Pendant ce projet nous allons voir comment résoudre des problèmes à priori difficiles par essai/erreur en utilisant des exceptions.

1 Modélisation du problème

J'imagine que tout le monde connaît les règles du sudoku, voici tout de même un petit rappel du jeu. Le sudoku est un jeu sur une grille de 9x9 cases, découpée en 9 sous carrés de 3x3 cases. Certaines cases contiennent déjà un entier de 1 à 9 le but est de remplir toutes les cases avec de tels entiers en vérifiant les contraintes que chaque entier doit apparaître une unique fois sur chaque ligne, sur chaque colonne et dans chaque sous-carré.

Nous allons ici fabriquer un programme qui résout toutes les grilles de sudoku, pour cela on va modéliser chaque case par une liste des valeurs possibles de cette case, on a donc le type `case = int list`. Une grille est alors définie comme une liste de liste de cases :
`type grille = case list list`.

Question 1 [4] Écrire une fonction `sure: case -> int option` qui renvoie `Some a` si la case ne contient qu'un entier a on est alors sûr de la valeur de cette case et `None` dans les autres cas.

Question 2 Cherché une grille de sudoku facile sur internet (ou ailleurs). Écrire une valeur `grille_test : grille` qui encode cette grille de sudoku.

Question 3 [8] On va tout d'abord écrire trois fonctions qui vont nous permettre de manipuler plus facilement les lignes colonnes et sous carrés. Les trois fonctions ont le même type :
`int*int -> int -> int * int` les deux premiers arguments sont un couple (i, j) désignant une case de la grille le troisième argument est un entier entre 1 et 9.
La fonction `iter_colonne (i,j)` est une bijection de $[1; 9]$ vers les cases de la colonne contenant la case (i, j).
La fonction `iter_ligne (i,j)` est une bijection de $[1; 9]$ vers les cases de la ligne contenant la case (i, j).
La fonction `iter_carre (i,j)` est une bijection de $[1; 9]$ vers les cases du sous carré contenant la case (i, j).

Question 4 [14] Écrire une fonction qui imprime une grille à l'écran. *Vous pourrez utiliser les fonctions* `print_string: string -> unit` ; `print_newline : unit -> unit` ; `print_int: int -> unit` ;

2 Résolution de contrainte simple

Question 5 Écrire la fonction : `map_grille : (int -> int -> case -> case) -> grille -> grille` tel que `map_grille f g` applique la fonction `f` à toutes les cases de la grille `g`. Les 2 premiers paramètres de `f` sont les coordonnées en colonne et ligne de la case numérotées de 0 à 8. Cette fonction nécessite des fonctions auxiliaires récursives.

Au début de la résolution, chaque case d'une grille contient soit une seule valeur, soit toutes les valeurs (c.-à-d. la liste `[1;2;3;4;5;6;7;8;9]`). Dans cette partie nous allons supprimer au fur et à

mesure des éléments dans les listes en applicant directement les règles du sudoku. Les chiffres entre crochets donnent une idée du nombre de lignes nécessaire pour écrire la fonction.

Pour simplifier les tests, je vous fournis le code suivant, la valeur `grille_test` est valide pour commencer la résolution.

```
let build_grille = fun g ->
  map_grille (fun _ -> fun _ -> fun c ->
     match c with
           [] -> [1;2;3;4;5;6;7;8;9]
         | [x] -> [x]
         | _ -> failwith "Grille incorrect"
    );;
let grille_test_incomplete [
  [[1]; [ ]; [ ];   [8]; [3]; [ ];    [ ]; [ ]; [2]];
  [[5]; [7]; [ ];   [ ]; [ ]; [1];    [ ]; [ ]; [ ]];
  [[ ]; [ ]; [ ];   [5]; [ ]; [9];    [ ]; [6]; [4]];

  [[7]; [ ]; [4];   [ ]; [ ]; [8];    [5]; [9]; [ ]];
  [[ ]; [ ]; [3];   [ ]; [1]; [ ];    [4]; [ ]; [ ]];
  [[ ]; [5]; [1];   [4]; [ ]; [ ];    [3]; [ ]; [6]];

  [[3]; [6]; [ ];   [7]; [ ];  [4];   [ ]; [ ]; [ ]];
  [[ ]; [ ]; [ ];   [6]; [ ];  [ ];   [ ]; [7]; [9]];
  [[8]; [ ]; [ ];   [ ]; [5];  [2];   [ ]; [ ]; [3]]]
];;

let grille_test = build_grille grille_test_incomplete;;
```

Dans la suite le but est d'enlever le plus de valeur possible dans chaque case jusqu'à ce que la valeur devienne sure. La plupart des fonctions ont un argument `acc` de type `case` qui contient la liste des valeurs possible pour une case. Pour chacune des cases, il faut : regarder pour toutes les cases qui sont dans la même colonne, ligne et carré si la valeur est sure l'enlever des possibilités. Les questions suivantes découpent ce problème, chaque question est assez courte. pour simplifier la lecture des types, je définis les abréviations de types suivantes :

```
type coord = int*int
type iter = int -> int -> int -> coord
```

Les fonctions `iter_column, iter_ligne, iter_carre` sont de type `iter`.

Question 6 [2] Écrire la fonction `get : grille -> int -> int -> case` tel que `get g i j` renvoit le contenu de la case à la ligne i et à la colonne j de la grille `g`. *On pourra utiliser la fonction List.nth*

Question 7 [4] Écrire la fonction `retire_valeur_case : grille -> coord -> case -> case` telle que `retire_case grille acc (l,m)` regarde si la valeur de la case au coordonné `(l,m)` est sure. Si elle l'est, la fonction renvoie la liste des possibilités `acc` sans cette valeur. Sinon `acc` est renvoyé inchangé. *On pourra utiliser la fonction List.filter*

Question 8 [3] Écrire la fonction `retire_valeur_case_it : grille -> coord -> case -> iter -> int -> case` tel que `retire_valeur_case_it g (i,j) acc it k` calcule les coordonnées `(l,m)` avec l'itérateur `it` appliqué aux coordonnées `(i,j)` et l'entier `k`. Si $(i,j) \neq (k,l)$ la fonction applique `retire_valeur_case` sinon elle renvoie `acc`.

Question 9 [4] Écrire la fonction `retire_valeur_case_it_app : grille -> coord -> case -> it -> case` qui applique la fonction précédente pour toutes les valeurs dans $[0;1;2;3;4;5;6;7;8]$. *Vous pouvez utiliser la fonction* `List.fold_left` *ou utiliser une fonction auxiliaire récursive.*

Question 10 [5] Écrire la fonction `iter_all: grille -> coord -> case -> case` qui applique la fonction précédente sur les trois itérateurs.

Question 11 [4] Écrire la fonction `enleve_sur : grille -> grille` qui applique la fonction précédente sur chaque case de la grille.

Faites des tests sur la grille `grille_test` à chaque application de `enleve_sur` le nombre de possibilités diminue, au bout de 5 applications la grille doit être résolue. Faites des tests avec d'autres grilles.

```
grille_test :
+-------------------------------+-------------------------------+-------------------------------+
|1          123456789 123456789 |8          3          123456789 |123456789 123456789 2          |
|5          7         123456789 |123456789 123456789 1          |123456789 123456789 123456789 |
|123456789 123456789 123456789 |5          123456789 9          |123456789 6         4          |
+-------------------------------+-------------------------------+-------------------------------+
|7          123456789 4         |123456789 123456789 8          |5         9         123456789 |
|123456789 123456789 3          |123456789 1         123456789 |4         123456789 123456789 |
|123456789 5         1          |4         123456789 123456789 |3         123456789 6         |
+-------------------------------+-------------------------------+-------------------------------+
|3          6         123456789 |7          123456789 4         |123456789 123456789 123456789 |
|123456789 123456789 123456789 |6          123456789 123456789 |123456789 7         9          |
|8          123456789 123456789 |123456789 5         2          |123456789 123456789 3          |
+-------------------------------+-------------------------------+-------------------------------+
enleve_sure grille_test :
+-------------------------+-------------------------+-------------------------+
|1          49         69 |8          3          67 |79         5          2  |
|5          7          2689 |2          246        1  |89         38         8  |
|2          238        28 |5          27         9  |178        6          4  |
+-------------------------+-------------------------+-------------------------+
|7          2          4  |23         26         8  |5          9          1  |
|269        289        3  |29         1          567 |4          28         78 |
|29         5          1  |4          279        7  |3          28         6  |
+-------------------------+-------------------------+-------------------------+
|3          6          259 |7          89         4  |128        1258       158 |
|24         124        25 |6          8          3  |128        7          9  |
|8          149        79 |19         5          2  |16         14         3  |
+-------------------------+-------------------------+-------------------------+
enleve_sure (enleve_sure grille_test) :
+-------------------------+-------------------------+-------------------------+
|1          49         69 |8          3          6  |79         5          2  |
|5          7          69 |2          46         1  |9          3          8  |
|2          38         8  |5          7          9  |17         6          4  |
+-------------------------+-------------------------+-------------------------+
|7          2          4  |3          6          8  |5          9          1  |
|69         89         3  |9          1          56 |4          28         7  |
|9          5          1  |4          29         7  |3          28         6  |
+-------------------------+-------------------------+-------------------------+
|3          6          259 |7          9          4  |128        128        5  |
|4          14         25 |6          8          3  |12         7          9  |
|8          149        79 |19         5          2  |16         14         3  |
+-------------------------+-------------------------+-------------------------+
enleve_sure (enleve_sure (enleve_sure grille_test)) :
+-------------------------+-------------------------+-------------------------+
|1          49         9  |8          3          6  |7          5          2  |
|5          7          6  |2          4          1  |9          3          8  |
|2          3          8  |5          7          9  |1          6          4  |
+-------------------------+-------------------------+-------------------------+
|7          2          4  |3          6          8  |5          9          1  |
|6          8          3  |9          1          5  |4          28         7  |
|9          5          1  |4          2          7  |3          28         6  |
+-------------------------+-------------------------+-------------------------+
|3          6          2  |7          9          4  |128        128        5  |
|4          1          25 |6          8          3  |12         7          9  |
|8          19         79 |1          5          2  |16         14         3  |
+-------------------------+-------------------------+-------------------------+
enleve_sure (enleve_sure (enleve_sure (enleve_sure grille_test))) :
+-------------------------+-------------------------+-------------------------+
|1          4          9  |8          3          6  |7          5          2  |
|5          7          6  |2          4          1  |9          3          8  |
|2          3          8  |5          7          9  |1          6          4  |
+-------------------------+-------------------------+-------------------------+
|7          2          4  |3          6          8  |5          9          1  |
|6          8          3  |9          1          5  |4          2          7  |
|9          5          1  |4          2          7  |3          8          6  |
+-------------------------+-------------------------+-------------------------+
|3          6          2  |7          9          4  |8          18         5  |
|4          1          5  |6          8          3  |2          7          9  |
|8          9          7  |1          5          2  |6          4          3  |
+-------------------------+-------------------------+-------------------------+
enleve_sure (enleve_sure (enleve_sure (enleve_sure (enleve_sure grille_test)))) :
+-------------------------+-------------------------+-------------------------+
|1          4          9  |8          3          6  |7          5          2  |
|5          7          6  |2          4          1  |9          3          8  |
|2          3          8  |5          7          9  |1          6          4  |
+-------------------------+-------------------------+-------------------------+
|7          2          4  |3          6          8  |5          9          1  |
|6          8          3  |9          1          5  |4          2          7  |
|9          5          1  |4          2          7  |3          8          6  |
+-------------------------+-------------------------+-------------------------+
```

```
|3        6        2        |7        9        4        |8        18       5        |
|4        1        5        |6        8        3        |2        7        9        |
|8        9        7        |1        5        2        |6        4        3        |
+--------------------------+--------------------------+--------------------------+
```