Marcel Délèze

SUDOKU, ensemble des solutions

Entrée des données

Grille vierge (à recopier sur la grille "donnee" pour entrer une nouvelle grille):

```
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □;
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □
□ □ □ □ □ □ □ □ □
```

Grille de données (à remplir):

```
          □ □ □ 9 □ □ □ □ 6
          □ □ □ 7 □ □ □ □ 1
          □ □ 5 □ 2 8 3 □ □
          □ 9 8 1 □ □ □ □ □
donnee = □ 7 □ □ □ □ □ 3 □;
          □ □ □ □ □ 4 9 8 □
          □ □ 3 5 4 □ 6 □ □
          1 □ □ □ □ 3 □ □ □
          2 □ □ □ □ 9 □ □ □
```

Représentation interne des grilles:

```
donnee = Apply[sdk, donnee /. {□ → Null}]
```

```
sdk[{Null, Null, Null, 9, Null, Null, Null, Null, 6}, {Null, Null, Null, 7, Null, Null, Null, Null, 1}, {Null, Null, 5, Null, 2, 8, 3, Null, Null},
 {Null, 9, 8, 1, Null, Null, Null, Null, Null}, {Null, 7, Null, Null, Null, Null, Null, 3, Null}, {Null, Null, Null, Null, Null, 4, 9, 8, Null},
 {Null, Null, 3, 5, 4, Null, 6, Null, Null}, {1, Null, Null, Null, Null, 3, Null, Null, Null}, {2, Null, Null, Null, Null, 9, Null, Null, Null}]
```

Affichage d'une grille:

```
affiche[grille_sdk] := FrameBox[GridBox[Apply[List, grille] /. {Null → "·"}, RowLines → {False, False, True, False, False, True, False, False},
    ColumnLines → {False, False, True, False, False, True, False, False}]] // DisplayForm

affiche[g_List] := Map[affiche, g]

affiche[donnee]
```

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

Résolution

Détermination des chiffres possibles

Extractions de lignes, de colonnes et de carrés:

```
ligne[grille_sdk, i_] := grille[[i]]

colonne[grille_sdk, j_] := Transpose[Apply[List, grille]][[j]]

carre[grille_sdk, i_, j_] :=
 Module[{ib, jb, ic, jc}, ib = 3 Quotient[i - 1, 3] + 1; jb = 3 Quotient[j - 1, 3] + 1; Flatten[Table[grille[[ib + ic, jb + jc]], {ic, 0, 2}, {jc, 0, 2}], 1]]
```

Pour une case vide, les chiffres possibles (qu'on pourrait essayer) sont ceux qui ne se trouvent ni dans la ligne, ni dans la colonne, ni dans le carré:

```
case[grille_sdk, i_, j_] := grille[[i, j]] /; IntegerQ[grille[[i, j]]]

case[grille_sdk, i_, j_] := Complement[Range[1, 9], ligne[grille, i], colonne[grille, j], carre[grille, i, j]]

possible[grille_sdk] := Apply[sdk, Table[case[grille, i, j], {i, 1, 9}, {j, 1, 9}]];
```

```
affiche[possible[donnee]]
```

{3, 4, 7, 8} {3, 4, 6, 8, 9} {4, 6, 7, 9}	{1, 2, 3, 4, 8} {2, 3, 4, 6, 8} {1, 4, 6}	{1, 2, 4, 7} {2, 4, 6, 9} 5	9 7 {4, 6}	{1, 3, 5} {3, 5, 6} 2	{1, 5} {5, 6} 8	{2, 4, 5, 7, 8} {2, 4, 5, 8} 3	{2, 4, 5, 7} {2, 4, 5, 9} {4, 7, 9}	6 1 {4, 7, 9}
{3, 4, 5, 6} {4, 5, 6} {3, 5, 6}	9 7 {1, 2, 3, 5, 6}	8 {1, 2, 4, 6} {1, 2, 6}	1 {2, 6, 8} {2, 3, 6}	{3, 5, 6, 7} {5, 6, 8, 9} {3, 5, 6, 7}	{2, 5, 6, 7} {2, 5, 6} 4	{2, 4, 5, 7} {1, 2, 4, 5} 9	{2, 4, 5, 6, 7} 3 8	{2, 4, 5, 7} {2, 4, 5} {2, 5, 7}
{7, 8, 9} 1 2	{8} {4, 5, 6, 8} {4, 5, 6, 8}	3 {4, 6, 7, 9} {4, 6, 7}	5 {2, 6, 8} {6, 8}	4 {6, 7, 8} {1, 6, 7, 8}	{1, 2, 7} 3 9	6 {2, 4, 5, 7, 8} {1, 4, 5, 7, 8}	{1, 2, 7, 9} {2, 4, 5, 7, 9} {1, 4, 5, 7}	{2, 7, 8, 9} {2, 4, 5, 7, 8, 9} {3, 4, 5, 7, 8}

Détermination des grilles qui comportent un chiffre de plus

Tester les impasses.

Former la grille suivante, ou plusieurs grilles lorsqu'il y a plusieurs possibilités, comme suit:

* choisir une case pour laquelle la liste des chiffres à essayer est de longueur minimale;
* pour chaque chiffre à essayer, former une grille comportant ce chiffre

```
suivant[grille_sdk] := Module[{e1, li, lj, nPoss, i, j, k}, e1 = possible[grille];
  If[MemberQ[e1, {}, {2}], {},
   nPoss = 10;
   Do[If[1 ≤ Length[e1[[i, j]]] < nPoss,
     li = i; lj = j; nPoss = Length[e1[[i, j]]]],
    {i, 1, 9}, {j, 1, 9}];
   Table[ReplacePart[grille, e1[[li, lj]][[k]], {li, lj}], {k, 1, nPoss}]]]

suivant[g_List] := Flatten[Map[suivant, g]]
```

Itération

```
resous[grille_] := Nest[suivant, donnee, 81 - Count[donnee, _Integer, {2}]]
```

```
affiche[resous[donnee]]
```

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