vous avez completer la grille entierement, vous avez gagne." << endl
            << "Fin de la partie\n";
    }

    return 0;
}
```

Annexe 1 – maquettes d'écran prévues dans les spécifications

Situations initiales

Jeux d'essais 1

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 3 1 1
O U I !
Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 2

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 5 5 5
O U I !
Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 3

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 1 6 4
O U I !
Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 4

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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 5 8 4
Erreur # Valeur incompatible #
Valeurs possibles : 2 5 9
Appuyez sur une touche pour continuer ...
```

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

Tour 3, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 5

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

Tour 3, Erreur : 1/3, Abandon
Proposition (cf. x y i) ? 5 8 4
Erreur # Valeur incompatible #
Valeurs possibles : 2 5 9
Appuyez sur une touche pour continuer ...
```

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

Tour 4, Erreur : 2/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 6

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

Tour 3, Erreur : 1/3, Abandon
Proposition (cf. x y i) ? 2 2 1
Erreur # Valeur incompatible #
Valeurs possibles : 2 4 7
Appuyez sur une touche pour continuer ...
```

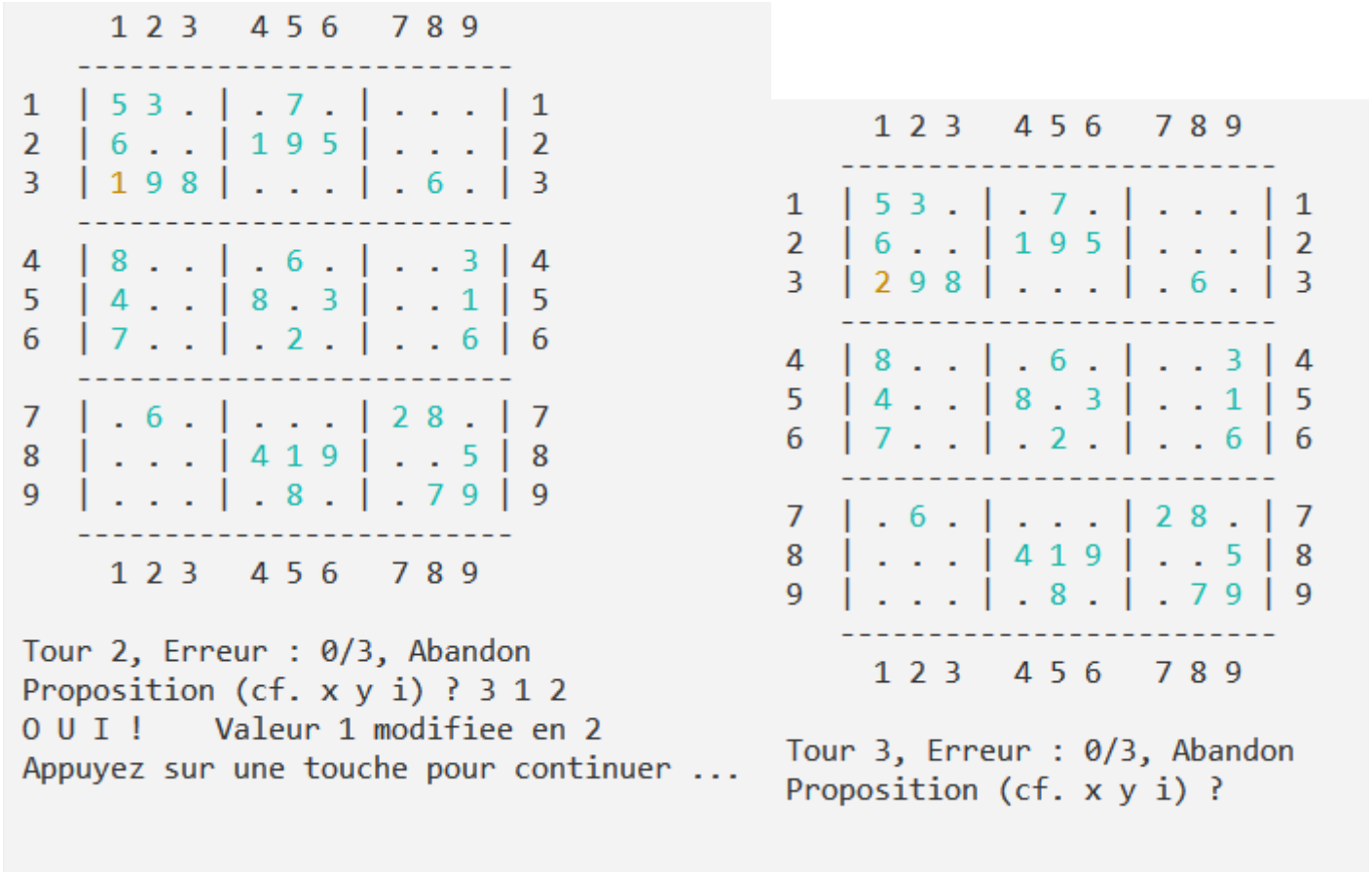
```
    1 2 3   4 5 6   7 8 9
  -------------------------
1 | 5 3 . | . 7 . | . . . | 1
2 | 6 . . | 1 9 5 | . . . | 2
3 | 1 9 8 | . . . | . 6 . | 3
  -------------------------
4 | 8 . . | . 6 . | . . 3 | 4
5 | 4 . . | 8 . 3 | . . 1 | 5
6 | 7 . . | . 2 . | . . 6 | 6
  -------------------------
7 | . 6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

Tour 4, Erreur : 2/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 7

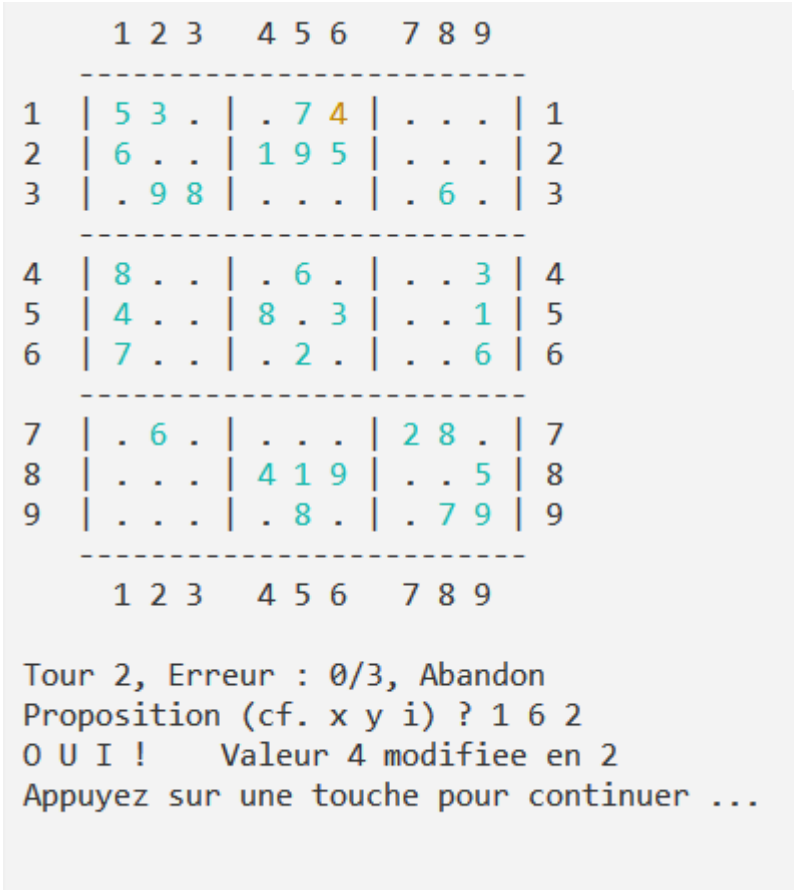
```
    1 2 3   4 5 6   7 8 9
  -------------------------
1 | 5 3 . | . 7 . | . . . | 1
2 | 6 . . | 1 9 5 | . . . | 2
3 | 1 9 8 | . . . | . 6 . | 3
  -------------------------
4 | 8 . . | . 6 . | . . 3 | 4
5 | 4 . . | 8 . 3 | . . 1 | 5
6 | 7 . . | . 2 . | . . 6 | 6
  -------------------------
7 | . 6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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 9 3 6
Erreur # Valeur incompatible #
Valeurs possibles : 1 2 3 4 5
Appuyez sur une touche pour continuer ...
```

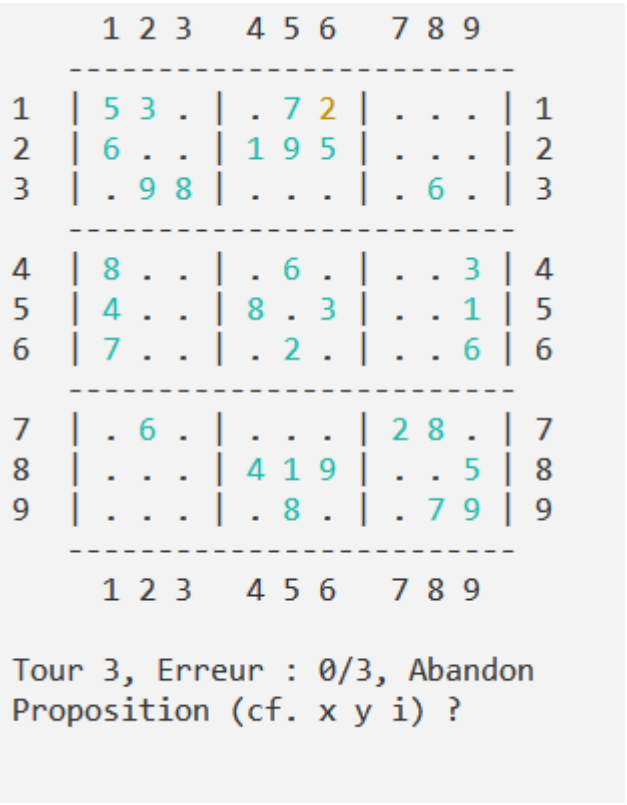
```
    1 2 3   4 5 6   7 8 9
  -------------------------
1 | 5 3 . | . 7 . | . . . | 1
2 | 6 . . | 1 9 5 | . . . | 2
3 | 1 9 8 | . . . | . 6 . | 3
  -------------------------
4 | 8 . . | . 6 . | . . 3 | 4
5 | 4 . . | 8 . 3 | . . 1 | 5
6 | 7 . . | . 2 . | . . 6 | 6
  -------------------------
7 | . 6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

Tour 3, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 8



Jeux d’essais 9





Jeux d'essais 10

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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 1 1 6
Erreur # Saisie sur grille de depart #

Appuyez sur une touche pour continuer ...
```

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

Tour 3, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 11

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 3 3 1
Erreur # Saisie sur grille de depart #

Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 12

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

Tour 2, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 9 5 7
Erreur # Saisie sur grille de depart #

Appuyez sur une touche pour continuer ...
```

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

Tour 3, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 13

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

Tour 3, Erreur : 2/4, Abandon
Proposition (cf. x y i) ? l e a
E R R E U R  D E  S A I S I E  ! ! !

Appuyez sur une touche pour continuer ...
```

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

Tour 4, Erreur : 3/4, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 14

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? a 2 3
E R R E U R  D E  S A I S I E  ! ! !

Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 15

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? l 5 c
E R R E U R  D E  S A I S I E  ! ! !

Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 16

Jeux d'essais 17

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

Tour 4, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 0 0 0
A B A N D O N ! !

Appuyez sur une touche pour continuer ...
```

```
                    ------------------------------
                    ||            SUDOKU          ||
                    ------------------------------


Le Sudoku est une grille (ici de 9x9) que vous devez completer avec des valeurs, de sorte que :
   -    Chaque ligne,
   -    Chaque colonne,
   -    Et chaque zone (3x3) de la grille
   Contienne les valeurs 1, 2, 3, 4, 5, 6, 7, 8 et 9.
Une grille incomplete vous est proposee en debut de partie. A vous de la completer en proposant une valeur
 a placer sur la grille.
Vous gagnez lorsque vous avez completer entierement la grille, vous perdez lorsque vous avez epuise votre
nombre de droits a l erreur defini par vous-meme.

Vous avez abandonne.
Fin de la partie
```

Jeux d'essais 18

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

Tour 7, Erreur : 2/3, Abandon
Proposition (cf. x y i) ? 3 8 7
Erreur # Saisie sur grille de depart #
P E R D U  ! ! ! plus de 3 erreurs

Appuyez sur une touche pour continuer ...
```

```
                    ------------------------------
                    ||            SUDOKU          ||
                    ------------------------------


Le Sudoku est une grille (ici de 9x9) que vous devez completer avec des valeurs, de sorte que :
   -    Chaque ligne,
   -    Chaque colonne,
   -    Et chaque zone (3x3) de la grille
   Contienne les valeurs 1, 2, 3, 4, 5, 6, 7, 8 et 9.
Une grille incomplete vous est proposee en debut de partie. A vous de la completer en proposant une valeur
 a placer sur la grille.
Vous gagnez lorsque vous avez completer entierement la grille, vous perdez lorsque vous avez epuise votre
nombre de droits a l erreur defini par vous-meme.

Vous avez fait 3 erreurs, vous avez perdu.
Fin de la partie
```

Jeux d'essais 19

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

Tour 4, Erreur : 3/4, Abandon
Proposition (cf. x y i) ? l e a
E R R E U R  D E  S A I S I E  ! ! !
P E R D U  ! ! ! plus de 4 erreurs

Appuyez sur une touche pour continuer ...
```

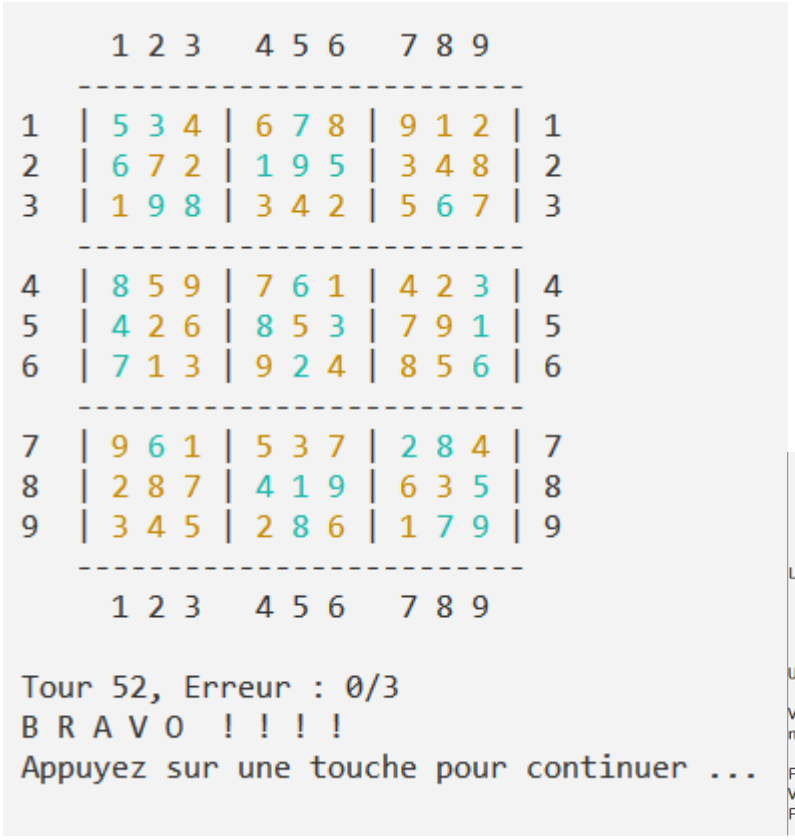
```
                    SUDOKU

Le Sudoku est une grille (ici de 9x9) que vous devez completer avec des valeurs, de sorte que :
   -    Chaque ligne,
   -    Chaque colonne,
   -    Et chaque zone (3x3) de la grille
   Contienne les valeurs 1, 2, 3, 4, 5, 6, 7, 8 et 9.
Une grille incomplete vous est proposee en debut de partie. A vous de la completer en proposant une valeur
 a placer sur la grille.
Vous gagnez lorsque vous avez completer entierement la grille, vous perdez lorsque vous avez epuise votre
nombre de droits a l erreur defini par vous-meme.

Vous avez fait 4 erreurs, vous avez perdu.
Fin de la partie
```

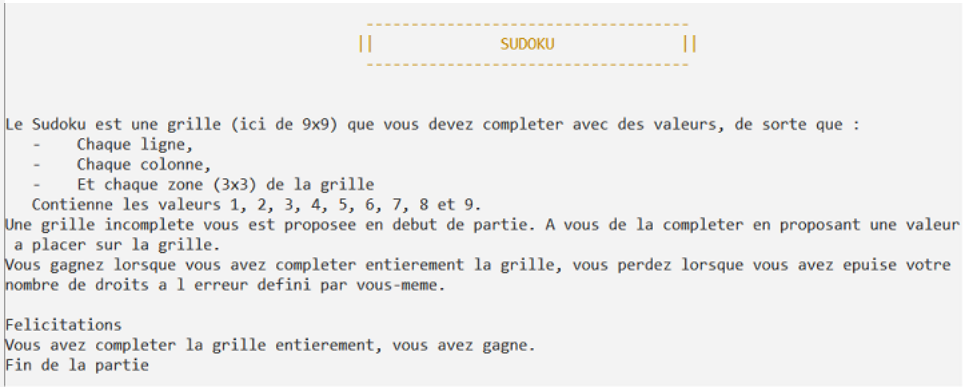
Jeux d'essais 20

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

 Tour 18, Erreur : 17/18, Abandon
 Proposition (cf. x y i) ? 1 1 6
 Erreur # Saisie sur grille de depart #
P E R D U  ! ! ! plus de 18 erreurs

 Appuyez sur une touche pour continuer ...
```

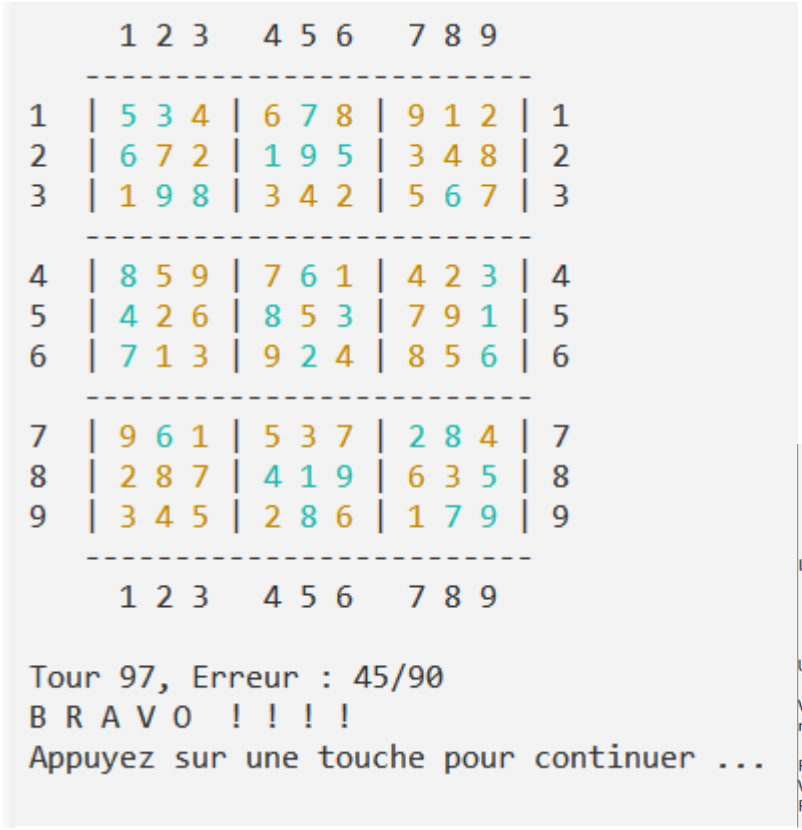
```
                    ----------------------------
                    ||          SUDOKU          ||
                    ----------------------------


Le Sudoku est une grille (ici de 9x9) que vous devez completer avec des valeurs, de sorte que :
   -    Chaque ligne,
   -    Chaque colonne,
   -    Et chaque zone (3x3) de la grille
   Contienne les valeurs 1, 2, 3, 4, 5, 6, 7, 8 et 9.
Une grille incomplete vous est proposee en debut de partie. A vous de la completer en proposant une valeur
 a placer sur la grille.
Vous gagnez lorsque vous avez completer entierement la grille, vous perdez lorsque vous avez epuise votre
nombre de droits a l erreur defini par vous-meme.

Vous avez fait 18 erreurs, vous avez perdu.
Fin de la partie
```

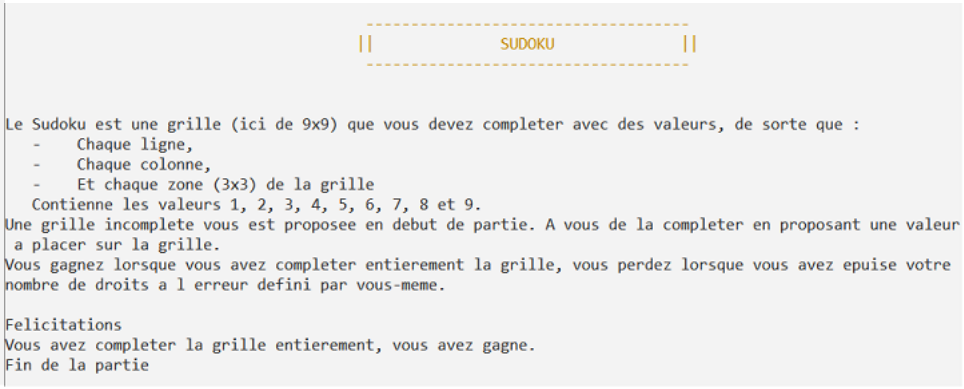
Jeux d'essais 21





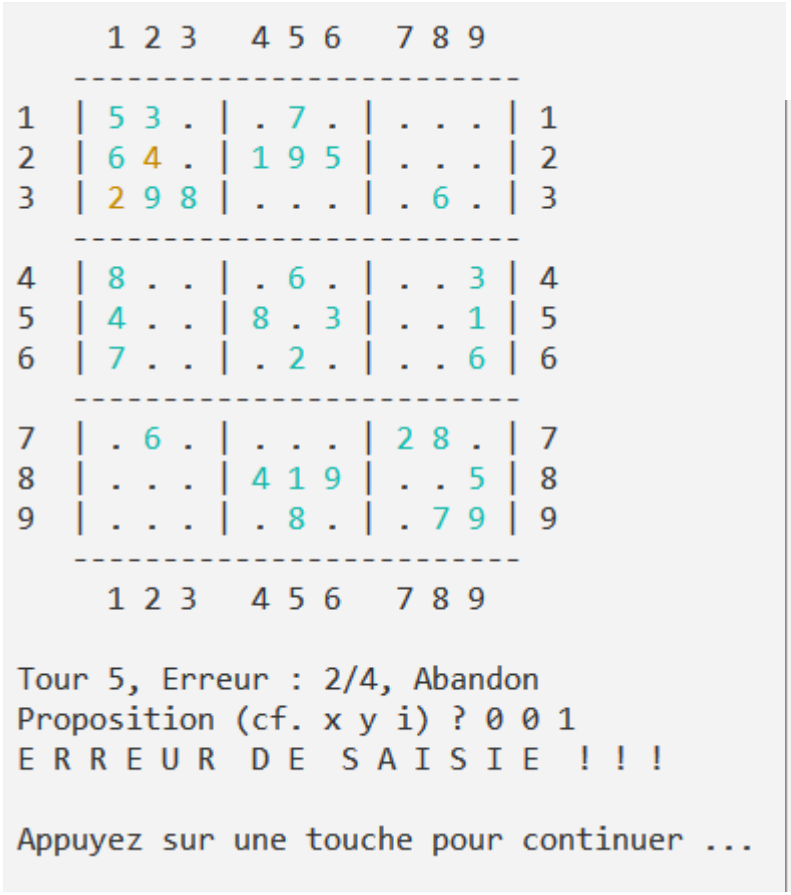
Jeux d'essais 22

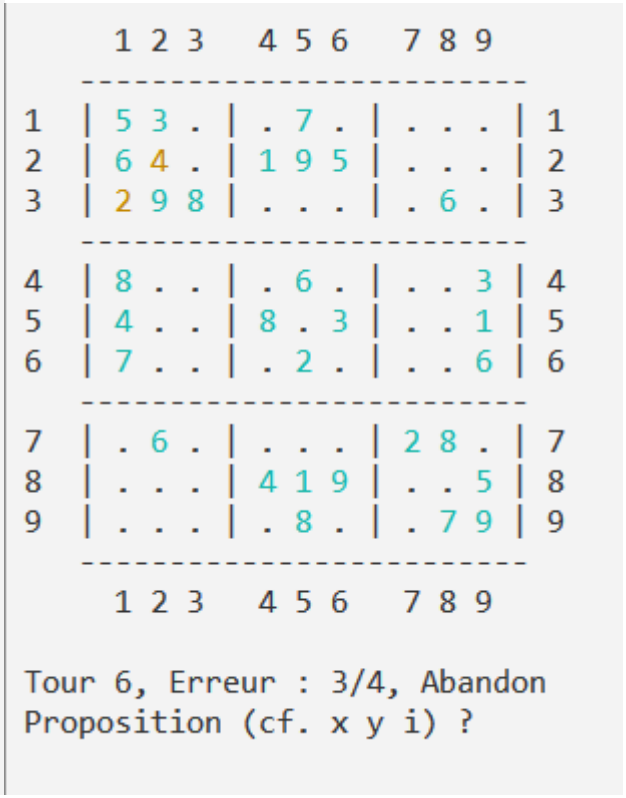




Situations complémentaires

Jeux d'essais 1





Jeux d'essais 2

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ? 0 1 1
E R R E U R  D E  S A I S I E  ! ! !

Appuyez sur une touche pour continuer ...
```

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

Tour 2, Erreur : 1/3, Abandon
Proposition (cf. x y i) ?
```

Jeux d’essais 3

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

Tour 1, Erreur : 0/54, Abandon
Proposition (cf. x y i) ?
```

```
Choix du nombre d'erreurs autorisees (>= 3) : 54
                                    Vous pouvez faire jusqu'a 54 erreurs

Appuyer sur une touche pour continuer ...
```

Jeux d’essais 4

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

```
Choix du nombre d'erreurs autorisees (>= 3) : -8
                                    Vous pouvez faire jusqu'a 3 erreurs

Appuyer sur une touche pour continuer ...
```

Jeux d’essais 5

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

```
Choix du nombre d'erreurs autorisees (>= 3) : 14,5
                                    Vous pouvez faire jusqu'a 3 erreurs

Appuyer sur une touche pour continuer ...
```

Jeux d’essais 6

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

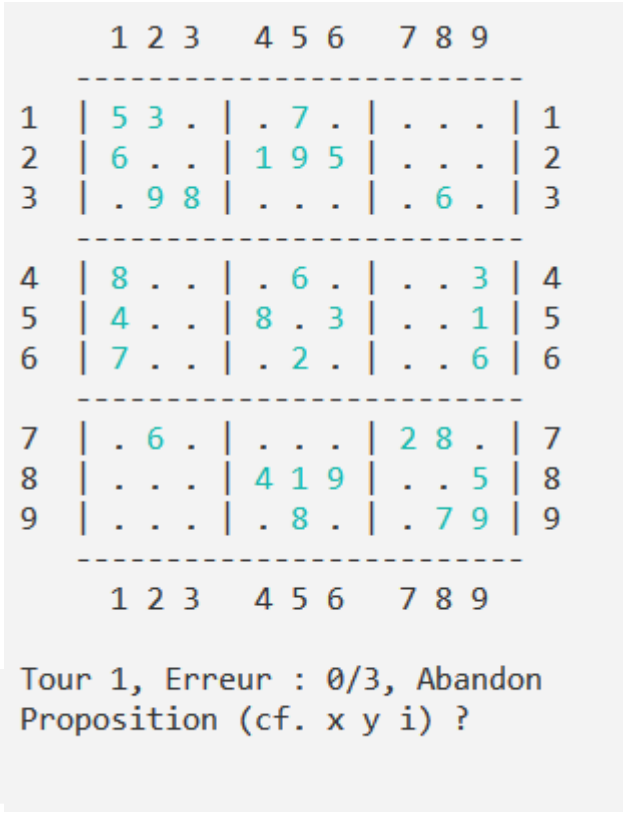
```
Choix du nombre d'erreurs autorisees (>= 3) : 14.5
                                    Vous pouvez faire jusqu'a 3 erreurs

Appuyer sur une touche pour continuer ...
```

Jeux d’essais 7

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

Tour 1, Erreur : 0/3, Abandon
Proposition (cf. x y i) ?
```

```
Choix du nombre d'erreurs autorisees (>= 3) : a
                                    Vous pouvez faire jusqu'a 3 erreurs

Appuyer sur une touche pour continuer ...
```

Jeux d’essais 8




```
Choix du nombre d'erreurs autorisees (>= 3) : abc
                                    Vous pouvez faire jusqu'a 3 erreurs

Appuyer sur une touche pour continuer ...
```

Annexe 2 : Dictionnaires des données

Par manque de temps, Nous n'avons pas pu finaliser toutes les descriptions des algorithmes et mettre également tous les dictionnaires des données correspondants

NOM	TYPE	SIGNIFICATION
nbErreur	Entier court non signé	Le nombre d'erreurs du joueur au cours du jeu.
nbTour	Entier court non signé	Le nombre de tours accomplis par le joueur.
stadeActuelPartie	stadePartie (type énuméré)	Le stade actuel auquel appartient le joueur durant une partie. Il peut prendre les valeurs suivantes : enCours, choixAbandon, erreurMax ou victoire.
grilleJeu [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions dans lequel va se dérouler le jeu.
grilleBase [NB_CASES][NB_CASES]	Tableau de caractères	Le tableau à 2 dimensions de début de jeu, contenant les chiffres de départ.
charNbErreurMax	Caractère	Le nombre d'erreurs maximum récupéré après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
nbErreurMax	Entier	Le nombre d'erreur maximum que le joueur s'autorise.
NBR_ERREUR_MIN	Constant entier court non signé	Le nombre d'erreurs minimum autorisé que le joueur peut saisir.
NB_TOUR_MIN_FIN	Constant entier court non signé	Le nombre de tours minimum pour pouvoir compléter la grille. Il a été calculé au préalable pour optimiser le jeu et éviter de faire des vérifications inutiles : Il y a 81 cases et 30 chiffres pré-rentrées donc le joueur peut finir le jeu en 51 coups minimum soit 51 tours.
charAbscisse	Caractère	L'abscisse de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charOrdonnee	Caractère	L'ordonnée de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
charElement	Caractère	L'élément de la proposition récupérée après sa saisie. Il sera ensuite converti en entier, mais il est utilisé seulement pour faire des vérifications pour les erreurs de saisie (sur les caractères)
propJoueur	Enregistrement	Proposition du joueur qui consiste à récupérer la saisie d'une abscisse, d'une ordonnée et d'un élément.
tabValIncomp[NB_CASES]	Tableau de caractères	Le tableau dans lequel on va ajouter (sans doublon) toutes les valeurs trouvées sur ligne, colonne et carré pour des coordonnées [x, y] rentrées. Ce tableau est utilisé pour afficher les valeurs possibles lorsque l'utilisateur se trompe (scénario 2)
tabValComp[NB_CASES]	Tableau de caractères	Tableau de 1 à 9 utilisé pour comparer les valeurs lors de l'affichage des valeurs possibles (scénario 2)

tabBoolVerif[NB_CASES]	Tableau de booléen	Le tableau dans lequel les valeurs incompatibles présentent dans le tableau tabValIncomp seront égale à faux, vrai sinon. Utiliser pour trouver les valeurs possible (scénario 2)
COTE_ZONE	Constant entier court non signé	Le côté du carre/de la zone qui est de 3. Utilisé pour la vérification sur la zone.
etatPropActuel	etatProp	L'état de la proposition actuel. Détermine s'il y a une erreur de saisie, une condition d'abandon, une erreur sur la grille de départ, la valeur est incompatible avec la grille (l'élément ne peut pas être placé), la valeur est compatible avec la grille (l'élément peut être placé) et en traitement (durant un tour, l'état de la proposition est en cours de traitement). Il peut donc prendre les valeurs suivantes : erreurSaisie, erreurGrilleDep, abandon, enTrait, valIncomp ou valComp.
zoneAbscisse	Entier court non signé	L'abscisse de la zone qui va être parcouru
zoneOrdonnee	Entier court non signé	L'ordonnée de la zone qui va être parcouru
		Correspond à la saisie son forme d'un chaine de caractère qui servira par la suite à faire les vérification de validation du nombre d'erreurs maximum