

Exemple 3

Sudoku is a pleasant pastime (for yachts, trains, planes, etc.) that allows you to stay in great intellectual shape.

Le SuDoku est un agréable passe temps (pour les bateaux, les trains, les avions, etc.) qui permet de se maintenir en grande forme intellectuelle.

We deal, with grids made up of 9 rows, 9 columns and 9 blocks, therefore having 81 cells.

Nous traitons, les grilles formées de 9 lignes, 9 colonnes et 9 blocs, donc ayant 81 cellules.

We assume that you have software for making a grid, or that you have a data grid at your disposal.

Nous supposons que vous disposez d'un logiciel de fabrication d'une grille, ou que vous avez à votre disposition une grille de données.

Sudoku rule: You must complete the given grid using the 9 numbers from 1 to 9 so that in each row, in each column and in each block these 9 numbers appear only once.

Règle du Sudoku : Il faut compléter la grille donnée en utilisant les 9 chiffres de 1 à 9 de telle manière que dans chaque ligne, dans chaque colonne et dans chaque bloc ces 9 nombres n'apparaissent qu'une et une seule fois.

We consider that the search for the solution is a fight.

Nous considérons que la recherche de la solution est un combat.

This makes it possible to determine winning numbers, thus simplifying the rest.

Cela permet de déterminer des nombres gagnants, donc simplifier la suite.

We use two text file for presentation.

Nous utilisons deux fichier texte pour la présentation

sudoku_grid_9x9.txt to draw the grid

```
\linethickness=1.2pt
\putrectangle corners at 0 9 and 9 0
\putrule from 0 3 to 9 3 \putrule from 0 6 to 9 6
\putrule from 3 0 to 3 9 \putrule from 6 0 to 6 9
\linethickness=0.2pt
\plot 0 3 9 3 / \plot 0 6 9 6 \plot 3 0 3 9 / \plot 6 0 6 9 /
\setdashes<2pt>
\plot 0 1 9 1 \plot 0 2 9 2 \plot 0 4 9 4 \plot 0 5 9 5 \plot 0 7 9 7 \plot 0 8 9 8 /
\plot 1 0 1 9 \plot 2 0 2 9 \plot 4 0 4 9 \plot 5 0 5 9 \plot 7 0 7 9 \plot 8 0 8 9 /
\setdashes<.1pt>
```

sudoku_mark_9x9.txt for numbering rows and columns

```
$$\beginpicture\xyunits<30pt,30pt> \measures(0,9)(0,9)
\put{$L_1$} at -0.5 8.5 \put{$L_2$} at -0.5 7.5 \put{$L_3$} at -0.5 6.5
\put{$L_4$} at -0.5 5.5 \put{$L_5$} at -0.5 4.5 \put{$L_6$} at -0.5 3.5
\put{$L_7$} at -0.5 2.5 \put{$L_8$} at -0.5 1.5 \put{$L_9$} at -0.5 0.5
\put{$C_1$} at 0.5 9.5 \put{$C_2$} at 1.5 9.5 \put{$C_3$} at 2.5 9.5
\put{$C_4$} at 3.5 9.5 \put{$C_5$} at 4.5 9.5 \put{$C_6$} at 5.5 9.5
\put{$C_7$} at 6.5 9.5 \put{$C_8$} at 7.5 9.5 \put{$C_9$} at 8.5 9.5
\endpicture$$
```

Example :

We need to solve the following sukou
su_ex_10.txt

Nous devons résoudre le sukou suivant

For this we choose the following method:
we write possible numbers in each box

Pour cela nous choisissons la méthode suivante :
nous écrivons dans chaque case les nombres possibles

for example in box L_1C_2 , we have to eliminate the numbers

par exemple dans la case L_1C_2 , nous devons éliminer les nombres

4, 9, 2, 7, 1, 5, 8 and 1, 8, 4, 9 \Rightarrow possible 3, 6

and the same for all other empty boxes.

et de même pour toutes les autres cases vide.

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

Figure 1: example

The **Python** program to get the data grid (called combat grid) with possible numbers is **sudoku_9x9.py**

Le programme **Python** pour obtenir la grille de données (dite grille de combat) avec les nombres possibles est

```
# write file *.txt for TeX
from math import *
# data grid, hand write the 9 rows below (modified for new data)
L1 = [4,0,9,7,0,1,5,0,8]
L2 = [0,0,2,0,3,0,0,0,0]
L3 = [7,0,0,0,0,0,3,1,0]
L4 = [0,0,0,0,2,0,6,0,3]
L5 = [9,1,0,4,0,0,8,0,0]
L6 = [0,8,3,0,9,5,0,0,0]
L7 = [0,0,0,3,0,0,9,4,0]
L8 = [0,4,0,0,1,6,0,0,0]
L9 = [3,9,0,0,4,0,7,0,0]
grid = [L1,L2,L3,L4,L5,L6,L7,L8,L9]
n = len(grid)
print(' ***** Data grid ***** ')
print(' L1 = ',L1,'\n', ' L2 = ',L2,'\n', ' L3 = ',L3,'\n', ' L4 = ',L4,'\n', ' L5 = ',L5,'\n', ' L6 = ',L6,'\n',
      ' L7 = ',L7,'\n', ' L8 = ',L8,'\n', ' L9 = ',L9,'\n')
print(' ***** ')
name_file = input(' Name of data fich, without .txt : ')
my_file = open(name_file+".txt", "w")
# text for compilation PiCTeX
my_file.write(chr(36)+chr(36)+chr(92)+'beginpicture'+chr(92)+'\n'
              'xyunits<30pt,30pt>'+chr(92)+'measures(0,9)(0,9) \n') % adapt xyunits
my_file.write(chr(92)+'input sudoku_grid_9x9.txt'+'\n')
my_file.write(chr(92)+'input sudoku_mark_9x9.txt'+'\n')
# write grid
position = [[0 for i in range(n*n*n)] for j in range(n*n*n)]
count = [0 for i in range(n*n)]
# value grid position of data sudoku
# i row and j column
for i in range(n):
    for j in range(n):
        if grid[i][j]>0 :
            x, y = j+.5, 9-i-.5 ; # (x, y) coordonnate center
            a = grid[i][j]
            position[a][count[a]] = x
            position[a][count[a]+1] = y
```

```

        count[a] = count[a]+2
# write data grid sudoku
for i in range(1,n+1):
    my_file.write(chr(92)+'multiput{' +chr(36)+chr(92)+'mathbf ' + str(i)+chr(36)+'} at ')
    for j in range(count[i]):
        my_file.write(' '+str(position[i][j]))+' ')
    my_file.write(chr(47)+'\n')
# *****
# write candidate number
writeposition = [[0 for i in range(n*n*n)] for j in range(n*n*n)]
count = [0 for i in range(1,n*n*n)]
coordinate = [[-.3, .3], [0, .3], [.3, .3], [-.3, 0], [0, 0], [.3, 0], [-.3, -.3], [0, -.3], [.3, -.3]]
for r in range(n):
    for c in range(n):
        # number data for line, colum and bloc in grid[r][c]
        Lr = grid[r][:]
        # adapt for column
        Cc = [grid[i][c] for i in range(n)]
        bloc = Lr+Cc
        # define bloc[k]
        infr, infc, supr, supc = 0, 0, 3, 3
        if r in [3,4,5] : infr, supr = 3, 6
        elif r in [6,7,8] : infr, supr = 6, 9
        if c in [3,4,5] : infc, supc = 3, 6
        elif c in [6,7,8] : infc, supc = 6, 9
        for k in range(infr,supr):
            for l in range(infc,supc) :
                u = grid[k][l]
                bloc.append(u)
        p = len(bloc)
        for m in range(p) :
            # without number 0 in bloc
            if 0 in bloc : bloc.remove(0)
            possible = []
            for i in range(1,n+1) :
                # possible for r,c
                if i not in bloc : possible.append(i)
        a = grid[r][c]
        if a == 0 :
            # write possible numbers
            p = len(possible)
            for u in possible :
                # (x, y) coordinate center
                x, y = c+.5+coordinate[u-1][0], 9-r-.5+coordinate[u-1][1]
                writeposition[u][count[u]] = x
                writeposition[u][count[u]+1] = y
                count[u] = count[u]+2 # modif 2 en 1 le 12 / 07
            # write possible number in grid sudoku
            for i in range(1,n+1):
                my_file.write(chr(92)+'multiput{' +chr(36)+chr(92)+'scriptstyle{' + str(i)+'}' \
                    +chr(36)+'} at ')
            for j in range(count[i]):
                my_file.write(' '+str(writeposition[i][j]))+' ')
            my_file.write(chr(47)+'\n')
my_file.write(chr(92)+'endpicture$$')
my_file.close()

```

Don't forget to write the data grid, row by row. Ne pas oublier d'écrire la grille de données, lignes par lignes.

Name of data fich, without .txt : **su_ex_11.txt**

```

$$\beginpicture\xyunits<24pt,24pt\measures(0,9)(0,9)
\input sudoku_grid_9x9.txt \input sudoku_mark_9x9.txt
% data

```

```

\color{red}
\multiput{\$\mathbf{1\$} at 1.5 8.5 3.5 6.5 8.5 5.5 5.5 4.5 2.5 3.5 4.5 2.5 0.5 1.5 7.5 0.5 /
\multiput{\$\mathbf{2\$} at 6.5 6.5 4.5 5.5 8.5 4.5 0.5 3.5 7.5 2.5 3.5 1.5 2.5 0.5 /
\multiput{\$\mathbf{3\$} at 3.5 8.5 8.5 7.5 2.5 6.5 1.5 5.5 7.5 4.5 4.5 3.5 0.5 2.5
5.5 1.5 6.5 0.5 /
\multiput{\$\mathbf{4\$} at 7.5 5.5 3.5 3.5 2.5 1.5 4.5 0.5 /
\multiput{\$\mathbf{5\$} at 4.5 8.5 7.5 6.5 6.5 5.5 5.5 3.5 3.5 2.5 1.5 1.5 8.5 0.5 /
\multiput{\$\mathbf{6\$} at 4.5 7.5 3.5 5.5 6.5 4.5 1.5 3.5 2.5 2.5 7.5 1.5 5.5 0.5 /
\multiput{\$\mathbf{7\$} at 7.5 7.5 4.5 6.5 2.5 5.5 3.5 4.5 8.5 3.5 5.5 2.5 6.5 1.5 /
\multiput{\$\mathbf{8\$} at 3.5 7.5 5.5 5.5 7.5 3.5 4.5 1.5 /
\multiput{\$\mathbf{9\$} at 7.5 8.5 2.5 7.5 0.5 5.5 4.5 4.5 6.5 3.5 1.5 2.5 8.5 1.5 3.5 0.5 /
% possibles
\color{blue}
\multiput{\$\scriptstyle{1\$} at 0.2 7.8 3.2 5.8 3.2 3.8 6.2 3.8 8.2 3.8 0.2 2.8 2.2 2.8 8.2 2.8
2.2 0.8 8.2 0.8 /
\multiput{\$\scriptstyle{2\$} at 7.5 8.8 3.5 6.8 5.5 6.8 8.5 6.8 7.5 4.8 8.5 4.8
0.5 3.8 6.5 3.8 7.5 3.8 8.5 3.8 0.5 2.8 1.5 2.8 5.5 2.8 8.5 2.8 0.5 1.8 3.5 1.8 6.5 1.8
7.5 1.8 8.5 1.8 3.5 0.8 5.5 0.8 7.5 0.8 8.5 0.8 /
\multiput{\$\scriptstyle{3\$} at 1.8 8.8 5.8 4.8 7.8 1.8 /
\multiput{\$\scriptstyle{4\$} at 5.2 7.5 6.2 7.5 8.2 7.5 5.2 6.5 8.2 6.5 2.2 5.5
6.2 3.5 8.2 3.5 /
\multiput{\$\scriptstyle{5\$} at 0.5 7.5 1.5 7.5 3.5 7.5 1.5 6.5 2.5 6.5 3.5 6.5 4.5 6.5
0.5 5.5 1.5 5.5 2.5 5.5 7.5 5.5 2.5 4.5 7.5 4.5 8.5 4.5 0.5 2.5 1.5 2.5 2.5 2.5
4.5 2.5 8.5 2.5 0.5 1.5 2.5 1.5 3.5 1.5 7.5 1.5 8.5 1.5 2.5 0.5 3.5 0.5 7.5 0.5 8.5 0.5 /
\multiput{\$\scriptstyle{6\$} at 1.8 8.5 4.8 8.5 7.8 8.5 0.8 7.5 1.8 7.5 3.8 7.5 7.8 7.5 8.8 7.5
1.8 6.5 2.8 6.5 3.8 6.5 4.8 6.5 8.8 6.5 2.8 4.5 4.8 4.5 0.8 3.5 3.8 3.5
0.8 2.5 1.8 2.5 2.8 2.5 8.8 2.5 2.8 0.5 7.8 0.5 8.8 0.5 /
\multiput{\$\scriptstyle{7\$} at 7.2 7.2 8.2 7.2 1.2 5.2 2.2 5.2 5.2 5.2 7.2 5.2
2.2 4.2 4.2 4.2 5.2 4.2 7.2 4.2 8.2 4.2 7.2 3.2 8.2 3.2
1.2 2.2 2.2 2.2 4.2 2.2 5.2 2.2 2.2 1.2 /
\multiput{\$\scriptstyle{8\$} at 0.5 7.2 3.5 7.2 5.5 7.2 2.5 6.2 3.5 6.2 4.5 6.2 5.5 6.2
3.5 5.2 5.5 5.2 0.5 2.2 2.5 2.2 4.5 2.2 5.5 2.2 0.5 1.2 2.5 1.2 3.5 1.2 7.5 1.2
2.5 0.2 3.5 0.2 5.5 0.2 7.5 0.2 /
\multiput{\$\scriptstyle{9\$} at 3.8 7.2 5.8 7.2 7.8 7.2 8.8 7.2 3.8 6.2 5.8 6.2 8.8 6.2 7.
8 5.2 3.8 1.2 /
\endpicture

```

We obtain the so-called “battle” file `su_ex_11.txt` Nous obtenons le fichier dit de “bataille” `su_ex_11.txt`

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

Figure 2: example battle

We first choose the boxes which contain only one number (obligatory winner) line by line with the **associated elimination**

Nous choisissons en premier les cases qui ne contiennent qu'un seul numéro (gagnant obligatoirement) ligne par lignes avec l'**élimination associée**

L_1 : 3 in C_2 , 6 in C_5 , 2 in $C_8 \rightarrow L_1$ is winning

L_2 : 1 in C_1 , 4 in C_7

L_4 : 5 in C_1 , 7 in C_2 , 4 in C_3 , 1 in C_4 , 8 in C_6 , 9 in $C_8 \rightarrow L_4$ is winning

L_5 : 6 in C_3 , 7 in C_5 , 3 in C_6 , 5 in C_8 , 2 in $C_9 \rightarrow L_5$ is winning

L_6 : 6 in $C_4 \rightarrow$ block B_5 is winning, 2 in $C_1 \rightarrow$ block B_4 is winning, 1 in C_7 , 7 in C_8 , 4 in $C_9 \rightarrow$ block B_6 is winning

L_8 : 8 in C_1 , 9 in C_4 , 2 in C_7 , 5 in C_9

L_7 : 6 in C_1 , 8 in C_5

L_9 : 2 in C_6

we continue column by column

nous continuons colonne par colonne

C_1 is winning

C_2 : {6, 8} is in L_2C_2 and $L_3C_3 \rightarrow$ not in L_7 and L_3C_3 , 2 in L_7

C_3 : 8 in L_3

C_4 : {5, 8} is in L_2C_4 and L_9C_4 not in L_3 , 2 in L_3

C_5 : 5 in $L_3 \rightarrow C_5$ is winning \rightarrow 6 in L_3C_2 , 7 in L_2C_2 , 8 in L_2C_4 , 5 in L_9C_4 , 1 in L_9C_3

C_6 : 9 in L_2 , 4 in $L_3 \rightarrow$ block B_2 is winning, 7 in L_7 , 5 in L_7C_3 , 7 in $L_8C_3 \rightarrow$ block B_7 is winning

C_8 : 6 in L_2 , 3 in L_8 , 8 in $L_9 \rightarrow C_8$ is winning

C_9 : 9 in L_3 , 1 in L_7 , 6 in L_9

We obtain a winning grid.

Nous obtenons une grille gagnante.

We have taken and completed part of the **su_ex_11.txt** file which gives the **su_ex_12.txt** file

Nous avons repris et complété une partie du fichier **su_ex_11.txt** qui donne le fichier **su_ex_12.txt**

```
\multiput{\scriptstyle{1}} at 0.2 7.8 3.2 5.8 6.2 3.8 8.2 2.8 2.2 0.8 /
\multiput{\scriptstyle{2}} at 7.5 8.8 3.5 6.8 8.5 4.8 0.5 3.8 1.5 2.8 6.5 1.8 5.5 0.8 /
\multiput{\scriptstyle{3}} at 1.8 8.8 5.8 4.8 7.8 1.8 /
\multiput{\scriptstyle{4}} at 6.2 7.5 5.2 6.5 2.2 5.5 8.2 3.5 /
\multiput{\scriptstyle{5}} at 1.5 7.5 4.5 6.5 0.5 5.5 7.5 4.5 2.5 2.5 8.5 1.5 3.5 0.5 /
\multiput{\scriptstyle{6}} at 4.8 8.5 7.8 7.5 1.8 6.5 2.8 4.5 3.8 3.5 0.8 2.5 8.8 0.5 /
\multiput{\scriptstyle{7}} at 8.2 7.2 1.2 5.2 4.2 4.2 7.2 3.2 5.2 2.2 2.2 1.2 /
\multiput{\scriptstyle{8}} at 3.5 7.2 2.5 6.2 5.5 5.2 4.5 2.2 0.5 1.2 7.5 0.2 /
\multiput{\scriptstyle{9}} at 5.8 7.2 8.8 6.2 7.8 5.2 3.8 1.2 /
```

su_ex_12.txt

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

Figure 3: example victory