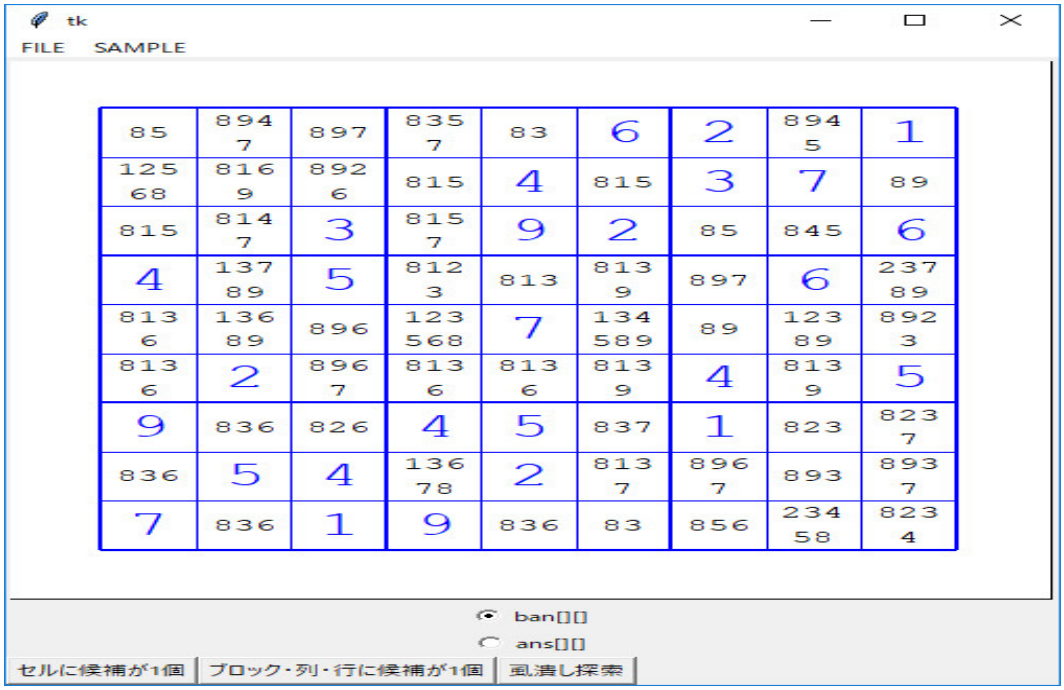
高知大学教育学部の情報数学のテキスト

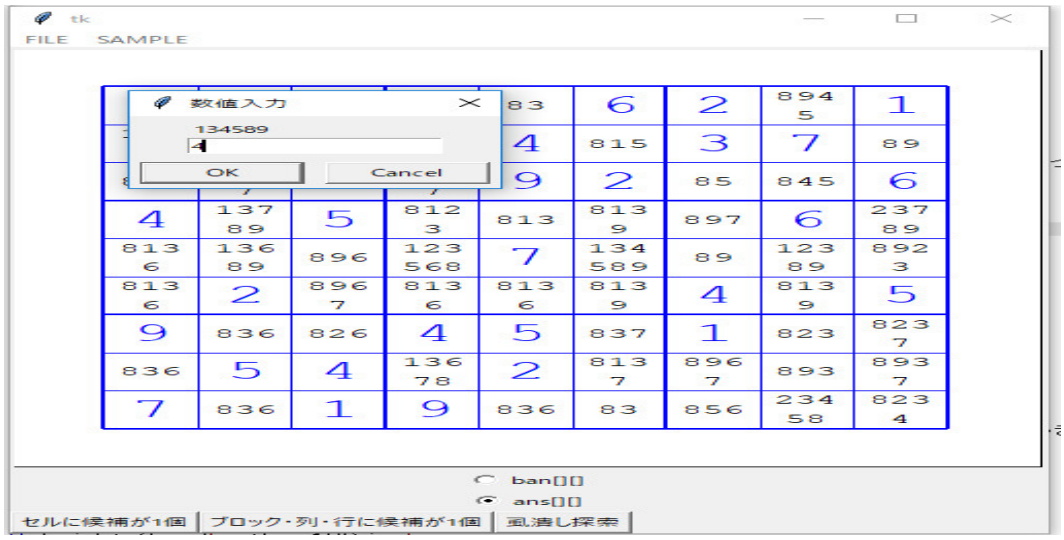
文責　：　高知大学名誉教授　中村　治

数独のヒントを表示してくれるプログラム (Python 版)

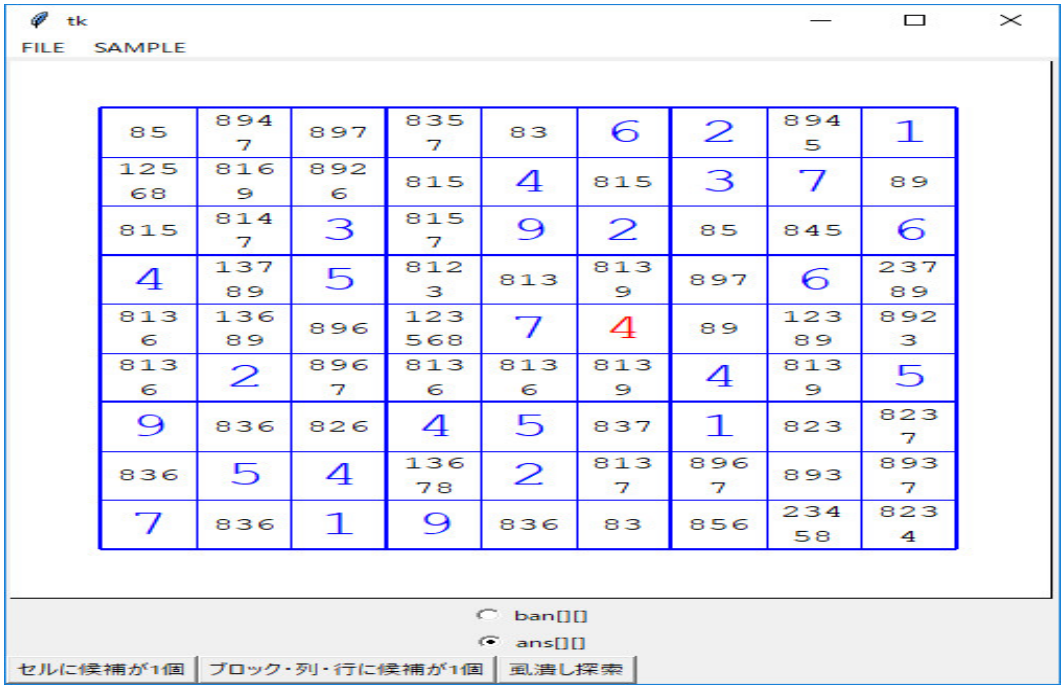
情報数学のコンピュータ言語を C++ から Python に変えたので、数独のヒントを表示してくれるプログラムも Python で作ってみました。



マウスで数字を入れたいセルをクリックすると



数値の入力を促すダイアログボックスが表示されるので、数字を入力すると



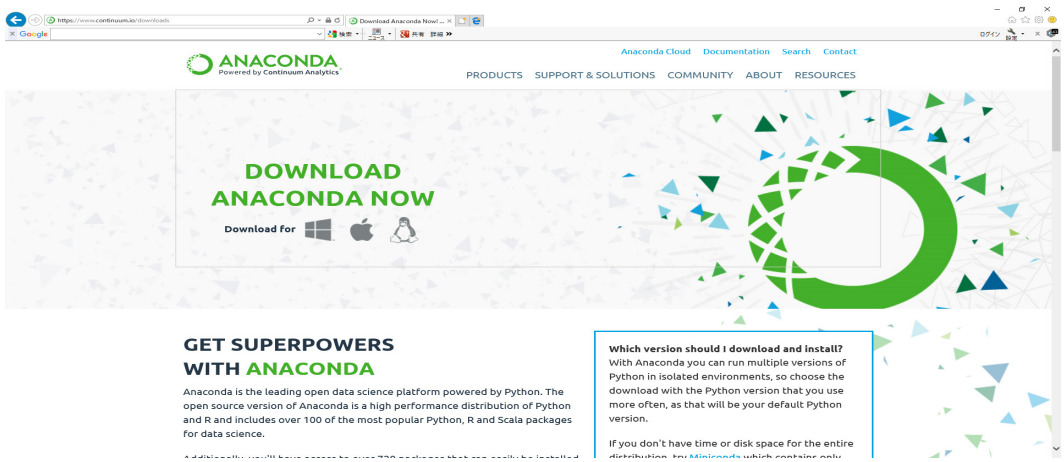
のように盤面に数字が入力出来ます。

現在 Python2 と Python3 が利用できますが、Python 2.7 と Python 3.6 は上位互換性はなく、Python2 は新たな改善はなされないことが決まっているので、これからは Python3 を使って行くのが良いです。Python 2.7 は時代遅れになってきましたから、Python3.6 でプログラミングします。

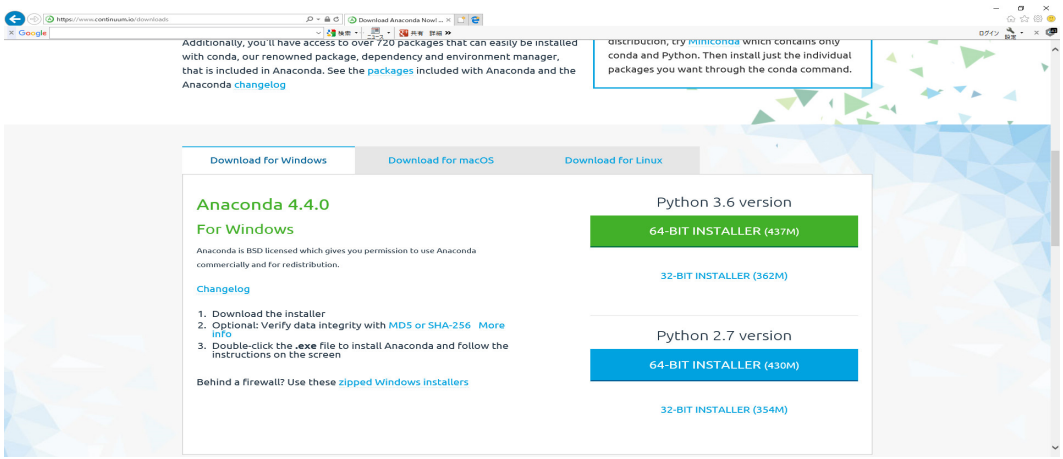
まず Python をインストールして、Python を使えるようにしましょう。ここでは Anaconda を使って Python をインストールします。Anaconda は Python 本体に加えて、よく使われるパッケージを一括してインストールできるようにしたものです。

`https://www.continuum.io/downloads`

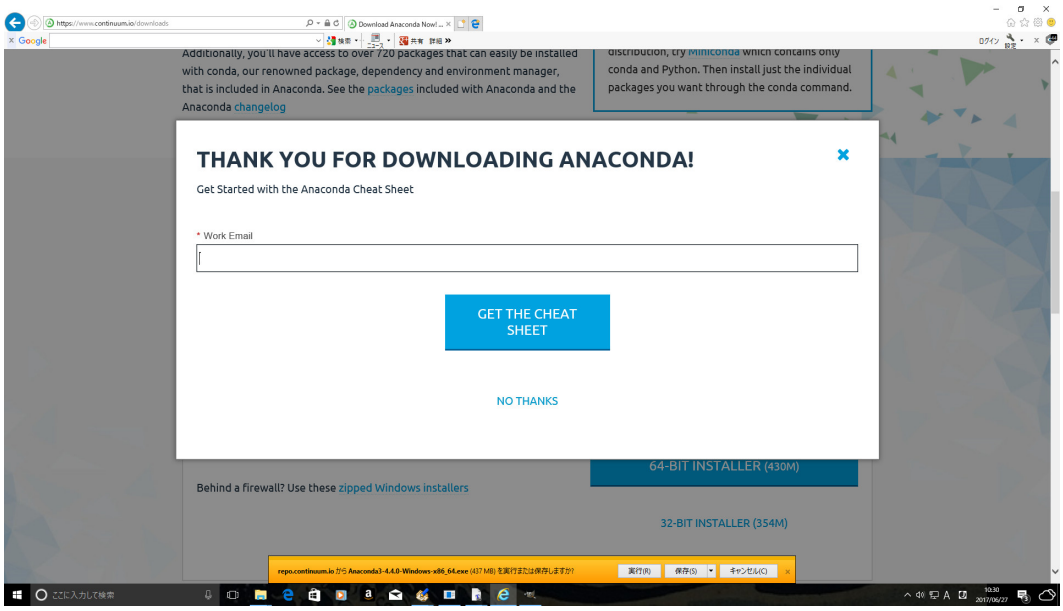
にアクセスします。



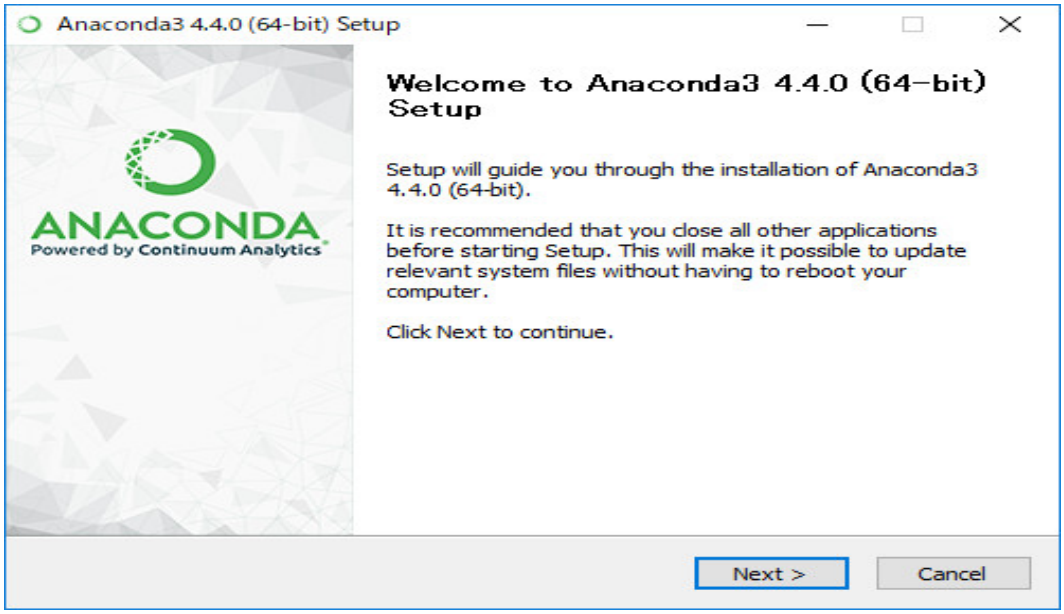
下の方の



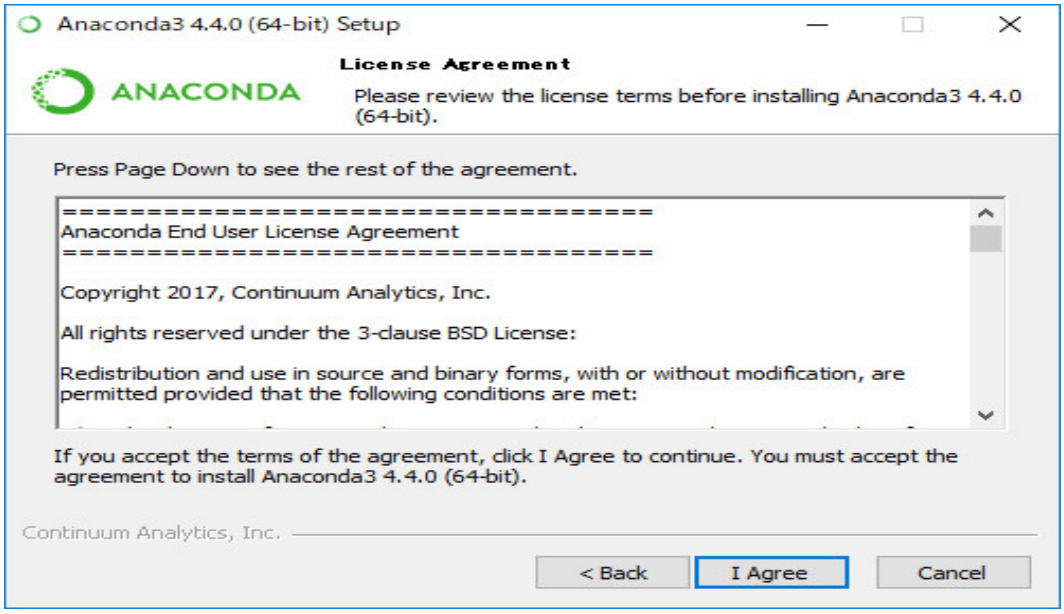
Python 3.6 version をクリックします。現在 Python2 と Python3 が利用できますが、Python 2.7 と Python 3.6 は上位互換性はなく、Python2 は新たな改善はなされないことが決まっているので、これからは Python3 を使って行くのが良いです。



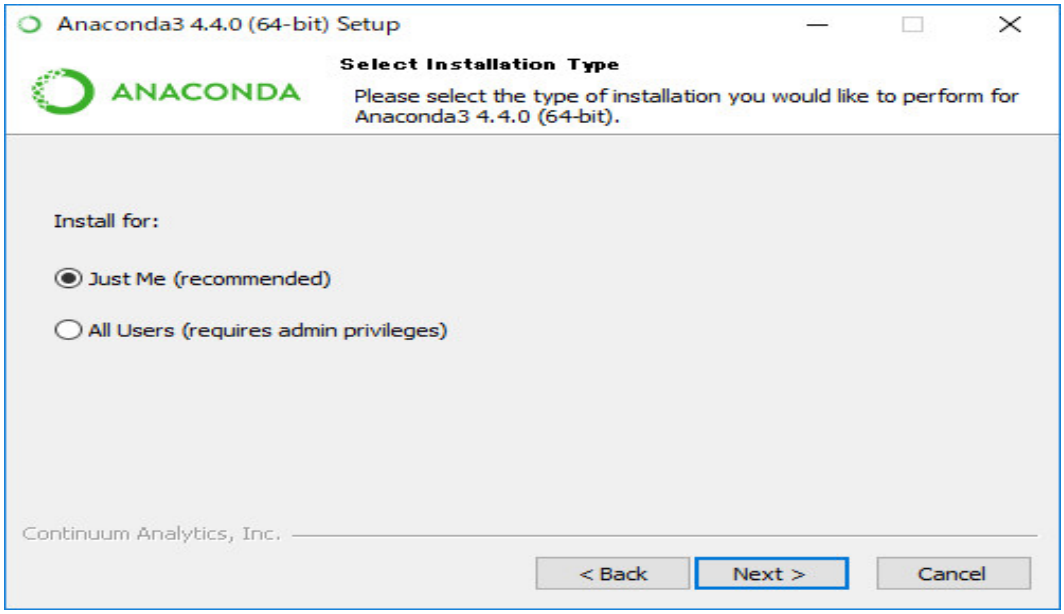
これは無視して、閉じて良いです。「実行」をクリックします。



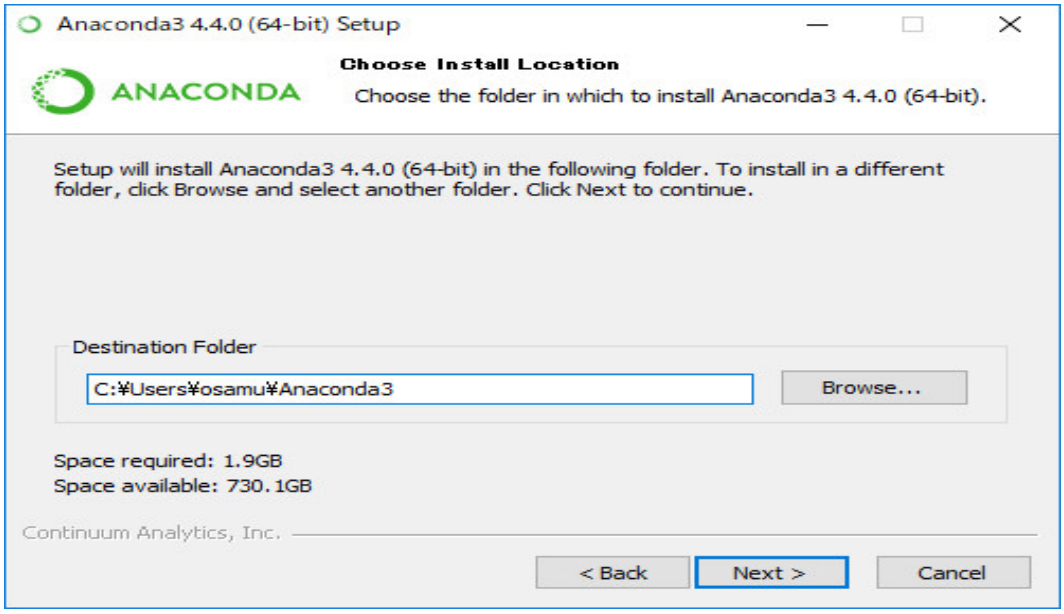
「Next」をクリックします。



「I Agree」をクリックします。



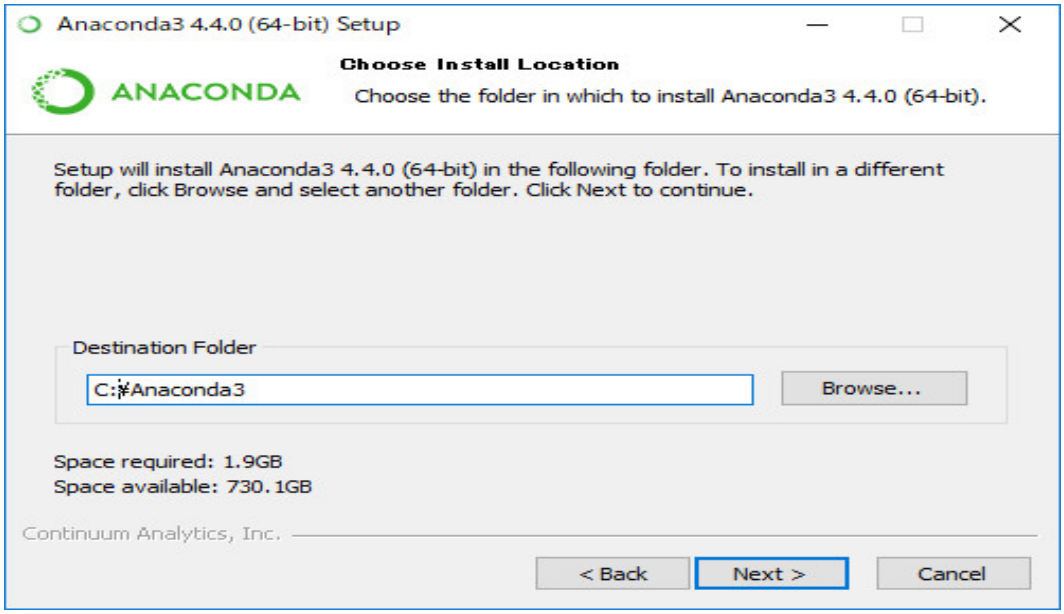
「Next」をクリックします。



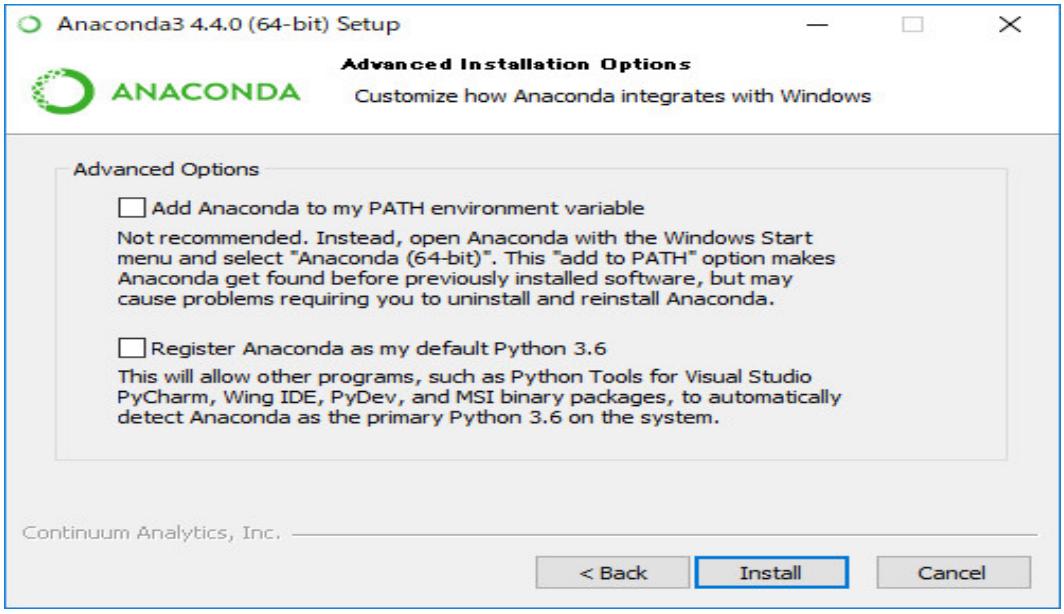
Destination Folder がデフォルトでは、後で色々問題が起こるのでここを

`C:\Anaconda3`

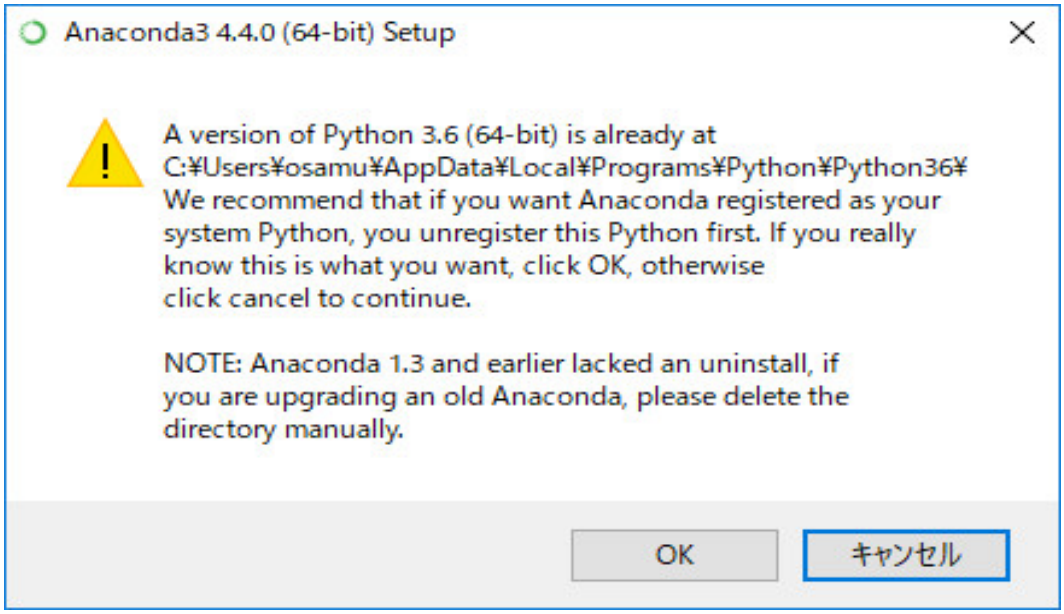
と修正します。



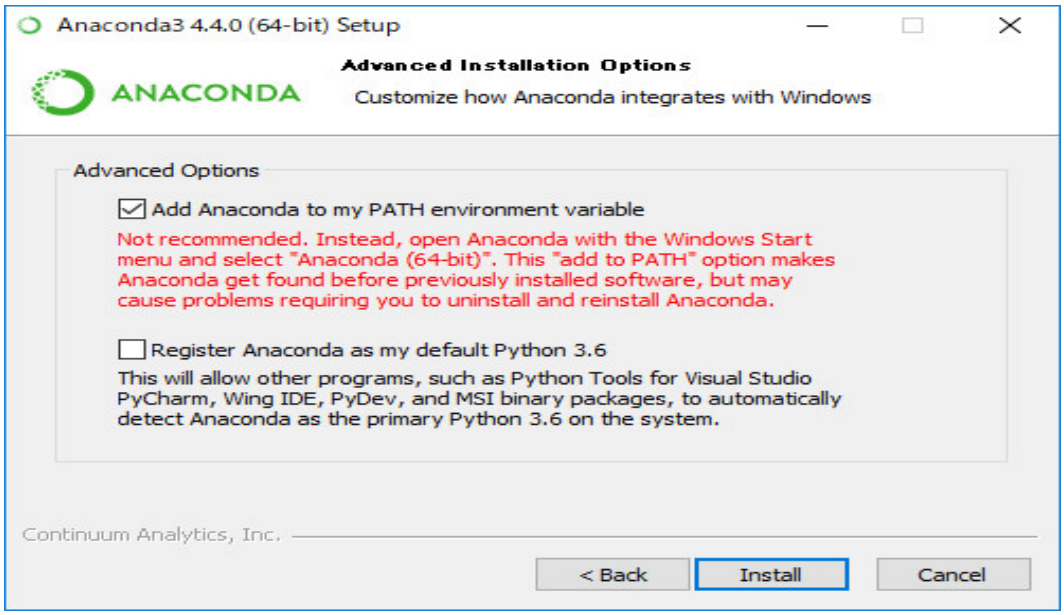
「Next」をクリックします。



通常は両方にチェックを入れます。しかし、すでに Python3.6 をインストールしている場合には、下にチェックを入れると

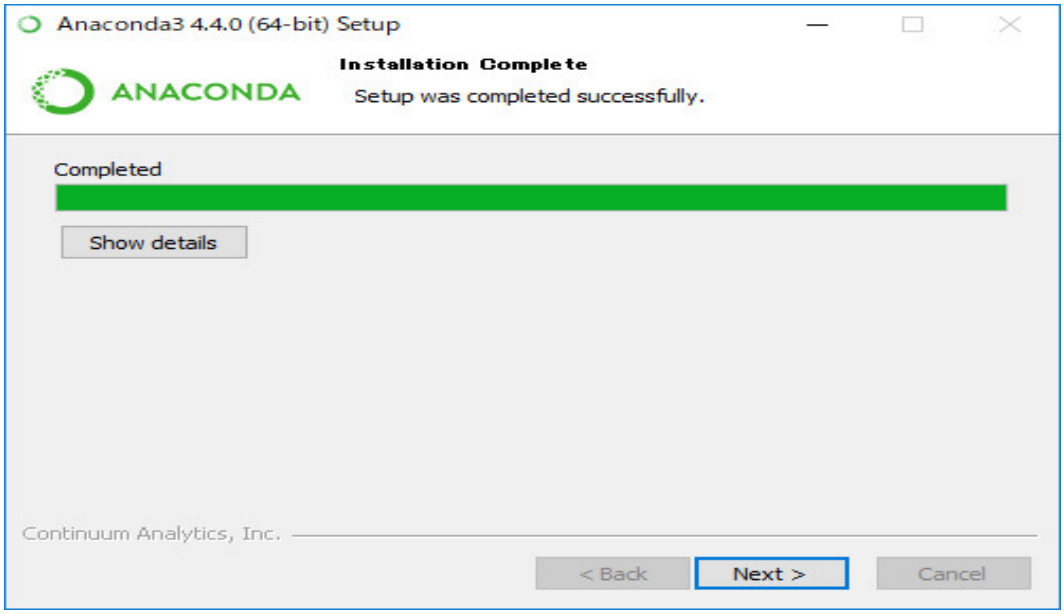


と表示されるので、「キャンセル」をクリックし、すでに Python3.6 をインストールしている場合には

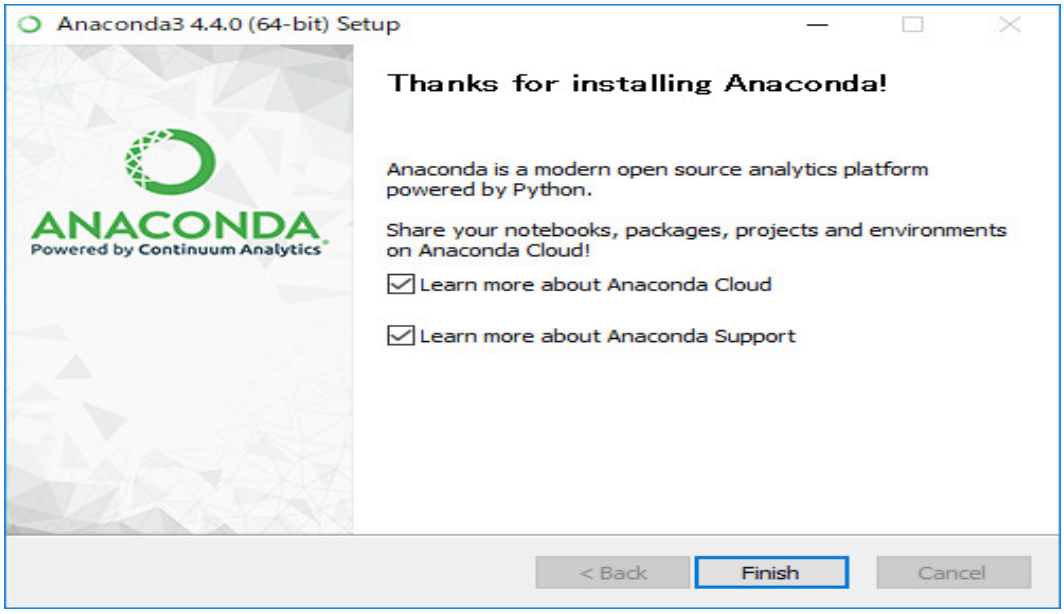


のように、上だけをチェックします。

「Install」をクリックします。

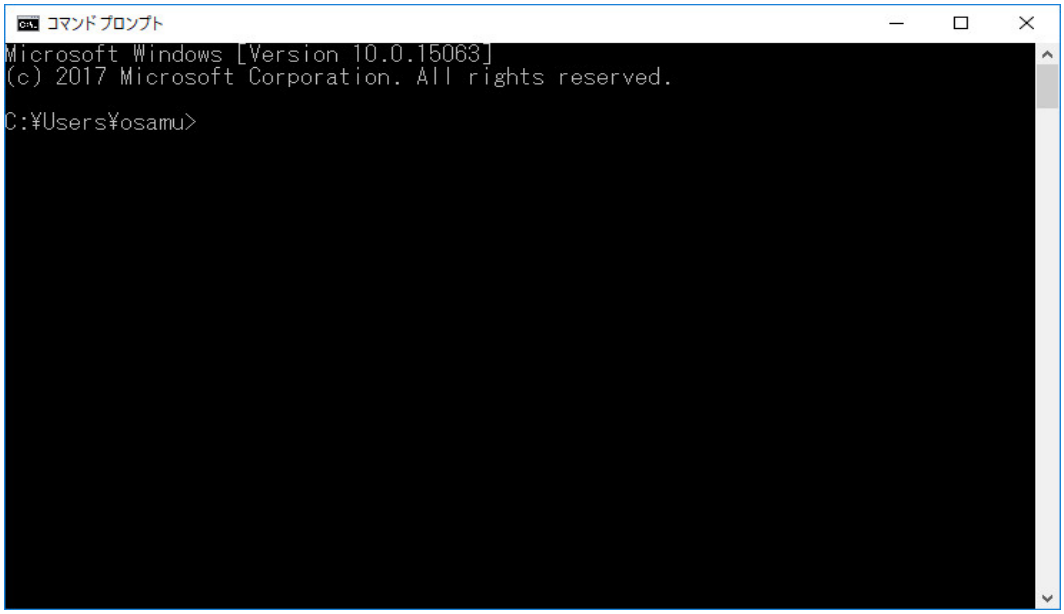


「Next」をクリックします。

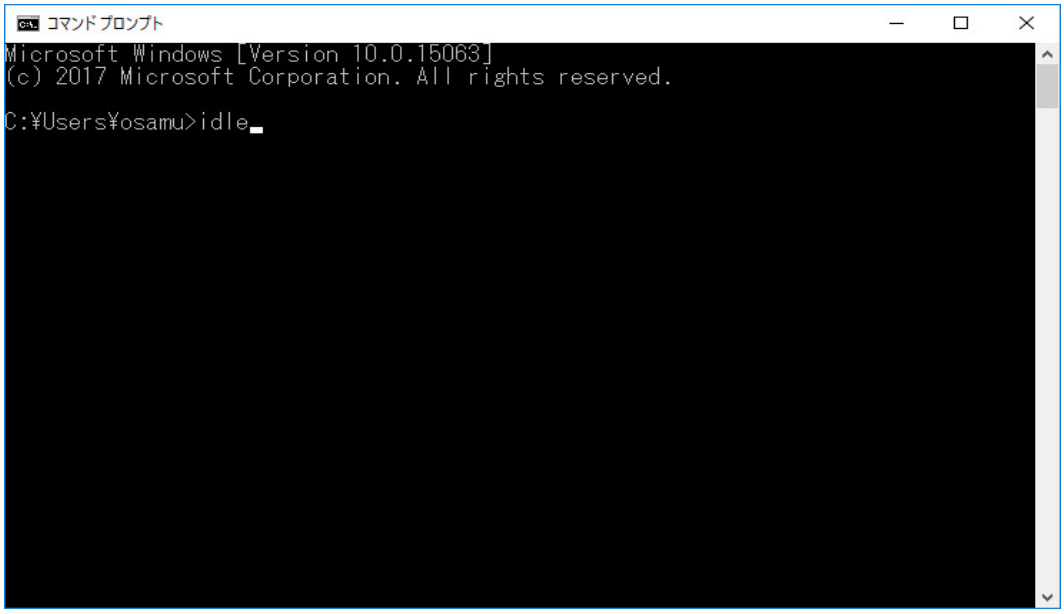


「Finish」をクリックします。

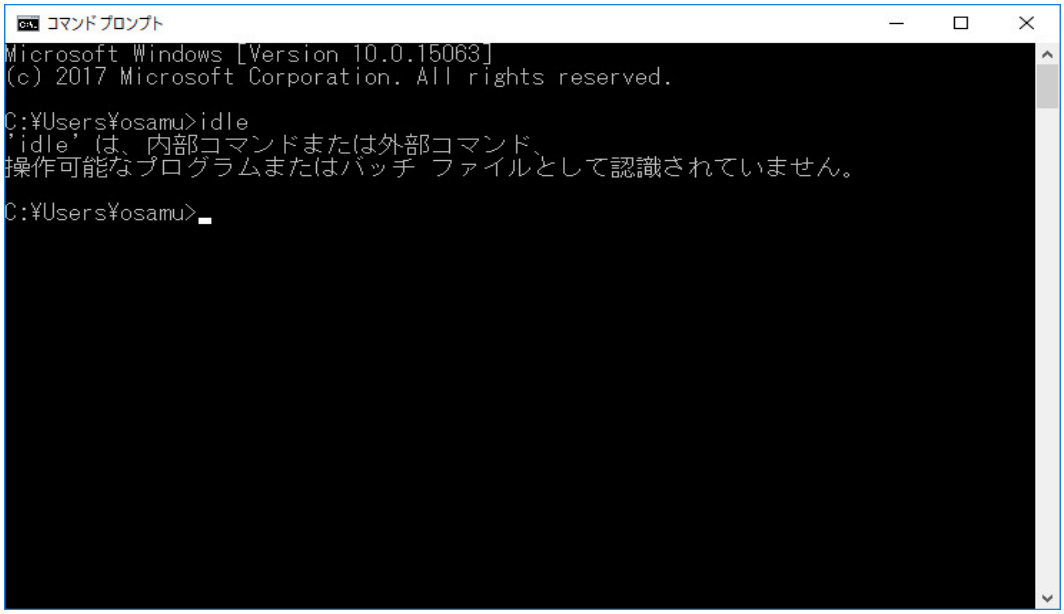
Anaconda をインストールした後に「コマンド　プロンプト」を起動し、


コマンド プロンプト
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
C:¥Users¥osamu>

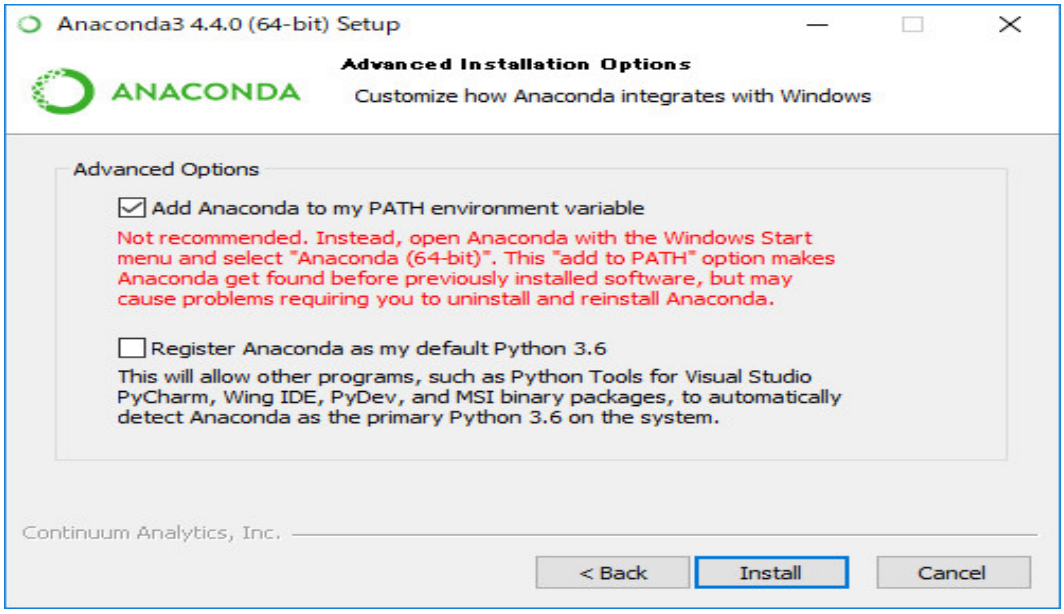
「idle」を


コマンド プロンプト
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
C:¥Users¥osamu>idle

実行します。

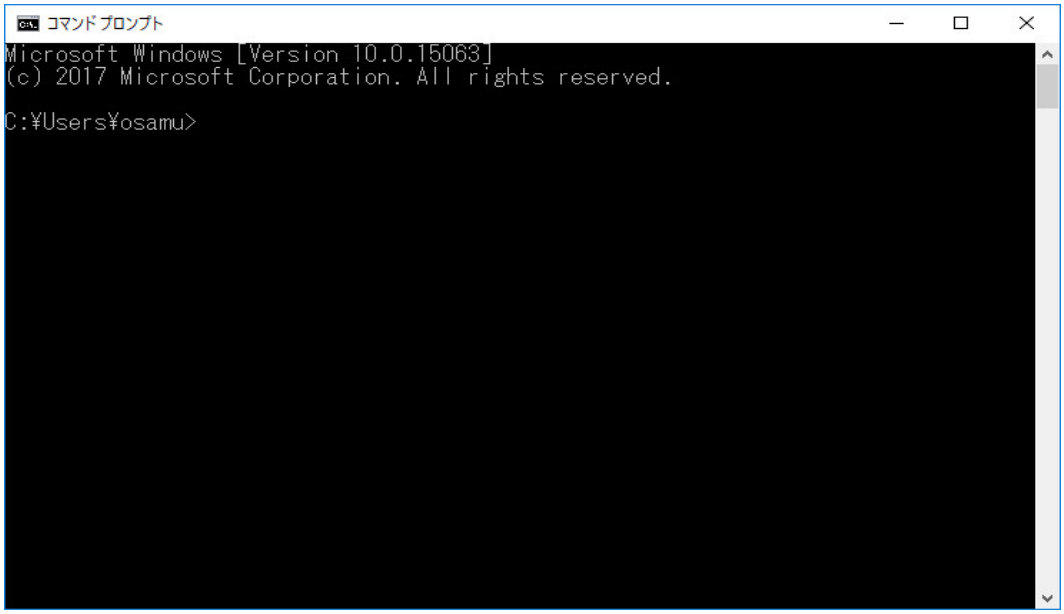


と表示されたら、

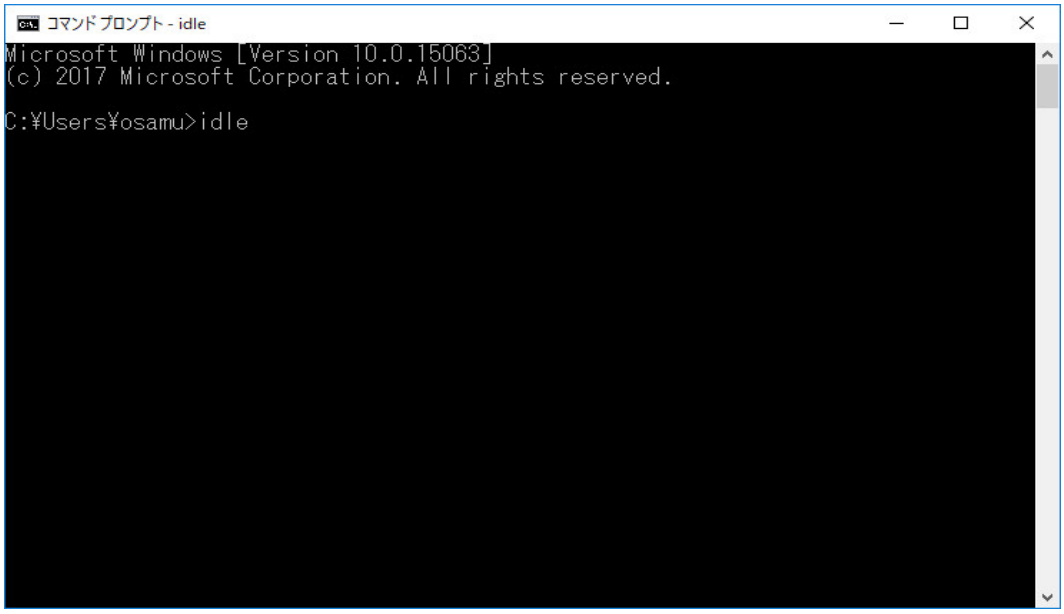


のチェックを忘れていた可能性があります。 Anaconda3 のフォルダーをエクスプローラーで削除して、もう一度インストールをやり直すのが簡単です。

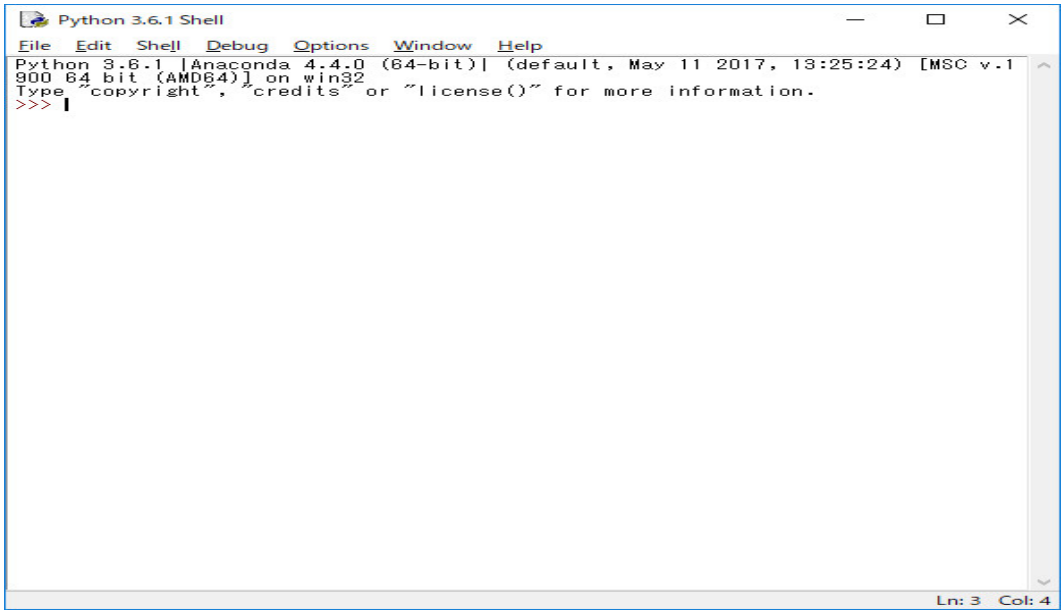
「コマンド　プロンプト」を起動し、


コマンド プロンプト
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
C:¥Users¥osamu>

「idle」を

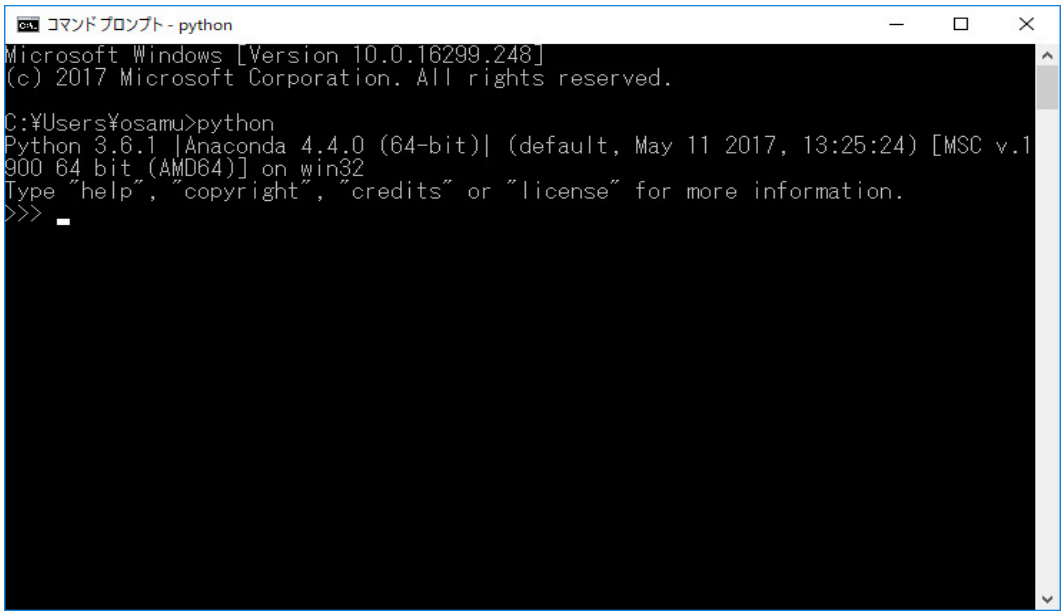

コマンド プロンプト - idle
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.
C:¥Users¥osamu>idle

実行します。



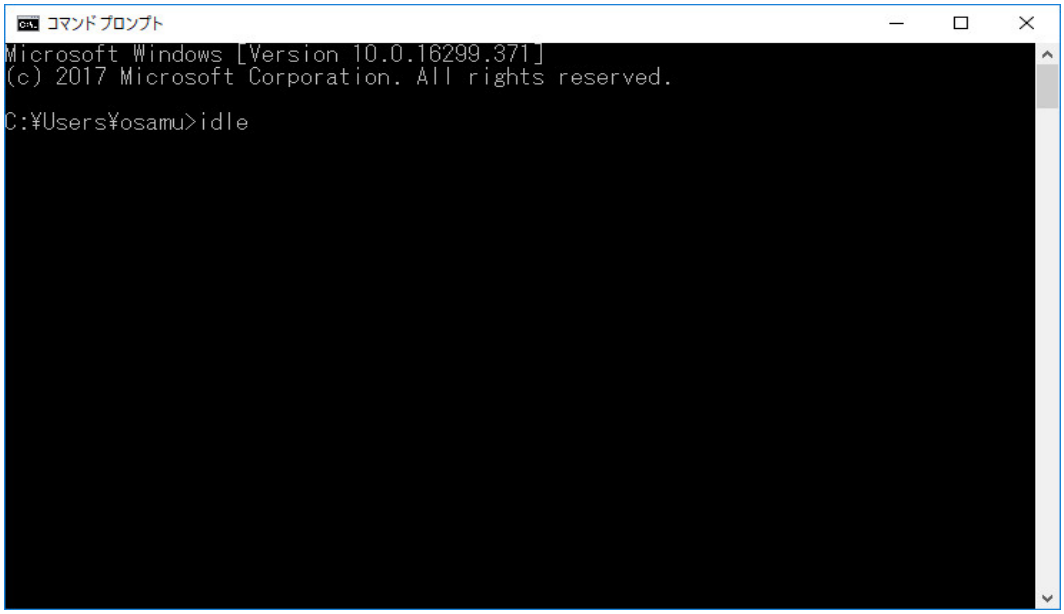
と「Python3.6.1 Shell」が起動したら、Anaconda のインストールは完成です。

「コマンド　プロンプト」を起動し、「python」を

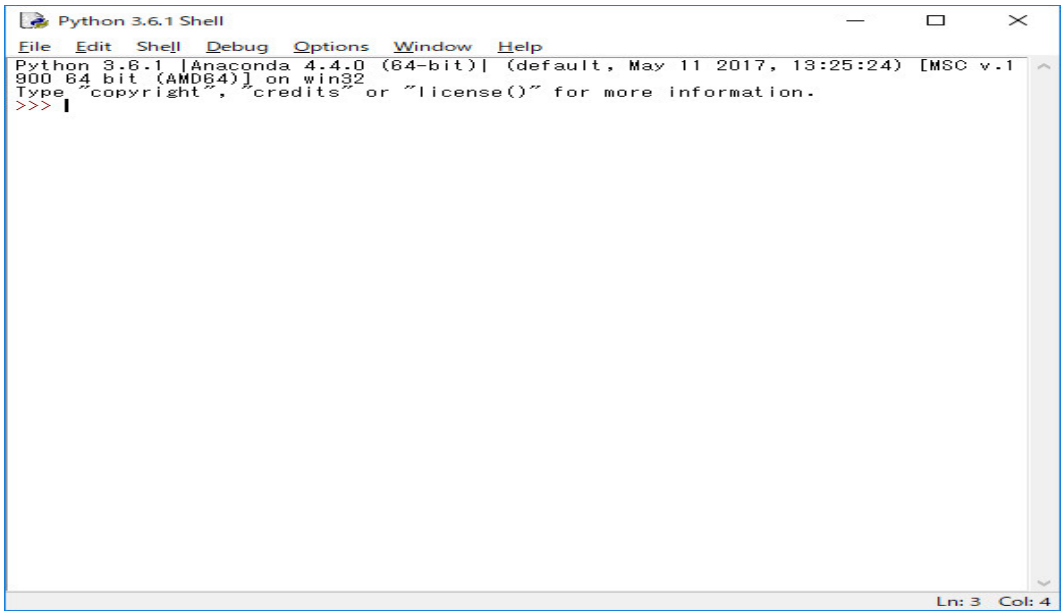


実行すると Python インタープリターが走り始めます。

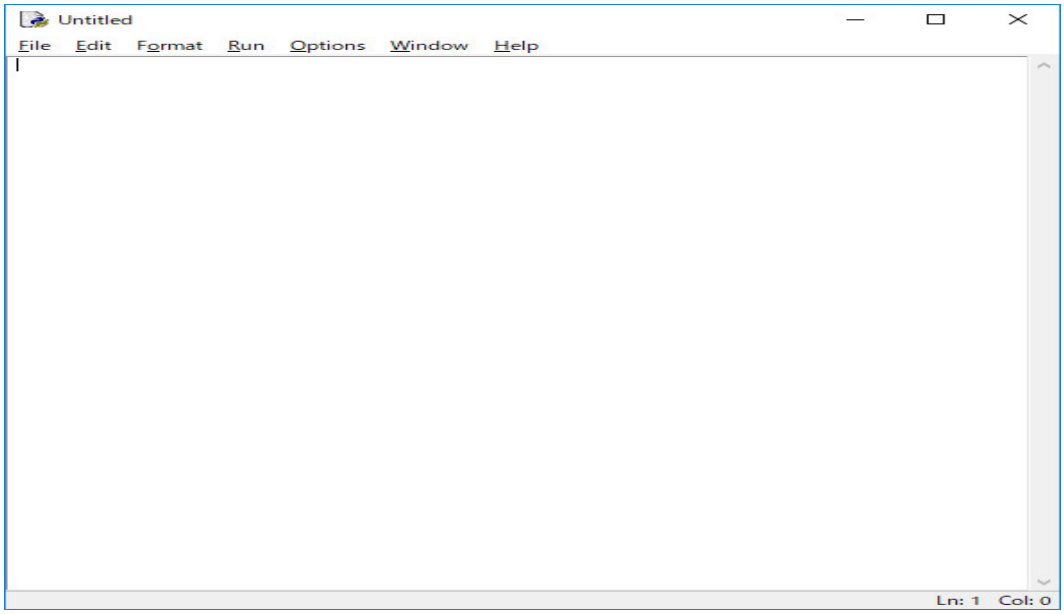
では、プログラムを作って行きます。コマンドプロンプトを立ち上げ、idle を実行します。



Python Shell が立ち上がります。



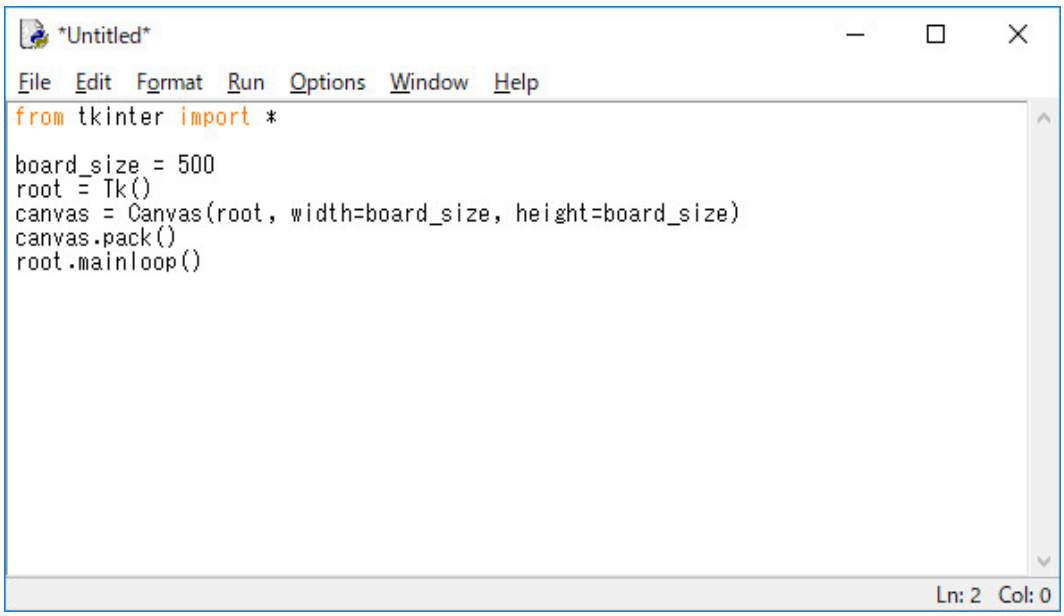
「File」 メニューの 「New File」をクリックします。



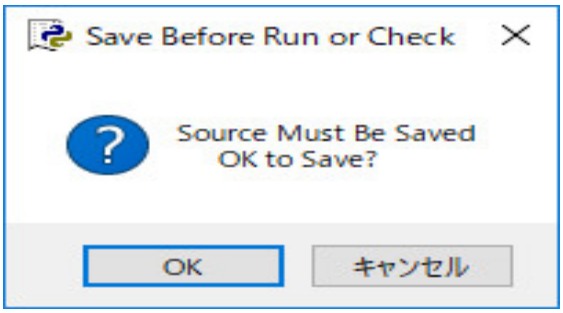
ここにプログラムを打ち込んでいきます。
まず

```
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
root.mainloop()
```

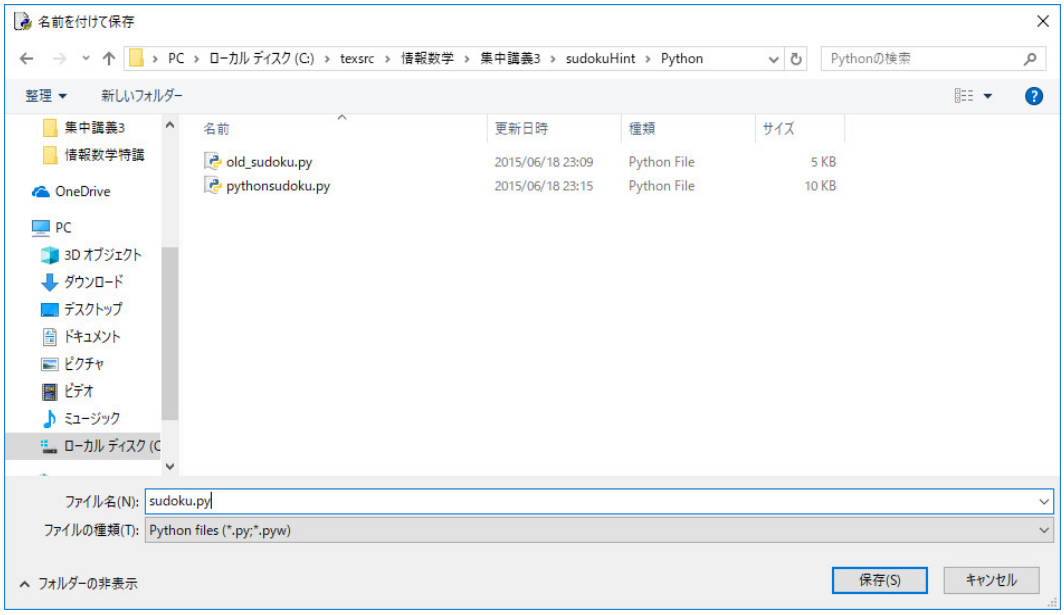
と打ち込みます。



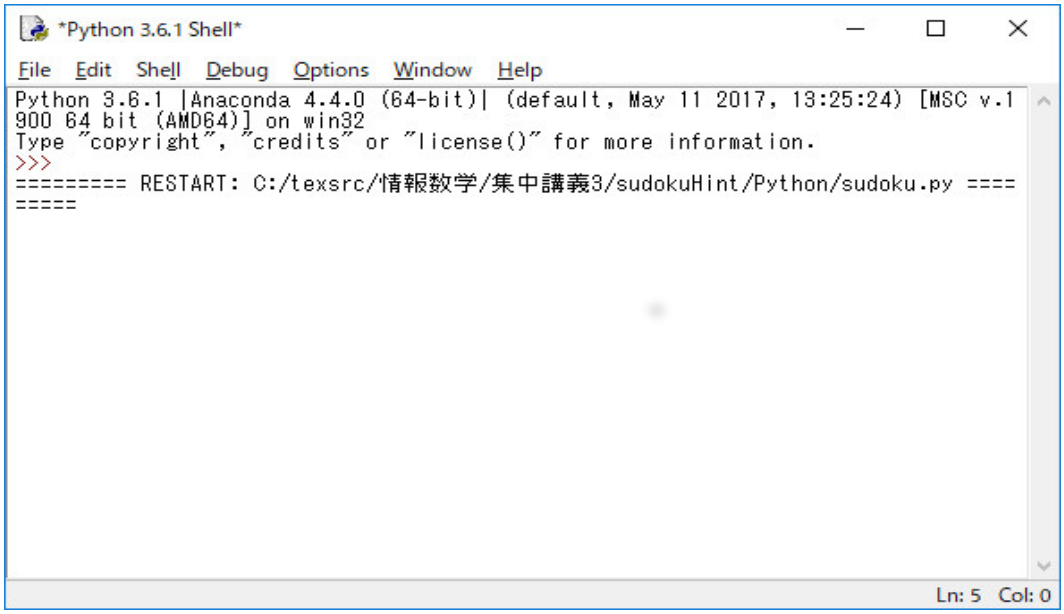
「Run」 メニューの 「Run Module」をクリックします。



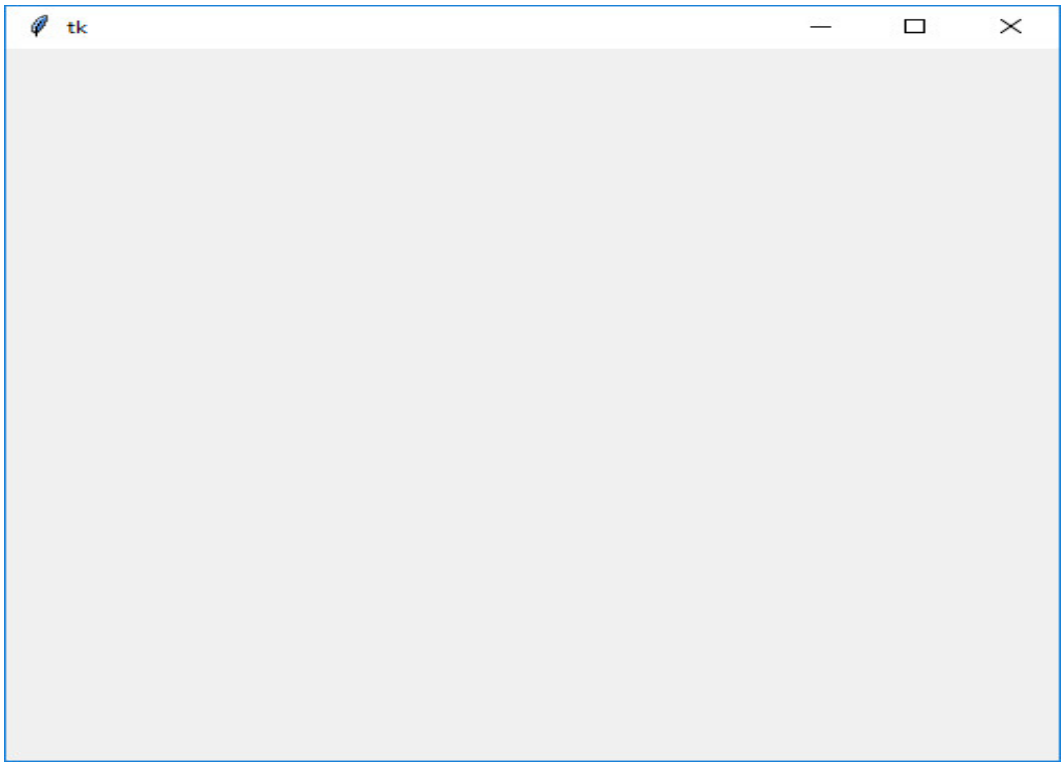
「OK」をクリックします。適当なファイル名を付けて



プログラムを保存します。プログラムが間違っていなければ



とプログラムが実行され、



のようなウインドウが開きます。プログラム

```
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
root.mainloop()
```

の簡単な説明をします。これは Python によるグラフィックスの最初に説明したものと殆ど同じです。Python でウインドウを使うプログラムでは普通、ライブラリー tkinter を使います。そのためにまず、

```
from tkinter import *
```

と tkinter をインポートします。インポートとは、Python の製作者が準備してくれているプログラム（ライブラリーと言います）を使えるようにすることです。最初にこのように書くものだと思ってください。理屈ではありません。次の

```
board_size = 500
```

は、変数 `board_size` に整数 500 を代入しています。次の次の行を

```
canvas = Canvas(root, width=500, height=500)
```

とするよりも、このように変数 board_size を使って

```
canvas = Canvas(root, width=board_size, height=board_size)
```

とする方が良いと一般に言われています。マジックナンバー 500 より board_size の方が、何を意味するかすぐ分かるからです。

```
root = Tk()
```

で、コンストラクタ Tk() を読んで、ウインドウを作り、それに root と名前を付けています。何をやっているか分からなくても気にしなくていいです。ウインドウを作りたい時はいつでもこのようにすれば良いです。

```
canvas = Canvas(root, width=board_size, height=board_size)
```

で、ウインドウ root の子供として（ウインドウ root の中に)、絵を描くためのキャンバス（サイズ 500 × 500）を作り、それに canvas という名前を付けています。

```
canvas.pack()
```

で、実際に canvas を root に貼り付けます（組み込みます)。最後に

```
root.mainloop()
```

で、無限ループで、ウインドウ上でマウス入力などを受け付けれるようにします。

兎も角、ライブラリー tkinter を使うプログラムは、理屈抜きに、いつでも

```
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
root.mainloop()
```

のプログラムを雛型として、これをコピーして、これを出発点としてプログラミングしていくと思ってください。

「File」 メニューの 「Open」を使って、保存したプログラムを読み込むことが出来ます。

数独の問題をキャンバス canvas に描いてみましょう。プログラムに

```
canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
```

```
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
```

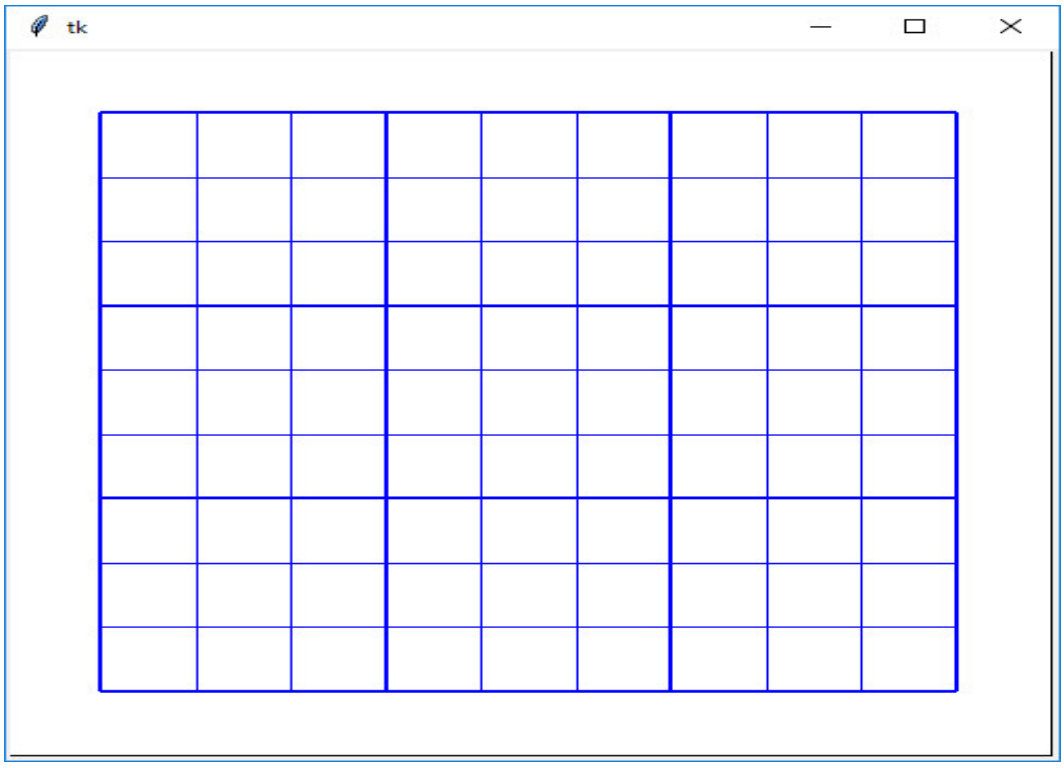
を追加します。プログラムの全体は

```
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()

canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')

root.mainloop()
```

となります。

```
sudoku.py - C:¥texsrc¥情報数学¥集中講義3¥sudokuHint¥Python¥sudoku.py (...
File  Edit  Format  Run  Options  Window  Help
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()

canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')

root.mainloop()
Ln: 24  Col: 0
```

実行すると



となります。

プログラムの解説をします。

```
canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
```

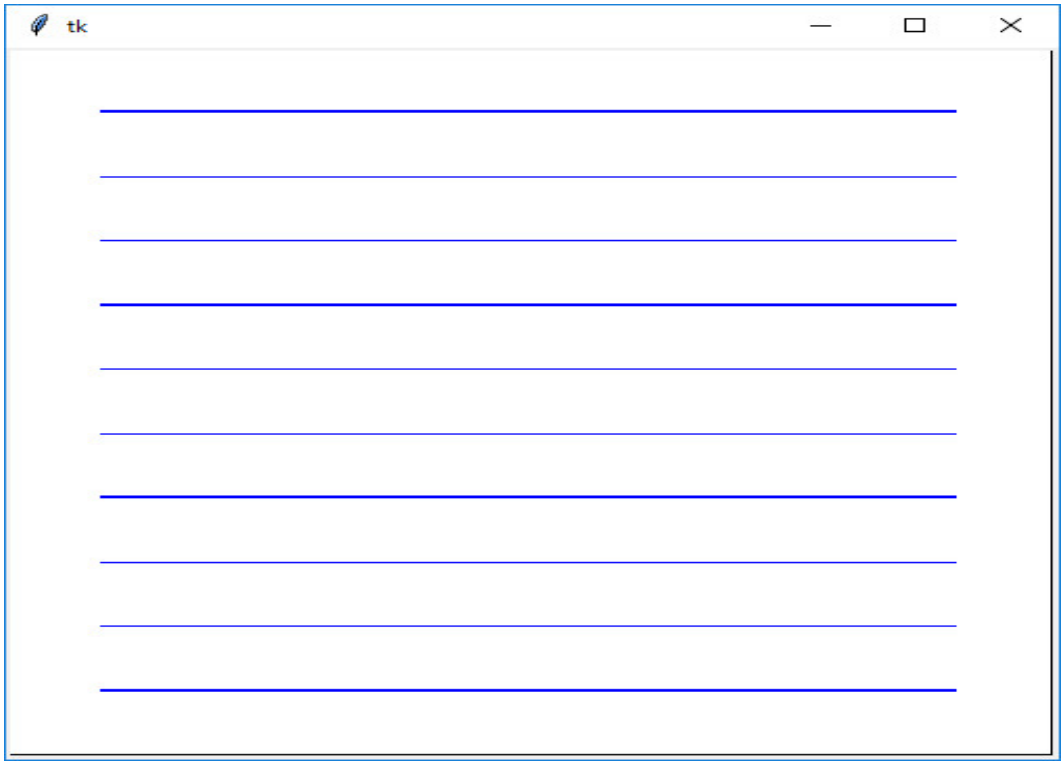
で、幅 board_size、 高さ board_size の矩形の左上隅 (0,0) を canvas の左上隅 (0,0) として、描き、white で塗りつぶしています。即ち、キャンバス全体を白く塗りつぶしています。

```
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
```

は、盤を描くための cell の大きさや盤の左上隅の座標を計算しています。

```
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
```

で、横線を引いています。



10 本の線を引くので、for 文を使って

```
for i in range(10):
   ......
```

としています。i が $0, 1, 2, \cdots, 9$ と変化しながら、インデント（字下げ）された部分の命令を実行します。

```
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
```

と、if 文で、i が 3 の倍数かどうかで、処理を分けています。i が 3 の倍数の時は

```
canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
```

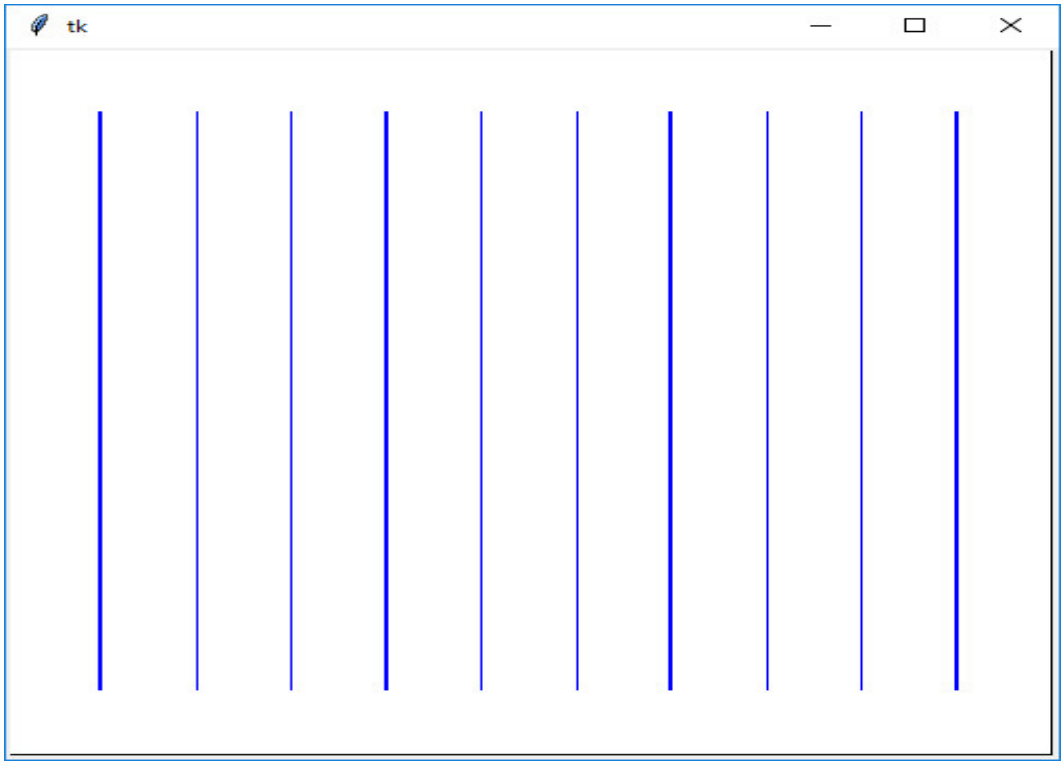
と create_line() 関数で、(w, h+i*s) と (w+9*s, h+i*s) を結ぶ直線を、blue の色で、幅 2.0 のペンで描いています。そうでないときは

```
canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
```

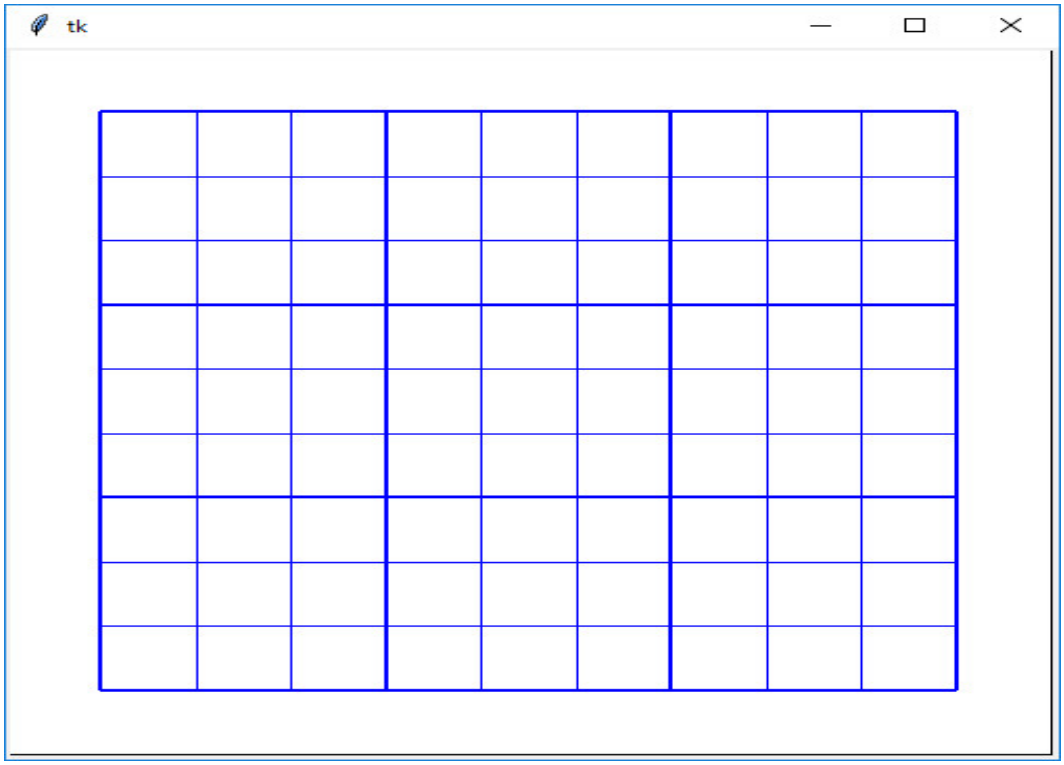
と、 create_line() 関数で、(w, h+i*s) と (w+9*s, h+i*s) を結ぶ直線を、blue の色で、幅 1.0（デフォルトですから、指定しなくてよい）のペンで描いています。

```
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
```

で、縦の線を描いています。



両方で



となります。次に、この盤に数字を描いてみます。文字を描くためにはフォントと色を指定しなければなりません。面倒です。簡単のため

```
f1 = "courier 24"
c1= "blue"
```

とします。

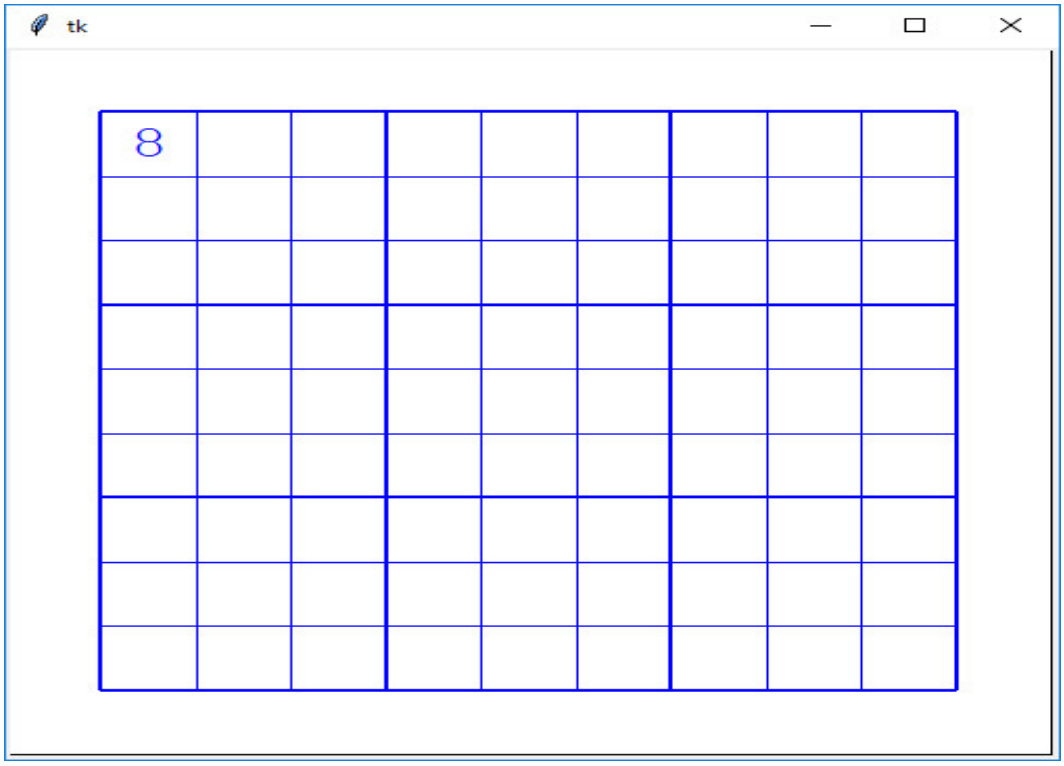
```
i = 0
k = 0
canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(8), font=f1, fill=c1)
```

のように、プログラミングします。プログラムの全体は

```
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()

canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
```

```
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')

f1 = "courier 24"
c1= "blue"
i = 0
k = 0
canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(8), font=f1, fill=c1)

root.mainloop()
```

となりました。実行すると

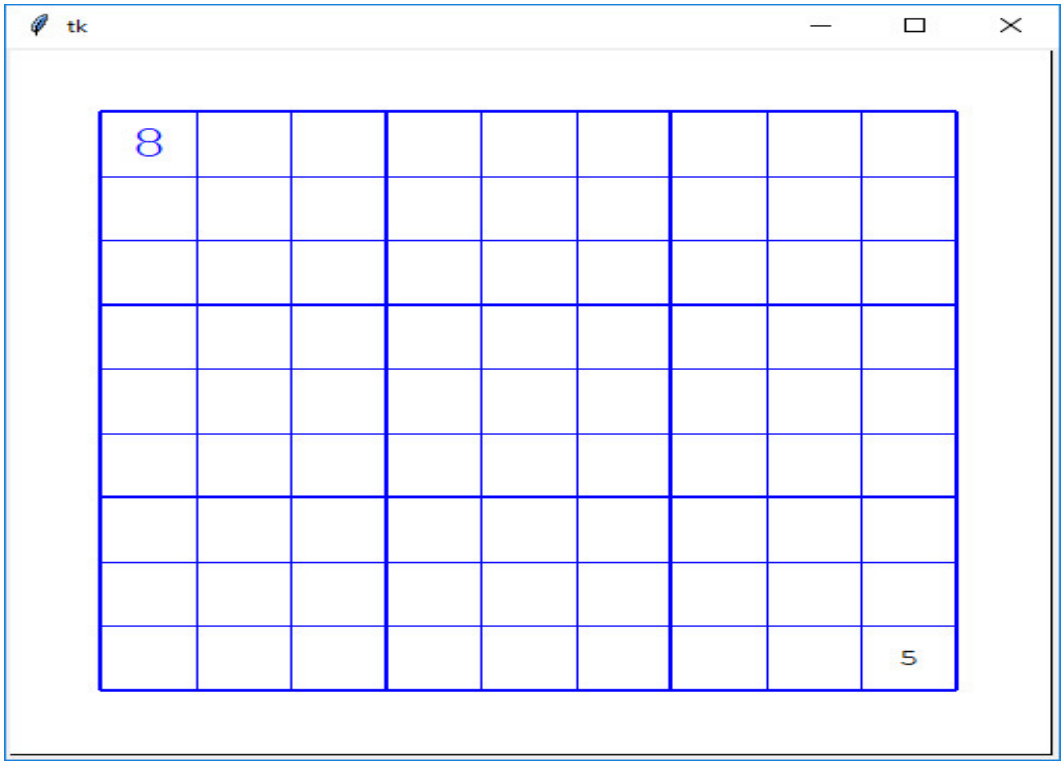


です。

```
f2 = "courier 12"
c3= "black"
```

```
i = 8
k = 8
canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(5), font=f2, fill=c3)
```

を追加し、実行すると



です。数字の表示方法が分かりました。数独の問題の数字の配置は変数 ban にセットしておいて、使います。

```
ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
```

のように、リストのリストで数独の問題の数字の配置を表すことにします。リストはオブジェクト（数字や文字列やリストなど）を並べて、 [ ] で囲んだものです。リストの要素は添数でアクセスできます。今の ban の場合、ban[0] は [0,1,0,0,0,0,9,0,3]、ban[1] は [0,9,0,0,0,7,0,0,0] 、ban[8] は [0,0,8,5,3,0,0,9,4] です。ban[0][0] は 0、ban[8][8] は 4 です。数字を伏せている所は 0 で表しています。これを使って、canvas に問題を表示してみます。

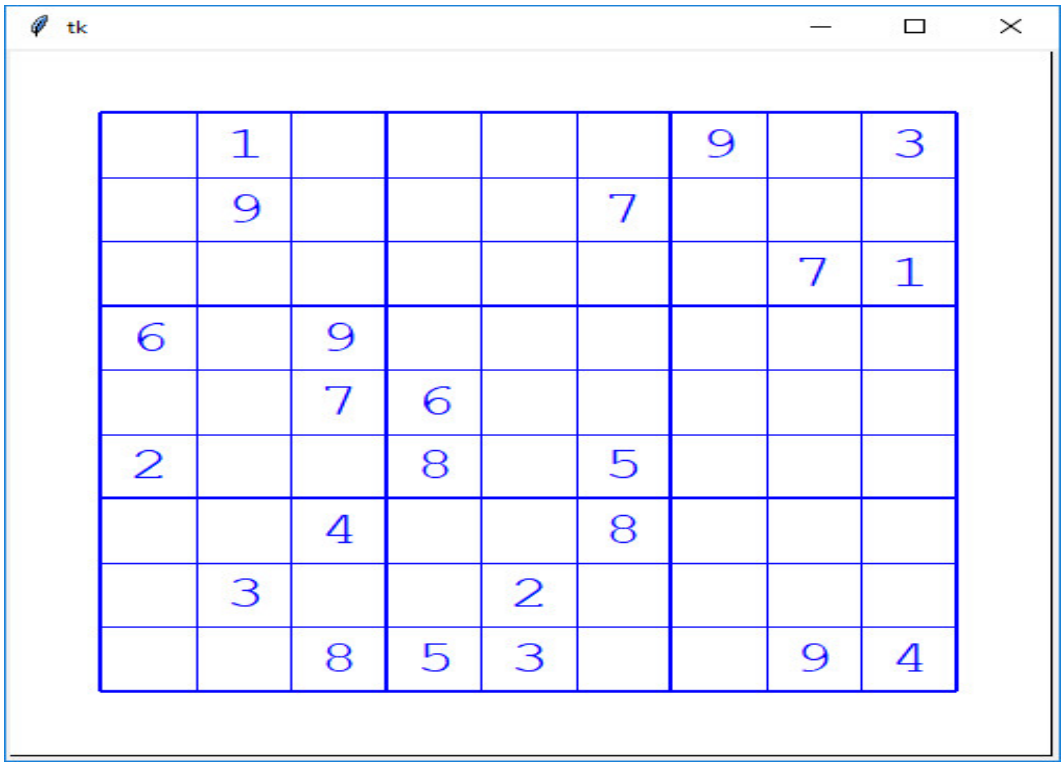
```
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in range(9):
    for i in range(9):
        if ban[k][i] > 0:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                        font=f1, fill=c1)
```

とします。プログラムの全体は

```
from tkinter import *

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()

canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')

ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
```

```
for k in range(9):
    for i in range(9):
        if ban[k][i] > 0:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                font=f1, fill=c1)

root.mainloop()
```

となります。実行すると



です。色々な問題を表示できるようにします。そのためにはメニューを作り、問題を選べるようにします。メニューの一般形は

```
def callback():
    print( 'called')
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=callback)
```

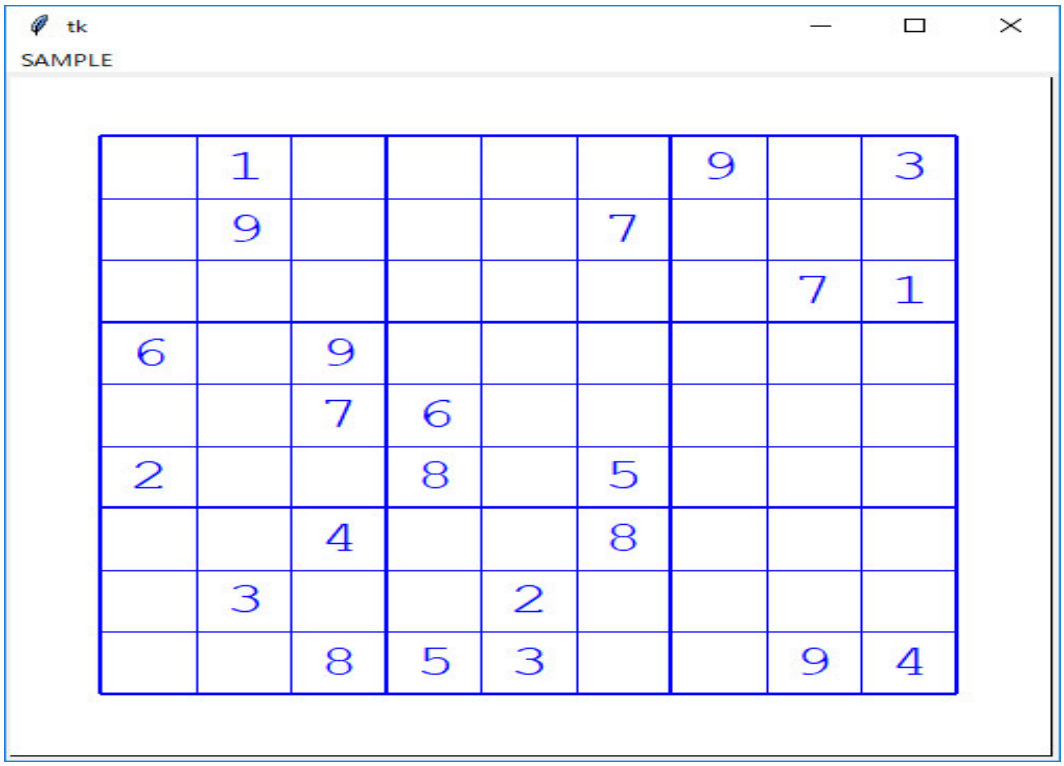
です。プログラムの全体は

```
from tkinter import *

def callback():
```

```
    print( 'called')

board_size = 500
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=callback)

def callback():
    print( 'called')

canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')

ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in range(9):
```

```
    for i in range(9):
        if ban[k][i] > 0:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                        font=f1, fill=c1)

root.mainloop()
```

となりました。実行すると



です。「SAMPLE」 というメニューができています。「SAMPLE」をクリックすると、「Sample1」というサブメニューを教示します。

```
menu_SAMPLE.add_command(label='Sample1', command=callback)
```

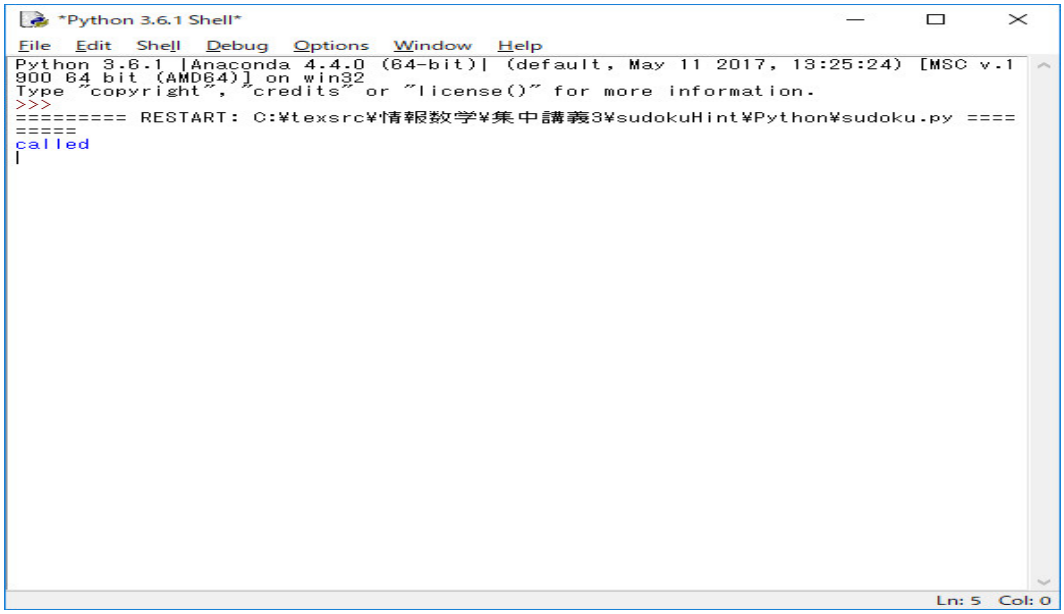
で、「Sample1」というサブメニューをクリックするとこの場合 callback() という関数が実行されます。 callback() という関数は

```
def callback():
    print( 'called')
```

と定義されていて、これは

```
menu_SAMPLE.add_command(label='Sample1', command=callback)
```

の前に書いておく必要があります。called と表示するだけです。




```
menu_SAMPLE.add_command(label='Sample1', command=callback)
```

を

```
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
```

と書き直して、関数 fun_sample1() を

```
ban = []
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    ShowBan()
```

と定義し、関数 ShowBan() を

```
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
```

```
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
```

と定義します。ここで、

```
ban = []
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    ShowBan()
```

の

```
    global ban
```

は、グローバル変数（関数の中だけでなく、どこからでもアクセスできる変数） ban の値を変更するので、このような宣言が必要です。

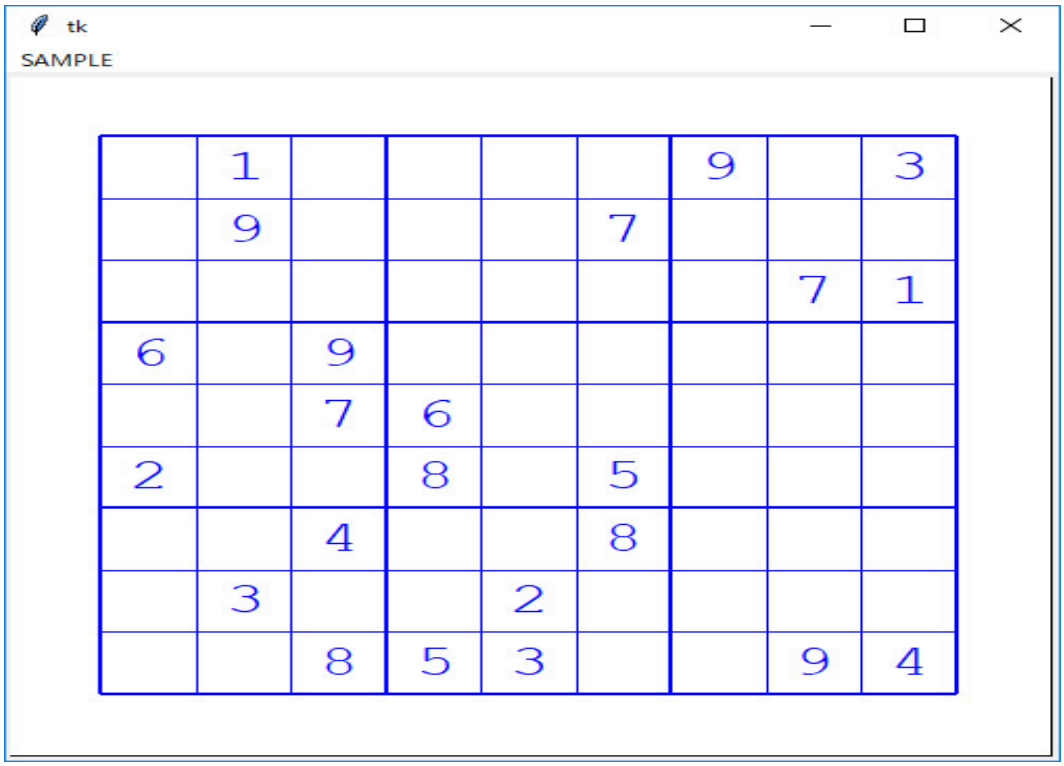
プログラムの全体は

```
from tkinter import *

board_size = 500
```

```
ban = []
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    ShowBan()
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
```

```
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)

root.mainloop()
```

となります。実行し、「SAMPLE」 メニューの「Sample1」サブメニューをクリックすると



となります。関数 fun_sample2() を

```
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    ShowBan()
```

と定義し、

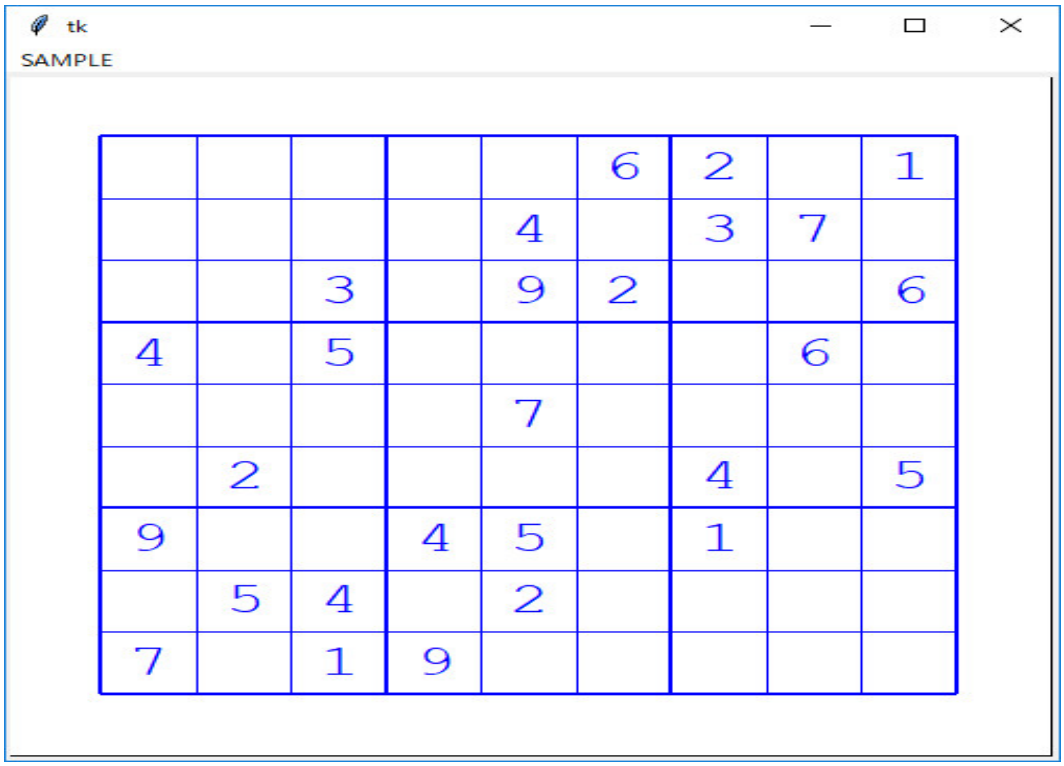
```
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
```

を追加すれば、プログラムの全体は

```
from tkinter import *

board_size = 500
ban = []
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    ShowBan()
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
```

```
            else:
                canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
        f1, f2 = "courier 24", "courier 12"
        c1, c2, c3 = "blue", "red", "black"
        for k in range(9):
            for i in range(9):
                if ban[k][i] > 0:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                       font=f1, fill=c1)


root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)

root.mainloop()
```

となります。実行し、「SAMPLE」 メニューの「Sample2」サブメニューをクリックすると



となります。これも私のプログラムで作った問題で、かなりの難問です。このようにして、問題の数を増やしていけばいいです。

次に、マウスで解を入力できるようにしましょう。

```
canvas.bind("<ButtonPress-1>", buttonPress)
```

で、マウスがクリックされたとき、関数 buttonPress() が実行されるようになります。関数 buttonPress() は

```
def buttonPress(event):
    global ans
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        ss = '123456789'
        result = sd.askinteger("数値入力", ss)
        ans[k][i] = int(result)
        ShowBan()
```

と定義します。更に、何か所か修正する必要があります。最初に

```
import tkinter.simpledialog as sd
import copy
```

と打ち込みます。 関数 fun_sample1() と fun_sample2() に

```
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
```

のように

```
    global ans
    ans = copy.deepcopy(ban)
```

をそれぞれ追加します。ここで、

```
    ans = copy.deepcopy(ban)
```

は、深いコピーと言われ、ans に ban をコピーして、セットします。こうすると、ban と ans は独立に値を持つようになります。浅いコピーと言われる

```
    ans = ban[:]
```

では、リストの要素だけのコピーで、リストの要素のリストは、ban と ans で共通になるので、単純なリストのコピーは浅いコピーで良いですが、リストのリストは深いコピーを使う必要があります。深いコピーを使う為には

```
import copy
```

のインポート文が必要になります。
関数 ShowBan() に

```
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
```

のように

```
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
```

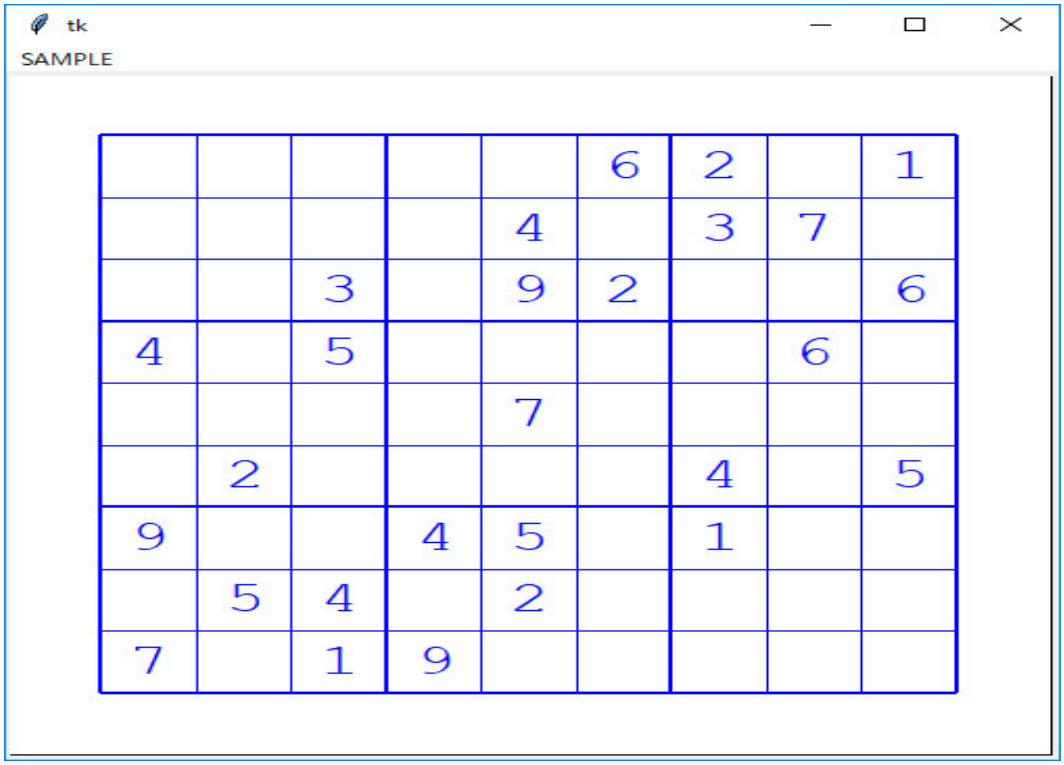
を追加します。プログラムの全体は

```
from tkinter import *
import tkinter.simpledialog as sd
import copy

board_size = 500
ban = []
ans = []
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
```

```
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
def buttonPress(event):
    global ans
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        ss = '123456789'
        result = sd.askinteger("数値入力", ss)
        ans[k][i] = int(result)
        ShowBan()
root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
```

```
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
canvas.bind("<ButtonPress-1>", buttonPress)

root.mainloop()
```

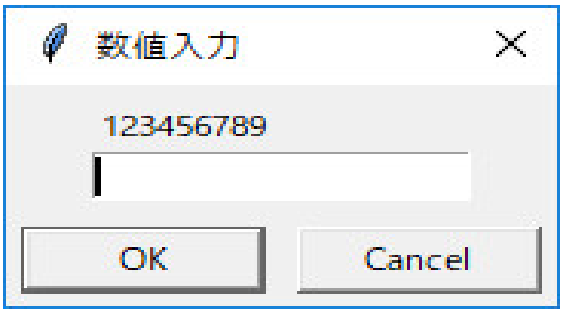
となります。実行し、「SAMPLE」 メニューの「Sample2」サブメニューをクリックすると



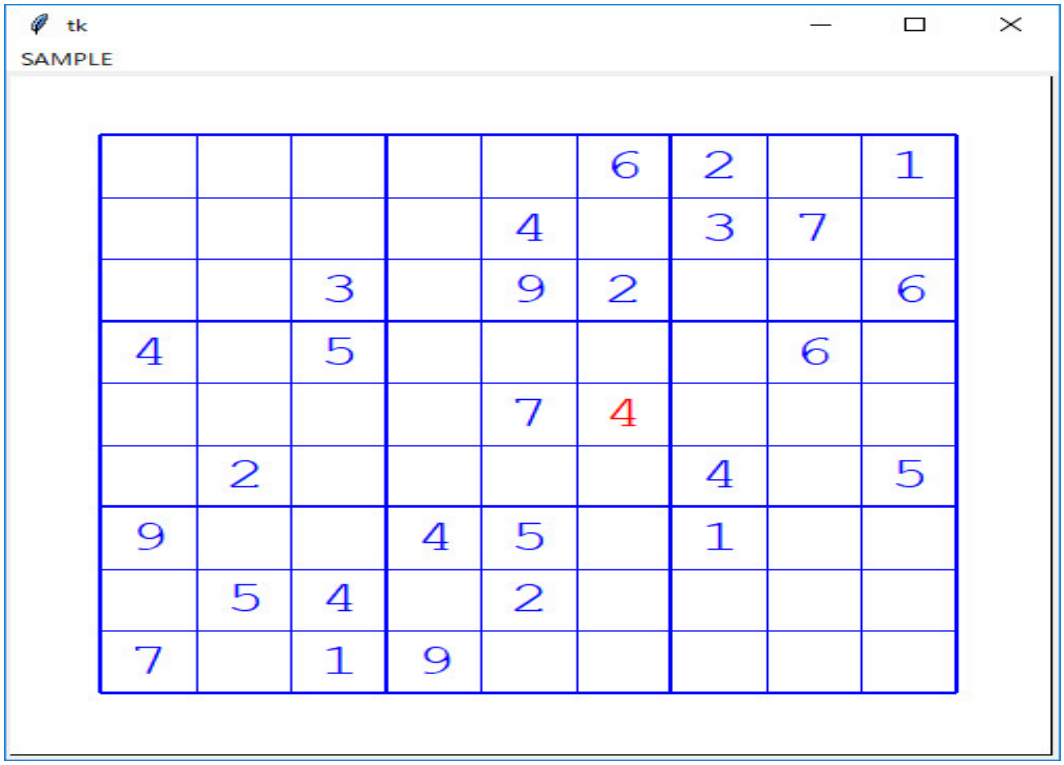
					6	2		1
				4		3	7	
		3		9	2			6
4		5					6	
				7				
	2					4		5
9			4	5		1		
	5	4		2				
7		1	9					

となります。私が C++ で作ったプログラムで調べると

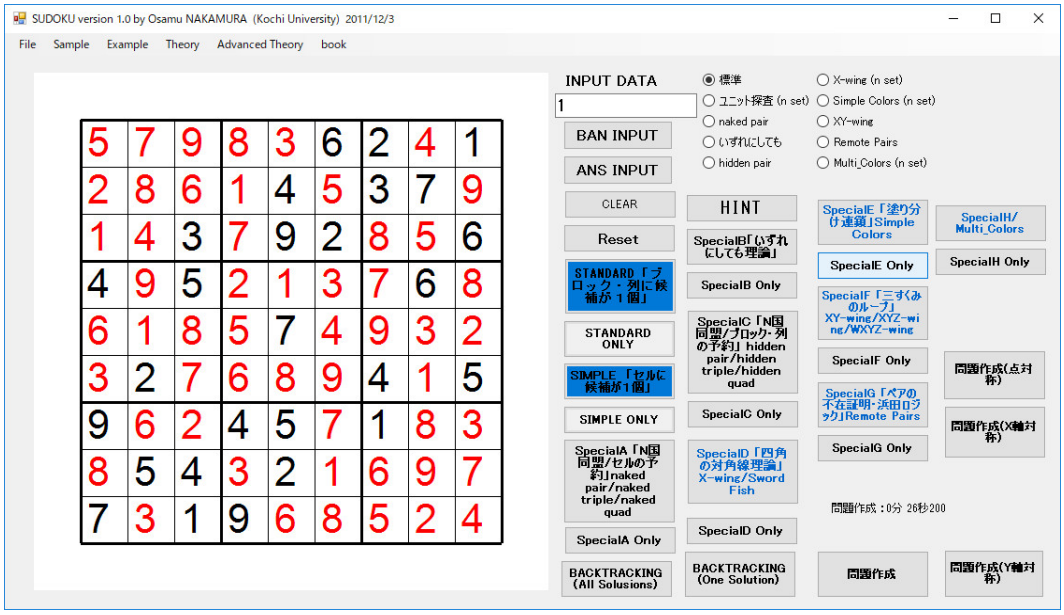
ですから、中央のブロックの７の右隣は４です。ここに答えを入れてみましょう。中央のブロックの７の右隣のセルをクリックします。



とダイアログボックスが開きます。4 を入力します。



となります。この問題の解は



です。「SpecialE Only」で解けます。

のようなヒントを表示するようにプログラムを修正しましょう。

```
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
```

```
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)
```

を追加し、 ShowBan() を

```
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
```

```
                ss2 = ss
                ss = ''
            ss += str(n)
        if ss1:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                font=f2, fill=c3)
        if ss2:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                font=f2, fill=c3)
        if ss1 and ss2 and ss:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                font=f2, fill=c3)
        elif ss1 and ss:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                               font=f2, fill=c3)
        else:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                font=f2, fill=c3)
```

と修正します。さらに、buttonPress() を

```
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        ans[k][i] = result
        ShowBan()
```

と修正します。ところで、

```
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
```

```
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)
```

において、cand はリストのリストで、要素は集合です。

```
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]
```

は

```
cand = [[set()]*9]*9
```

と同じだと思っていましたが、違うみたいです。また、

```
col = [[0]*9]*9
```

と

```
col = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
```

も、同じことではないみたいで、

```
col[k][i] = 1
```

のような代入をすると予想外のことが起こります。

数学で空集合を表す記号 {} は Python では、空の辞書と解釈されるので、空集合 {} は set() で表します。集合に要素を追加するには、集合のメソッド add() を使います。集合を空集合に変えるには、集合のメソッド clear() を使います。集合の和集合を作るには、2 項演算子 | を使います。集合の差集合を作るには、2 項演算子 - を使います。最後の

```
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)
```

で、最終的に、{1,2,3,,4,5,6,7,8,9} から、(k, i) を含む行、列、ブロックに現れる数字たちを除いた数字の集合が cand[k][i] にセットされています。

プログラム全体は

```
from tkinter import *
import tkinter.simpledialog as sd
import copy

board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]
```

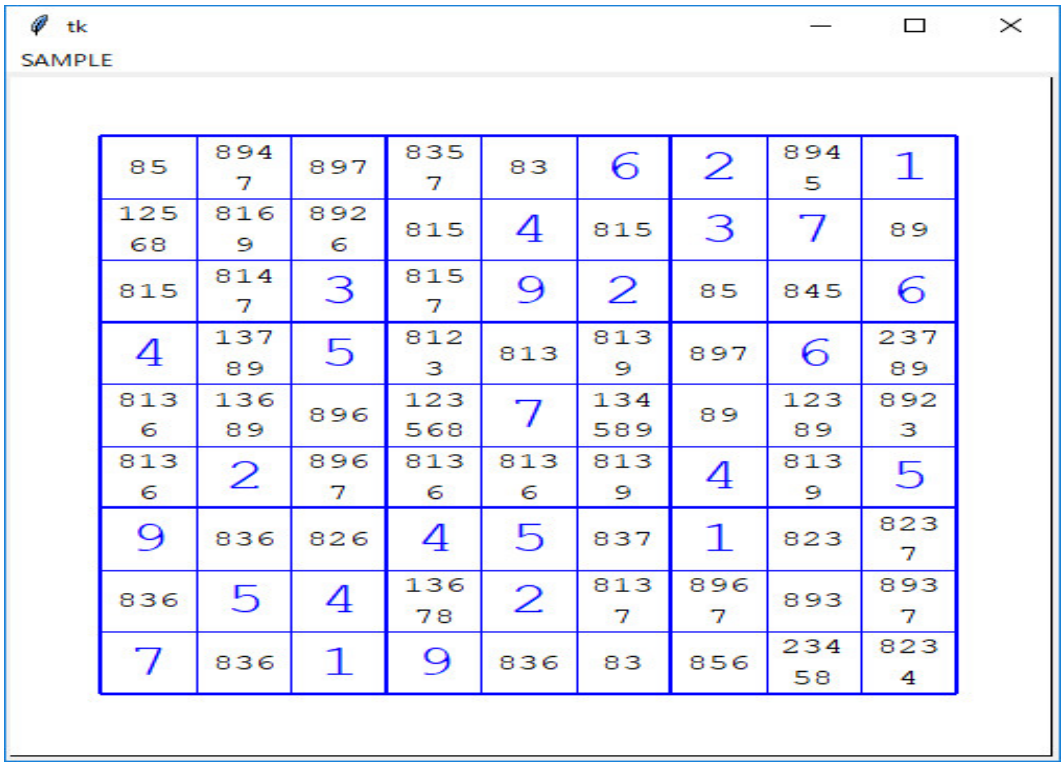
```
def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
```

```
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()

def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
```

```
                    ss1 = ss
                    ss =''
                elif cnt == 6:
                    ss2 = ss
                    ss = ''
                ss += str(n)
            if ss1:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                   font=f2, fill=c3)
            if ss2:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                   font=f2, fill=c3)
            if ss1 and ss2 and ss:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                   font=f2, fill=c3)
            elif ss1 and ss:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                  font=f2, fill=c3)
            else:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                   font=f2, fill=c3)
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        ans[k][i] = result
        ShowBan()


root = Tk()
canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
```

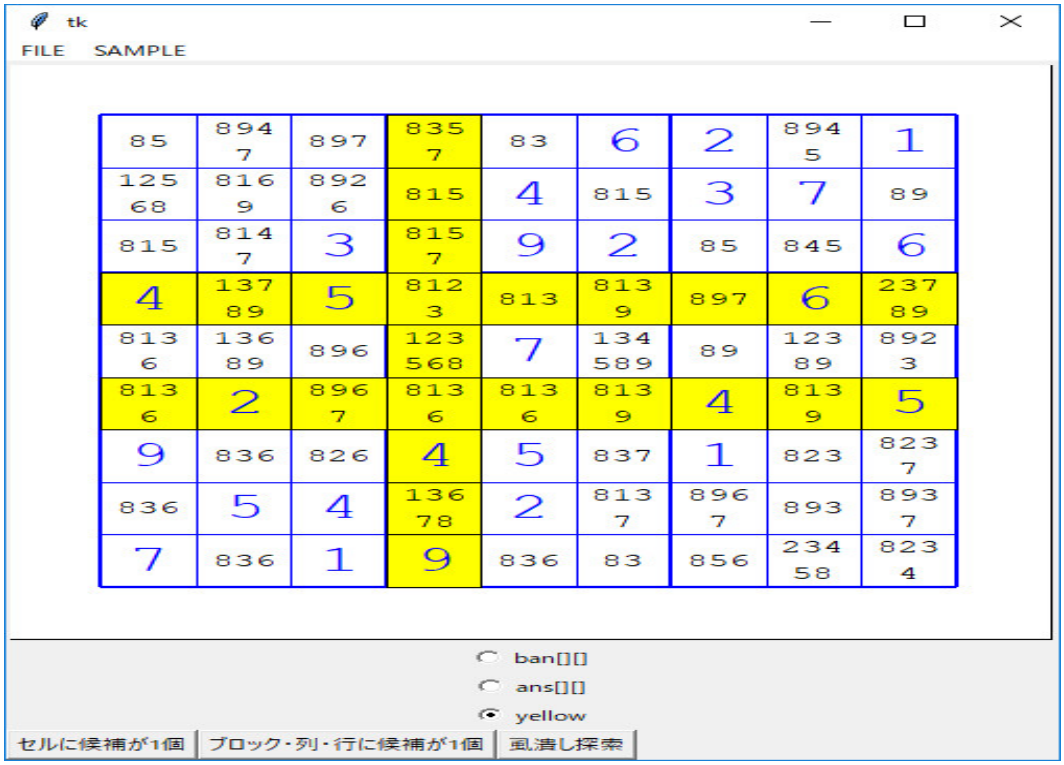
```
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
canvas.bind("<ButtonPress-1>", buttonPress)

root.mainloop()
```

となります。実行し、「SAMPLE」 メニューの「Sample2」サブメニューをクリックすると



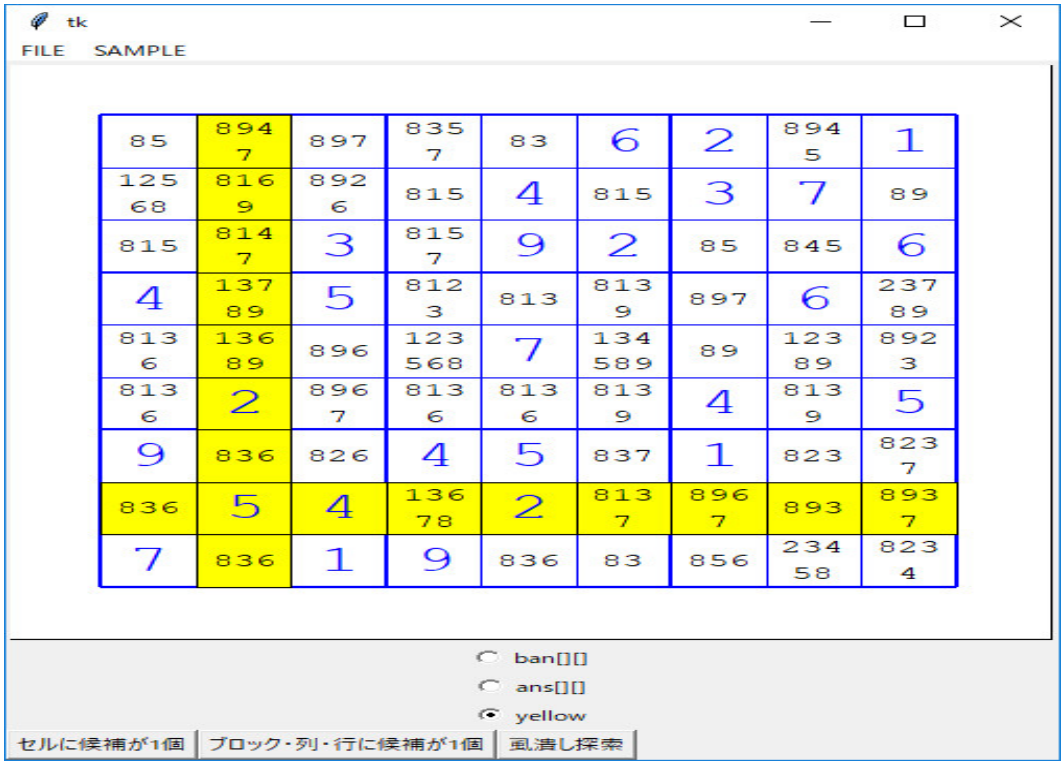
となります。

ヒントの小さい数字は、**1** から **9** までの数字のうち、縦の列、横の行、ブロックに現れる数字を除いた、そのセルに入りうる可能性のある数字を表しています。「セルに候補が **1** 個」の法則 **(One-choice: A cell that contains only one candidate value)** で、この数字が **1** 個ならその数字に確定します。今の場合、このようなセルはないです。しかし、中央のブロックの **7** の右側のセルに注目すると、この中央のブロックで、**4** が候補として挙げられているセルは **7** の右側のセルだけです。このことは、数独の解説書では、普通は **4** が置けないところに線を引いてみることをします。

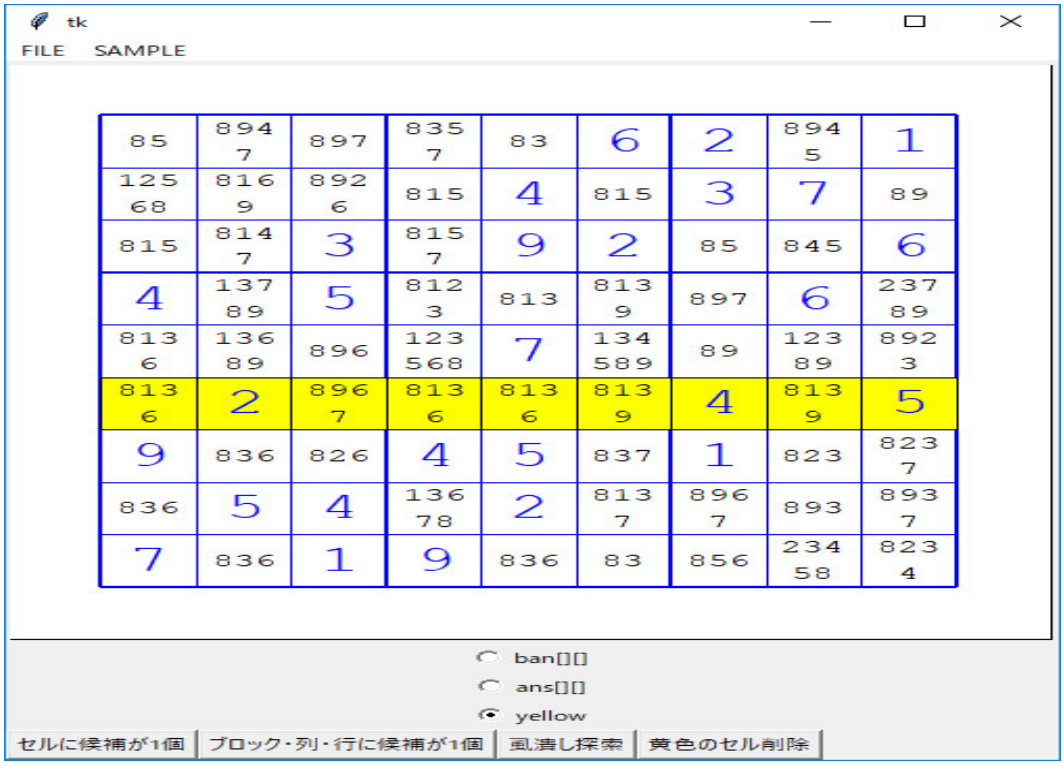


黄色の線上は 4 が置けなく、中央のブロックで４が置けるのは７の右隣だけです。

「ブロック・列・行に候補が 1 個」の法則 (**One-place: A region(row, column, or block) that has only one cell available for a given number**) で、このセルは４です。同じく、下段左のブロックの４の上のセルに注目すると、この行の小さい数字を見ると、２があるのは４の上のセルだけです。従って、このセルも「ブロック・列・行に候補が 1 個」の法則で２と確定します。

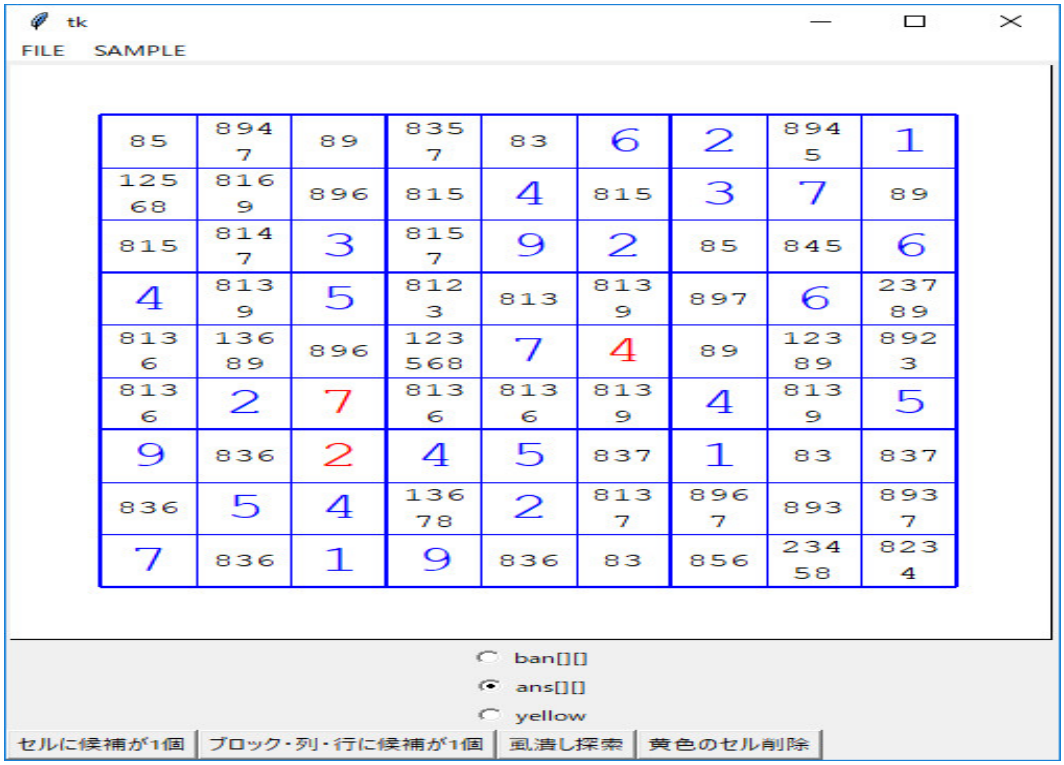


更に、6 行目に注目すると



２の右側のセルにはその行の他のセルの候補ではない７があるので、「ブロック・列・行に候補が 1 個」の法則で２の右側のセルは 7 であると確定します。

このようにして数字を確定していくとヒントの数字も変化し、続けて法則を適用できる場合もあります。



次に、問題を盤面上で入力できるようにします。まず、メニューで空っぽの盤を表示するものを作ります。

関数 fun_sample3() を

```
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
```

と定義し、

```
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)
```

を追加します。更に、ラジオボタンを２つ追加します。

```
r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()
```

ここが大事ですが

```
root = Tk()
val = IntVar()
val.set(0)
```

のように、

```
root = Tk()
```

の直後に

```
val = IntVar()
val.set(0)
```

を追加します。どうゆうわけか、ここでないと上手くいきません。最後に、関数 buttonPress()　を

```
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        ShowBan()
```

と修正します。プログラムの全体は

```
from tkinter import *
import tkinter.simpledialog as sd
import copy


board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def fun_sample1():
    global ban
    ban = [
```

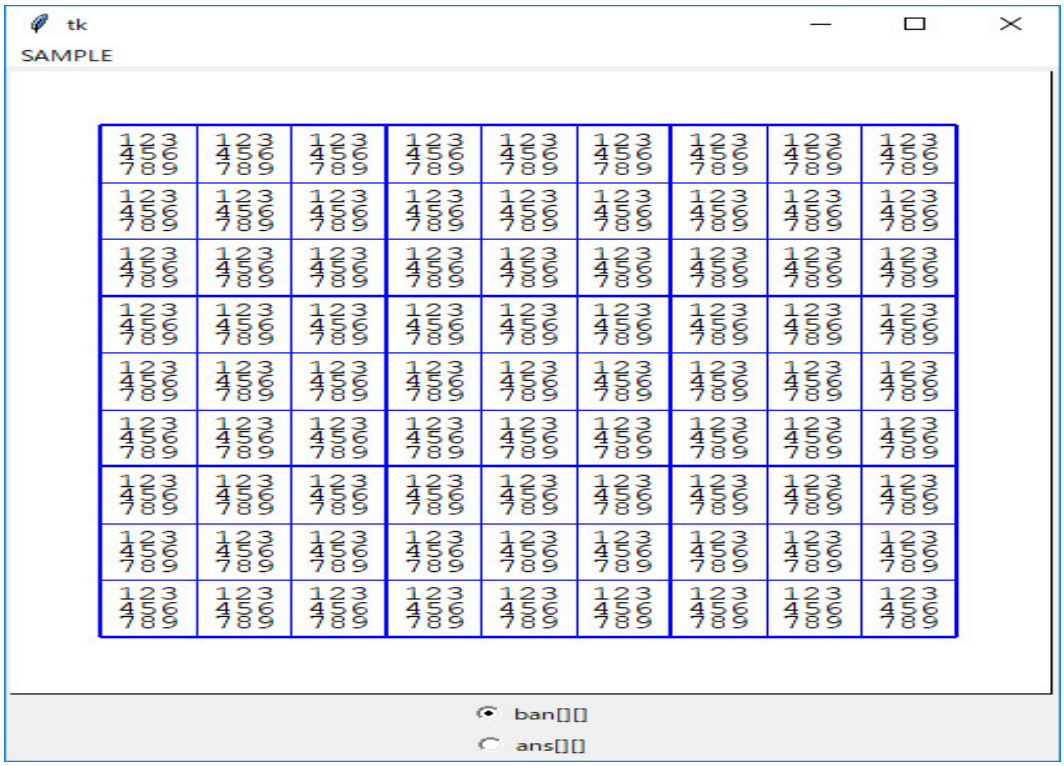
```
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
```

```
def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                       font=f2, fill=c3)
                if ss2:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
```

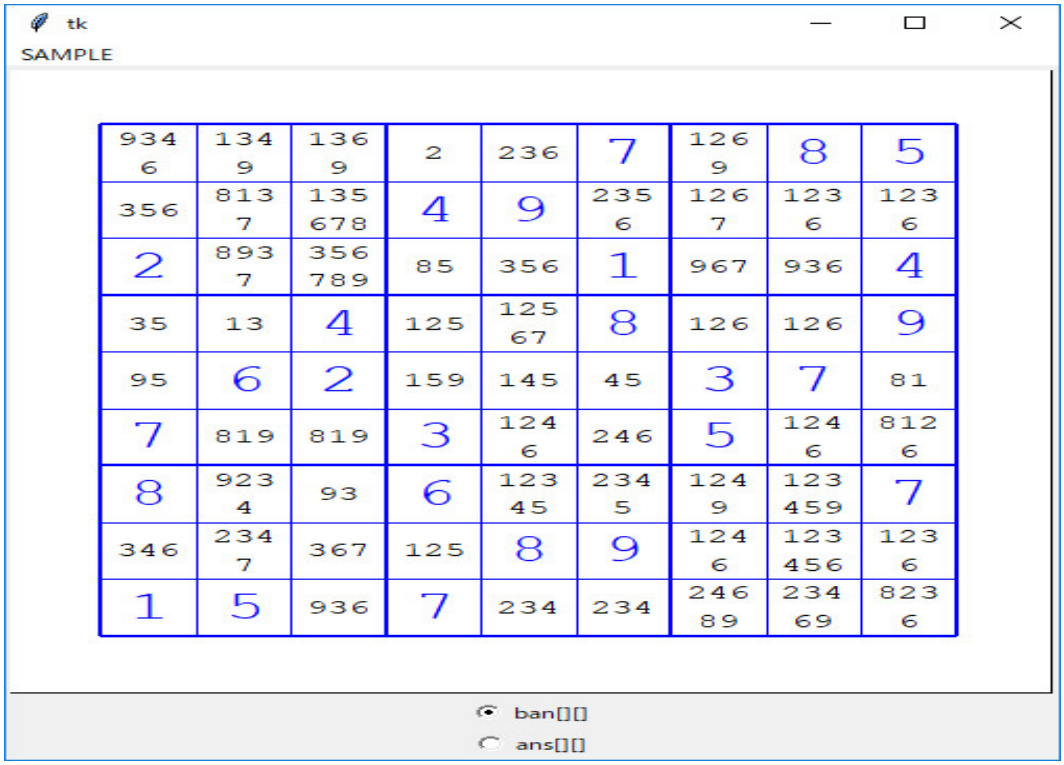
```
                                        font=f2, fill=c3)
                if ss1 and ss2 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                        font=f2, fill=c3)
                elif ss1 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                       font=f2, fill=c3)
                else:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                        font=f2, fill=c3)
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        ShowBan()


root = Tk()
val = IntVar()
val.set(0)

canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
```

```
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)

canvas.bind("<ButtonPress-1>", buttonPress)


r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()

root.mainloop()
```

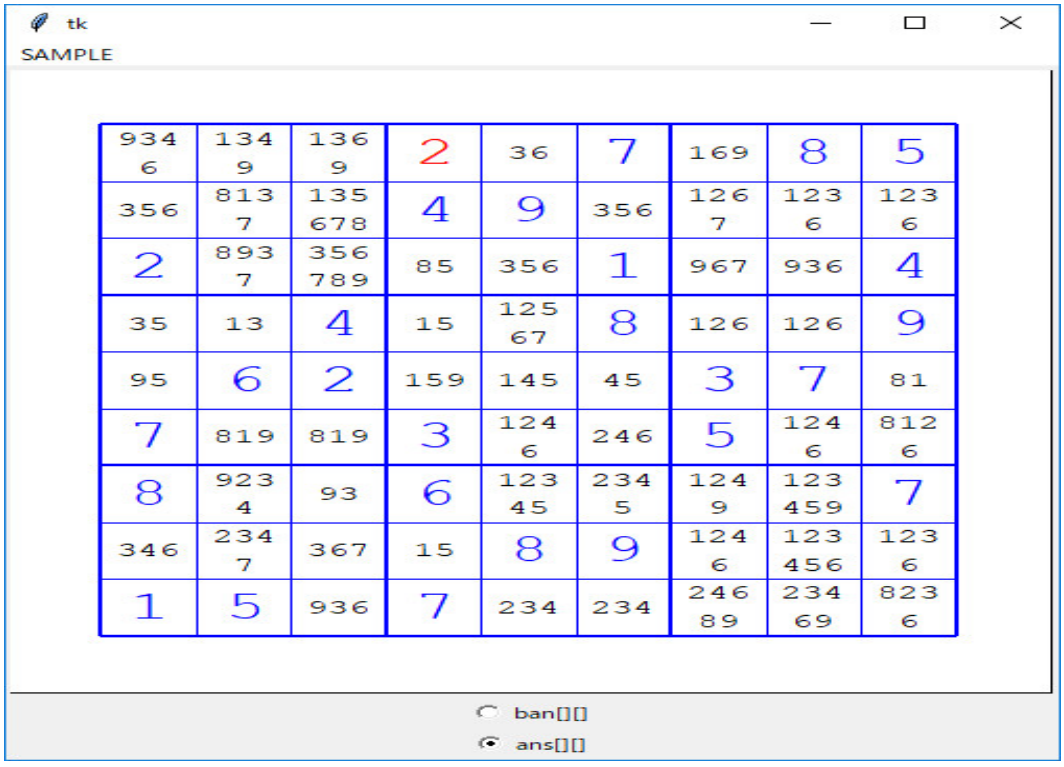
となります。実行し、「SAMPLE」 メニューの「Sample3」サブメニューをクリックすると



となります。セルをマウスでクリックして、問題を作ります。



は、私のプログラムが作った問題で、これは易しい問題です。ans[][] のラジオボタンをクリックすると解の数字が入力できます。上段中央のブロックの左上隅は候補が２だけですから、ここは２に確定します。入力してみましょう。



となります。
　次に、局面を保存できるようにしましょう。

```
import tkinter.filedialog as fd
import sys, os.path
```

をプログラムの先頭に追加します。

```
path_name = ""
file_name = ""
```

とメニュー

```
menu_FILE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="FILE", menu=menu_FILE)
menu_FILE.add_command(label='SaveAs', command=save_ban)
```

と関数 save_ban()

```
def save_ban():
    global path_name
    filename = fd.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        for x in ban:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        for x in ans:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        f.close()
```

を追加します。プログラムの全体は

```
from tkinter import *
import tkinter.simpledialog as sd
import copy
import tkinter.filedialog as fd
import sys, os.path
```

```
board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

path_name = ""
file_name = ""


def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
```

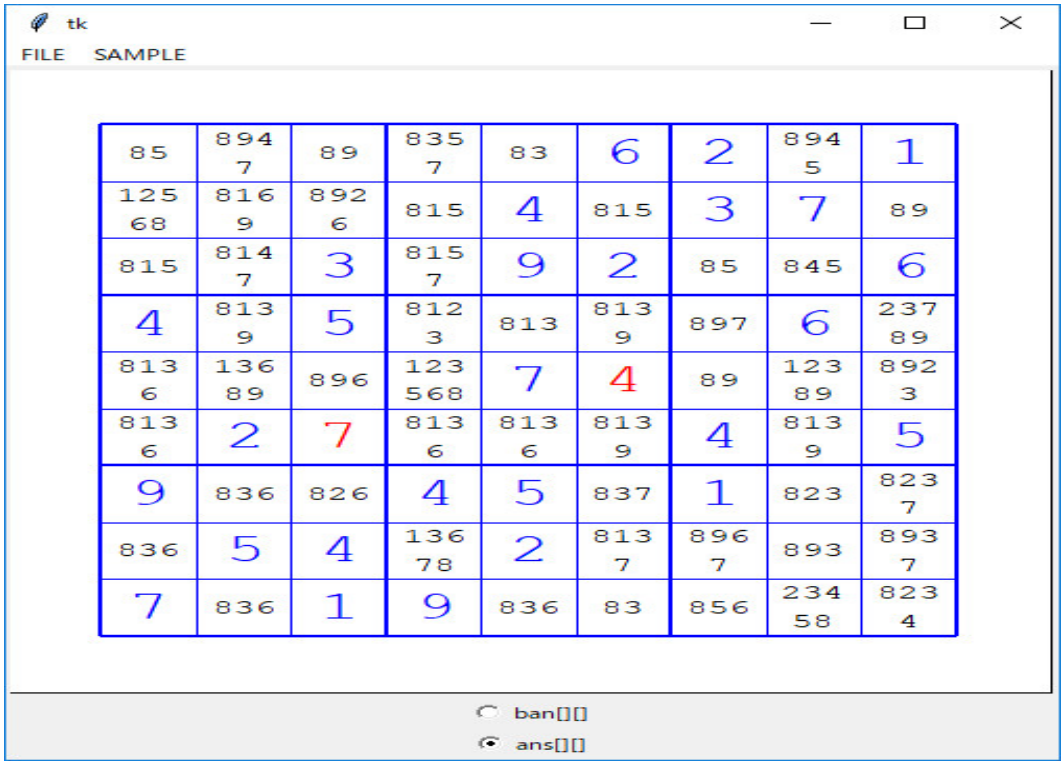
```
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()

def ShowBan():
```

```
canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
K = 9
s = board_size / (K+2)
w = h = (board_size - s * 9) / 2
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
for i in range(10):
    if i % 3 == 0:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
    else:
        canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in range(9):
    for i in range(9):
        if ban[k][i] > 0:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                font=f1, fill=c1)
        elif ans[k][i] > 0:
            canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                               font=f1, fill=c2)
        else:
            set_cand(k, i)
            ss = ''
            ss1 = ''
            ss2 = ''
            for cnt, n in enumerate(cand[k][i]):
                if cnt == 3:
                    ss1 = ss
                    ss =''
                elif cnt == 6:
                    ss2 = ss
                    ss = ''
                ss += str(n)
            if ss1:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                   font=f2, fill=c3)
            if ss2:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                   font=f2, fill=c3)
```

```
                if ss1 and ss2 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                       font=f2, fill=c3)
                elif ss1 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                      font=f2, fill=c3)
                else:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                       font=f2, fill=c3)
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        ShowBan()

def save_ban():
    global path_name
    filename = fd.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        for x in ban:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
```

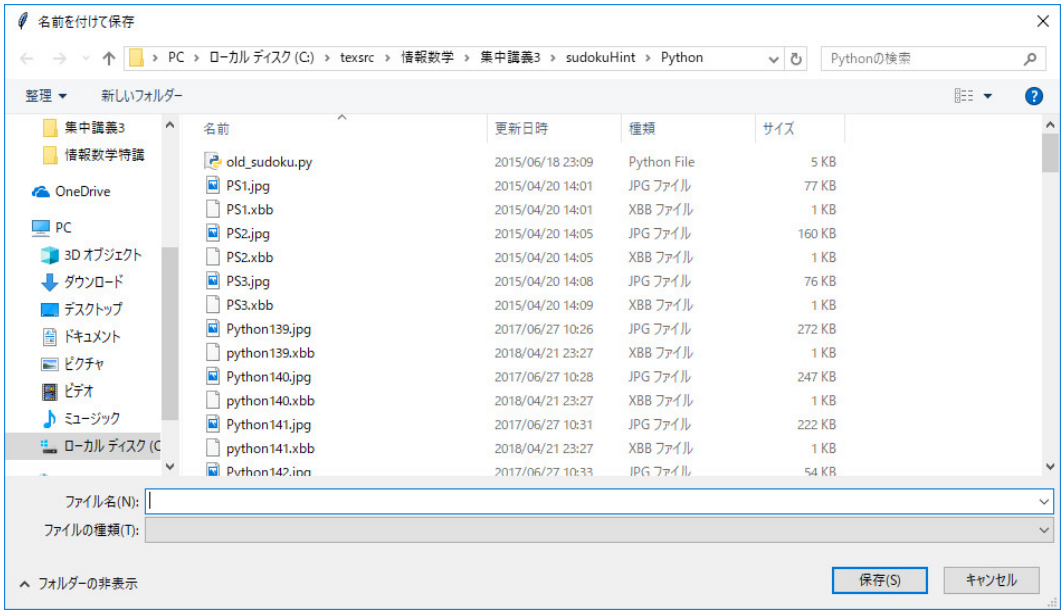
```
        for x in ans:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        f.close()


root = Tk()
val = IntVar()
val.set(0)

canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_FILE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="FILE", menu=menu_FILE)
menu_FILE.add_command(label='SaveAs', command=save_ban)

menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)

canvas.bind("<ButtonPress-1>", buttonPress)


r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()

root.mainloop()
```

となります。実行します。実行し、「SAMPLE」 メニューの「Sample2」サブメニューをクリックし、解の一部を入力し、

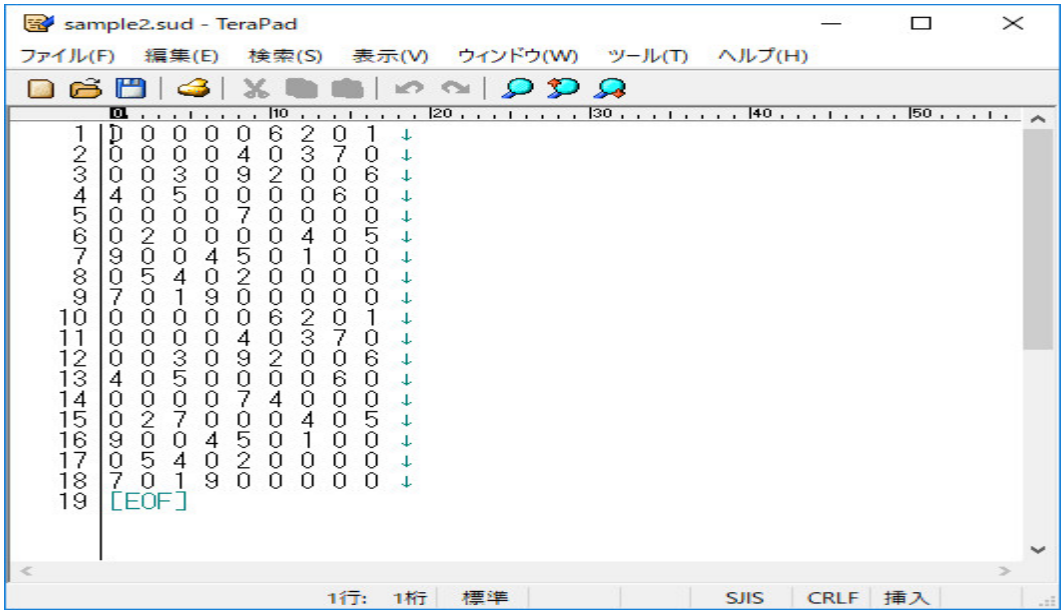


の局面を保存します。

「File」メニューの「SaveAs」サブメニューをクリックします。



となります。適当な名前を付けて、保存します。保存したデータは単に ban[][] と ans[][] のデータを並べたものです。



保存したデータを読み込むことが出来るようにしましょう。

`menu_FILE.add_command(label='Open', command=open_ban)`

を追加し、関数 `open_ban()`

```
def open_ban():
    global path_name, ban, ans
    ban = [[0]*9]*9
    ans = [[0]*9]*9
    filename = fd.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        for cnt in range(18):
            line = f.readline()
            s = line.split()
            if cnt < 9:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ban[cnt] = nlist
            else:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
```

```
                ans[cnt-9] = nlist
            cnt += 1
        f.close()
    ShowBan()
```

を追加します。全体のプログラムは

```
from tkinter import *
import tkinter.simpledialog as sd
import copy
import tkinter.filedialog as fd
import sys, os.path


board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

path_name = ""
file_name = ""


def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
```

```
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
```

```
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()

def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
```

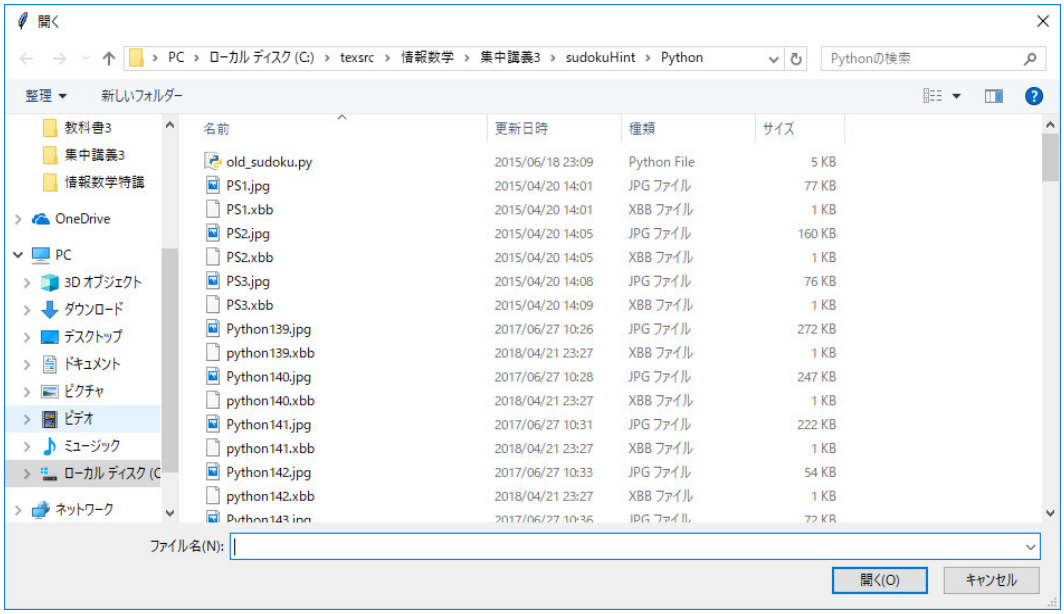
```
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                       font=f2, fill=c3)
                if ss2:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                       font=f2, fill=c3)
                if ss1 and ss2 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                       font=f2, fill=c3)
                elif ss1 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                      font=f2, fill=c3)
                else:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                       font=f2, fill=c3)
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        ShowBan()
```

```
def save_ban():
    global path_name
    filename = fd.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        for x in ban:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        for x in ans:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        f.close()
def open_ban():
    global path_name, ban, ans
    ban = [[0]*9]*9
    ans = [[0]*9]*9
    filename = fd.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        for cnt in range(18):
            line = f.readline()
            s = line.split()
            if cnt < 9:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ban[cnt] = nlist
            else:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ans[cnt-9] = nlist
```

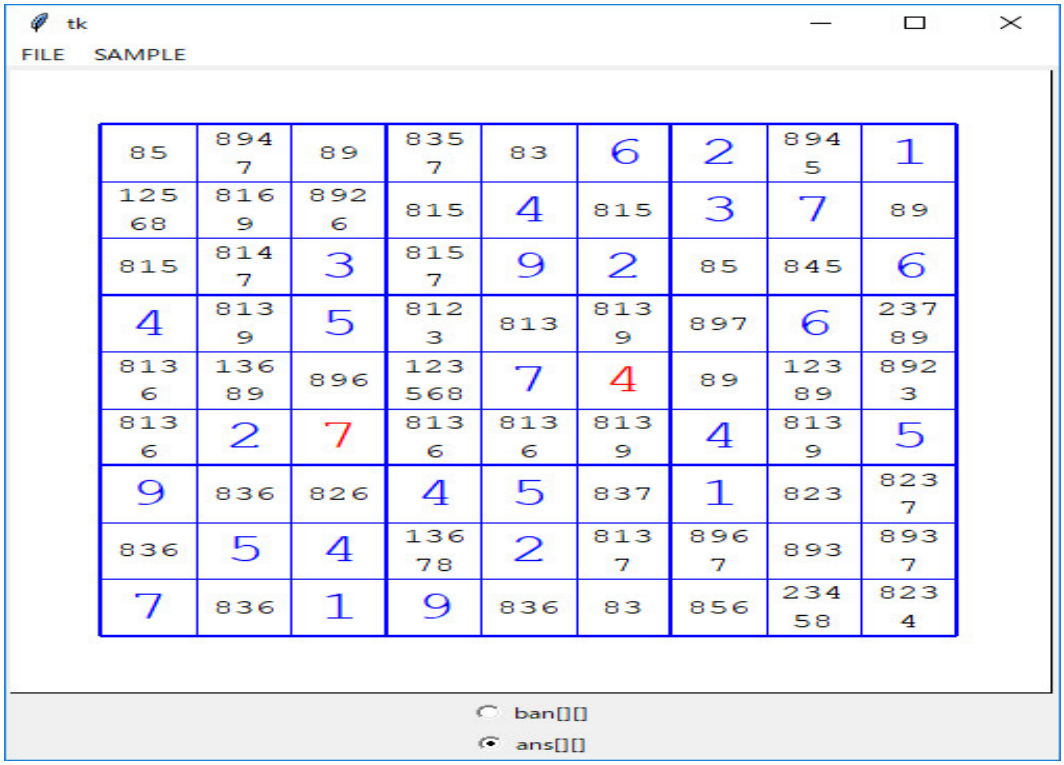
```
            cnt += 1
        f.close()
    ShowBan()


root = Tk()
val = IntVar()
val.set(0)

canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_FILE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="FILE", menu=menu_FILE)
menu_FILE.add_command(label='SaveAs', command=save_ban)
menu_FILE.add_command(label='Open', command=open_ban)

menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)

canvas.bind("<ButtonPress-1>", buttonPress)


r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()

root.mainloop()
```

となります。

実行し、「File」メニューの「Open」サブメニューをクリックすると



となります。保存したファイル名を入力します。保存した局面



が復元されました。

局面を印刷できるようにしてみましょう。かなりめんどくさいですし、文字に色を付ける方法が分からなかったので、フォントの大きさで、ban[][] の情報と ans[][] の情報を分けました。

```
import win32print
import win32con
```

```
import win32gui
import win32ui
```

をプログラムの先頭に置きます。メニューに

```
menu_FILE.add_separator()
menu_FILE.add_command(label='Print', command=print_ban)
```

を追加します。関数 print_ban() を

```
def print_ban():
    K = 9
    s = int(20000 / (K+2))
    w = h = int((20000 - s * 9) / 2)
    PRINTER_NAME = win32print.GetDefaultPrinter()
    hprinter = win32print.OpenPrinter(PRINTER_NAME)
    devmode = win32print.GetPrinter(hprinter, 9)["pDevMode"]
    if devmode == None:
        devmode = win32print.GetPrinter(hprinter, 8)["pDevMode"]
    devmode.PaperSize = win32con.DMPAPER_A4
    devmode.Fields |= win32con.DM_PAPERSIZE
    devmode.Orientation = win32con.DMORIENT_PORTRAIT # 縦
    devmode.Fields |= win32con.DM_ORIENTATION
    hdc = win32gui.CreateDC("WINSPOOL", PRINTER_NAME, devmode)
    dc = win32ui.CreateDCFromHandle(hdc)
    dc.SetMapMode(win32con.MM_HIMETRIC)
    dc.StartDoc("印刷ドキュメント")
    dc.StartPage()
    pen = win32ui.CreatePen(0, 5, 0x666666)
    pen2 = win32ui.CreatePen(0, 30, 0x666666)

    dc.SelectObject(pen)
    MM = 100

    for i in range(10):
        if i % 3 == 0:
            dc.SelectObject(pen2)
            dc.MoveTo((w, -(h+i*s)))
            dc.LineTo((w+9*s, -(h+i*s)))
            dc.SelectObject(pen)
        else:
            dc.MoveTo((w, -(h+i*s)))
            dc.LineTo((w+9*s, -(h+i*s)))
    for i in range(10):
        if i % 3 == 0:
```

```
            dc.SelectObject(pen2)
            dc.MoveTo((w+i*s, -h))
            dc.LineTo((w+i*s, -(h+9*s)))
            dc.SelectObject(pen)
        else:
            dc.MoveTo((w+i*s, -h))
            dc.LineTo((w+i*s, -(h+9*s)))

    PIXELS_PER_INCH = 1440 # 1 インチ毎のピクセル数
    INCH_PER_POINT = 72 # 1 インチ毎のポイント数
    SCALE_FACTOR = int(PIXELS_PER_INCH / INCH_PER_POINT) #  わざわざ計算せず 20 と
指定する場合が多い


    fontdict = {
        "height": SCALE_FACTOR * 60, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font = win32ui.CreateFont(fontdict) # CFont インスタンスを作成
    fontdict2 = {
        "height": SCALE_FACTOR * 40, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font2 = win32ui.CreateFont(fontdict2) # CFont インスタンスを作成
    fontdict3 = {
        "height": SCALE_FACTOR * 20, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font3 = win32ui.CreateFont(fontdict3) # CFont インスタンスを作成
    # フォントをセットすると今までセットされてたフォントを返してくれるので保持しとく。
    oldfont = dc.SelectObject(font)

    # フォントをセットすると今までセットされてたフォントを返してくれるので保持しとく。
    oldfont = dc.SelectObject(font)

    # 描画: http://msdn.microsoft.com/ja-jp/library/cc428775.aspx
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                dc.SelectObject(font)
```

```
                dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))

            elif ans[k][i] > 0:
                dc.SelectObject(font2)
                dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))
    dc.SelectObject(font3)
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.25)*s), ss1)
                if ss2:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss2)
                if ss1 and ss2 and ss:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
                elif ss1 and ss:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
                else:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss)
    # フォントを元に戻す
    dc.SelectObject(oldfont)

    dc.EndPage()
    dc.EndDoc()
    dc.DeleteDC()
```

と定義します。プログラムの全体は

```
from tkinter import *
import tkinter.simpledialog as sd
import copy
import tkinter.filedialog as fd
```

```
import sys, os.path
import win32print
import win32con
import win32gui
import win32ui


board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

path_name = ""
file_name = ""


def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
```

```
            cand[k][i].add(n)
        cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
```

```
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()

def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
```

```
                    ss += str(n)
                if ss1:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                       font=f2, fill=c3)
                if ss2:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                       font=f2, fill=c3)
                if ss1 and ss2 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                       font=f2, fill=c3)
                elif ss1 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                      font=f2, fill=c3)
                else:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                       font=f2, fill=c3)
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        ShowBan()

def save_ban():
    global path_name
    filename = fd.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
```

```
        f = open(filename, "w")
        for x in ban:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        for x in ans:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        f.close()
def open_ban():
    global path_name, ban, ans
    ban = [[0]*9]*9
    ans = [[0]*9]*9
    filename = fd.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        for cnt in range(18):
            line = f.readline()
            s = line.split()
            if cnt < 9:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ban[cnt] = nlist
            else:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ans[cnt-9] = nlist
            cnt += 1
        f.close()
    ShowBan()

def print_ban():
    K = 9
```

```
s = int(20000 / (K+2))
w = h = int((20000 - s * 9) / 2)
PRINTER_NAME = win32print.GetDefaultPrinter()
hprinter = win32print.OpenPrinter(PRINTER_NAME)
devmode = win32print.GetPrinter(hprinter, 9)["pDevMode"]
if devmode == None:
    devmode = win32print.GetPrinter(hprinter, 8)["pDevMode"]
devmode.PaperSize = win32con.DMPAPER_A4
devmode.Fields |= win32con.DM_PAPERSIZE
devmode.Orientation = win32con.DMORIENT_PORTRAIT # 縦
devmode.Fields |= win32con.DM_ORIENTATION
hdc = win32gui.CreateDC("WINSPOOL", PRINTER_NAME, devmode)
dc = win32ui.CreateDCFromHandle(hdc)
dc.SetMapMode(win32con.MM_HIMETRIC)
dc.StartDoc("印刷ドキュメント")
dc.StartPage()
pen = win32ui.CreatePen(0, 5, 0x666666)
pen2 = win32ui.CreatePen(0, 30, 0x666666)

dc.SelectObject(pen)
MM = 100

for i in range(10):
    if i % 3 == 0:
        dc.SelectObject(pen2)
        dc.MoveTo((w, -(h+i*s)))
        dc.LineTo((w+9*s, -(h+i*s)))
        dc.SelectObject(pen)
    else:
        dc.MoveTo((w, -(h+i*s)))
        dc.LineTo((w+9*s, -(h+i*s)))
for i in range(10):
    if i % 3 == 0:
        dc.SelectObject(pen2)
        dc.MoveTo((w+i*s, -h))
        dc.LineTo((w+i*s, -(h+9*s)))
        dc.SelectObject(pen)
    else:
        dc.MoveTo((w+i*s, -h))
        dc.LineTo((w+i*s, -(h+9*s)))

PIXELS_PER_INCH = 1440 # 1 インチ毎のピクセル数
INCH_PER_POINT = 72 # 1 インチ毎のポイント数
```

SCALE_FACTOR = int(PIXELS_PER_INCH / INCH_PER_POINT) # わざわざ計算せず 20 と指定する場合が多い

```
fontdict = {
    "height": SCALE_FACTOR * 60, # 10 ポイント
    "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
    "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
}
font = win32ui.CreateFont(fontdict) # CFont インスタンスを作成
fontdict2 = {
    "height": SCALE_FACTOR * 40, # 10 ポイント
    "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
    "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
}
font2 = win32ui.CreateFont(fontdict2) # CFont インスタンスを作成
fontdict3 = {
    "height": SCALE_FACTOR * 20, # 10 ポイント
    "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
    "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
}
font3 = win32ui.CreateFont(fontdict3) # CFont インスタンスを作成
# フォントをセットすると今までセットされてたフォントを返してくれるので保持しとく。
oldfont = dc.SelectObject(font)

# フォントをセットすると今までセットされてたフォントを返してくれるので保持しとく。
oldfont = dc.SelectObject(font)

# 描画: http://msdn.microsoft.com/ja-jp/library/cc428775.aspx
for k in range(9):
    for i in range(9):
        if ban[k][i] > 0:
            dc.SelectObject(font)
            dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))

        elif ans[k][i] > 0:
            dc.SelectObject(font2)
            dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))
dc.SelectObject(font3)
for k in range(9):
    for i in range(9):
        if ans[k][i] == 0:
            set_cand(k, i)
```

```
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.25)*s), ss1)
                if ss2:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss2)
                if ss1 and ss2 and ss:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
                elif ss1 and ss:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
                else:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss)
    # フォントを元に戻す
    dc.SelectObject(oldfont)

    dc.EndPage()
    dc.EndDoc()
    dc.DeleteDC()


root = Tk()
val = IntVar()
val.set(0)

canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_FILE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="FILE", menu=menu_FILE)
menu_FILE.add_command(label='SaveAs', command=save_ban)
menu_FILE.add_command(label='Open', command=open_ban)
menu_FILE.add_separator()
menu_FILE.add_command(label='Print', command=print_ban)
```

```
menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)

canvas.bind("<ButtonPress-1>", buttonPress)



r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()

root.mainloop()
```

となります。

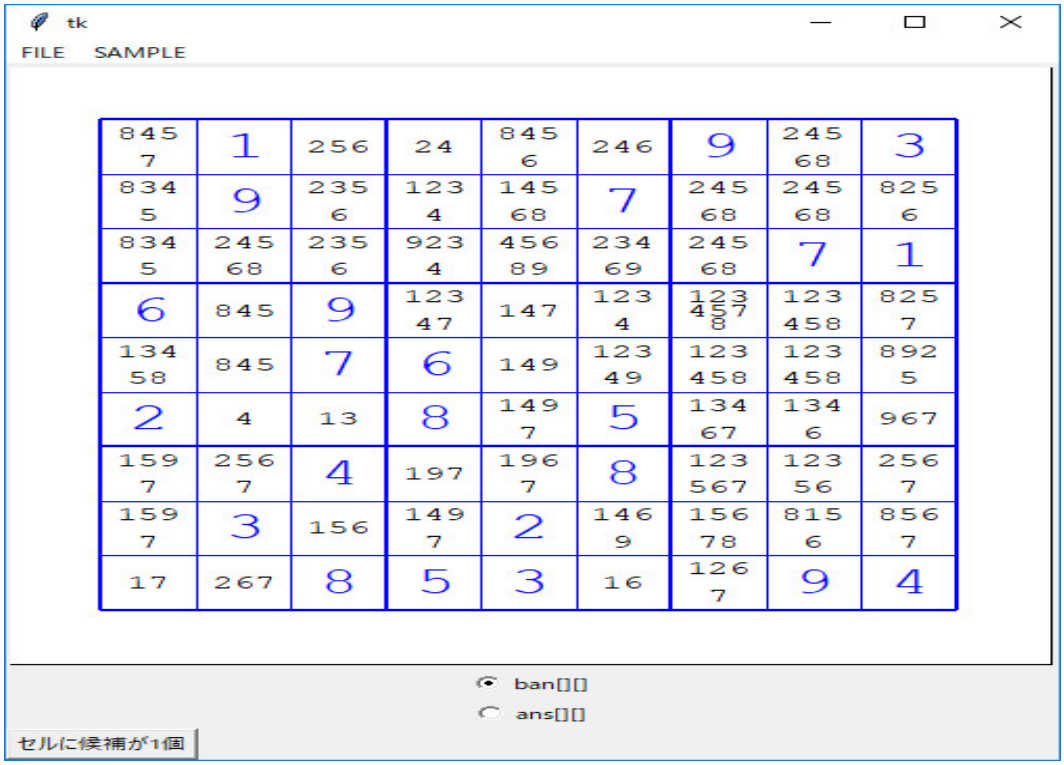
次に、途中で説明した「セルに候補が **1** 個」の法則をプログラミンして、コンピュータに数独を解かせるようにします。

```
button1 = Button(root, text='セルに候補が 1 個', command=button1_clicked)
button1.pack(side='left')
```

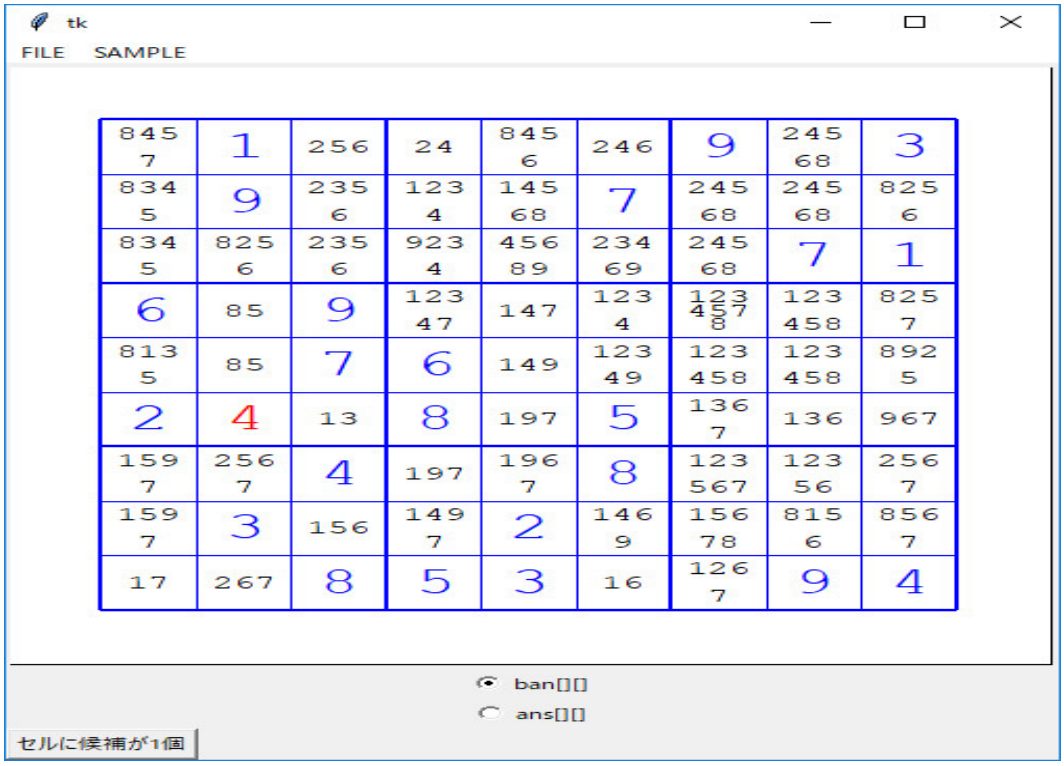
と関数 button1_clicked()

```
def button1_clicked():
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            set_cand(k, i)
            if len(cand[k][i]) == 1:
                ans[k][i] = cand[k][i].pop()
    ShowBan()
```

を追加します。実行し、「SAMPLE」 メニューの「Sample1」サブメニューをクリックすると



となります。「セルに候補が１個」のボタンをクリックすると



となります。

次に、途中で説明した「ブロック・列・行に候補が１個」の法則をプログラミンして、コンピュー

タに数独を解かせるようにします。

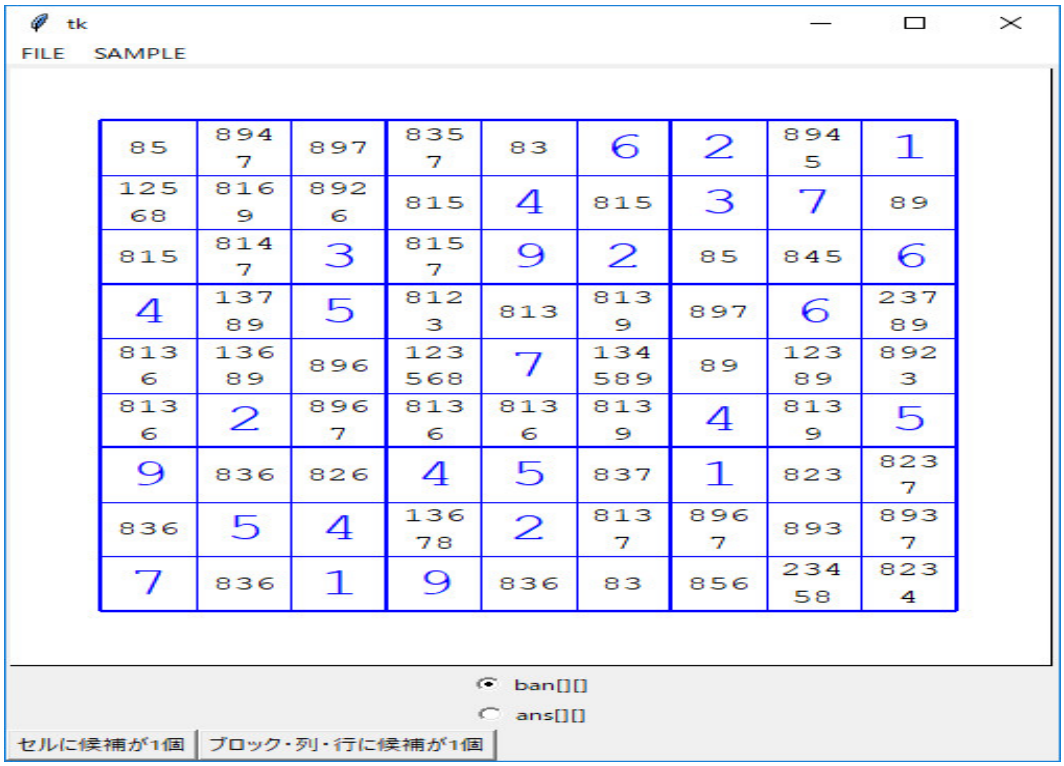
```
button2 = Button(root, text='ブロック･列･行に候補が1個', command=button2_clicked)
button2.pack(side='left')
```

と関数 button2_clicked()

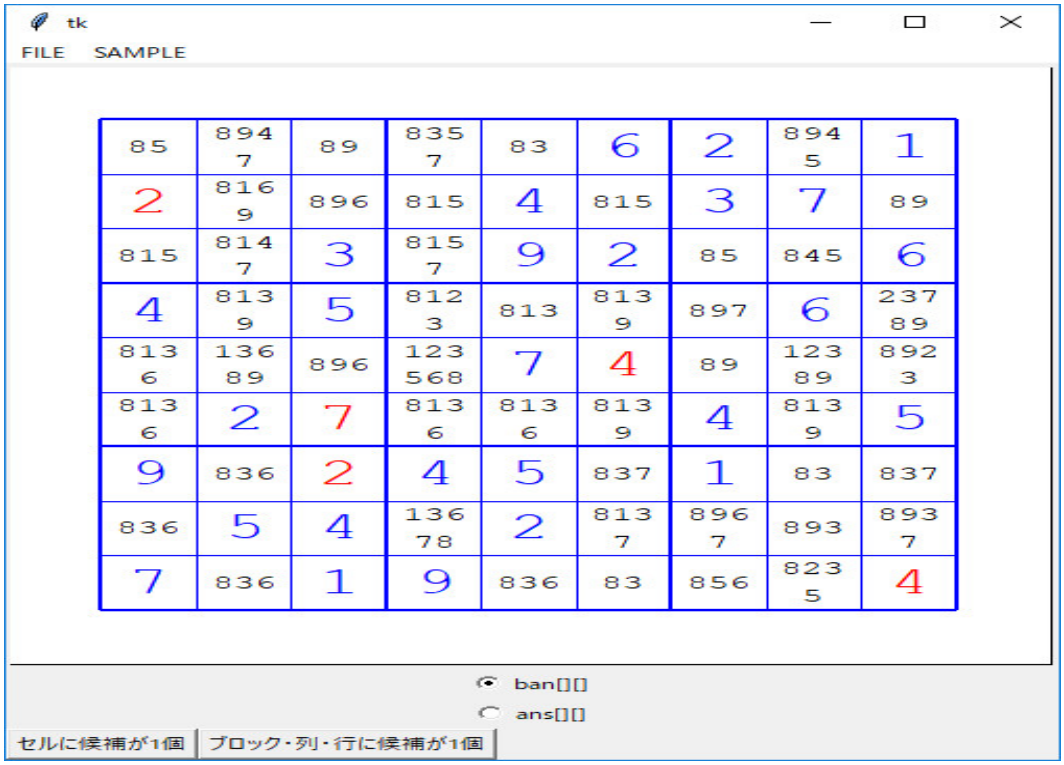
```
def button2_clicked():
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            ## 行のチェック
            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == i: continue
                    if ans[k][j] > 0: continue;
                    set_cand(k, j)
                    if n in cand[k][j]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
            ## 列のチェック
            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == k: continue
                    if ans[j][i] > 0: continue;
                    set_cand(j, i)
                    if n in cand[j][i]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
            ## ブロックのチェック
            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                kk = k // 3
```

```
                ii = i // 3
                for r in range(3*kk, 3*(kk+1)):
                    for c in range(3*ii, 3*(ii+1)):
                        if r == k and c == i: continue
                        if ans[r][c] > 0: continue
                        set_cand(r, c)
                        if n in cand[r][c]:
                            flag = True
                            break
                if not flag:
                    ans[k][i] = n
    ShowBan()
```

を追加します。実行し、「SAMPLE」 メニューの「Sample2」サブメニューをクリックすると



となります。「ブロック・列・行に候補が 1 個」のボタンをクリックすると



となります。

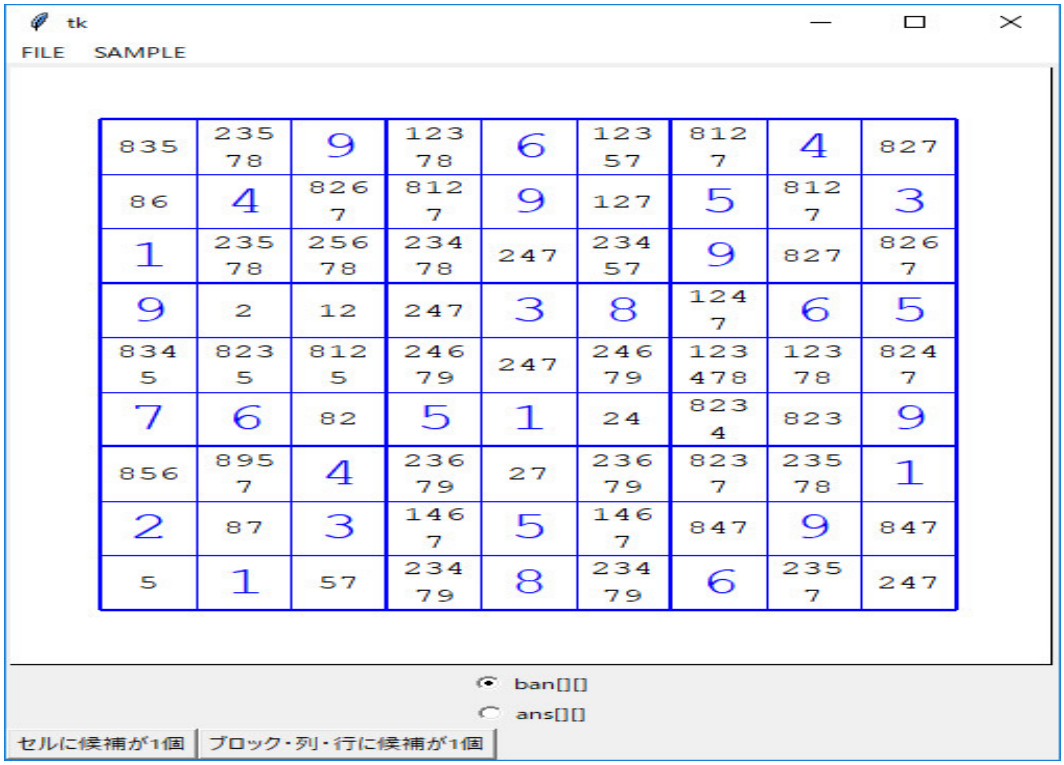
簡単な問題はこの２つの法則だけで解けます。

```
menu_SAMPLE.add_command(label='Sample4', command=fun_sample4)
```

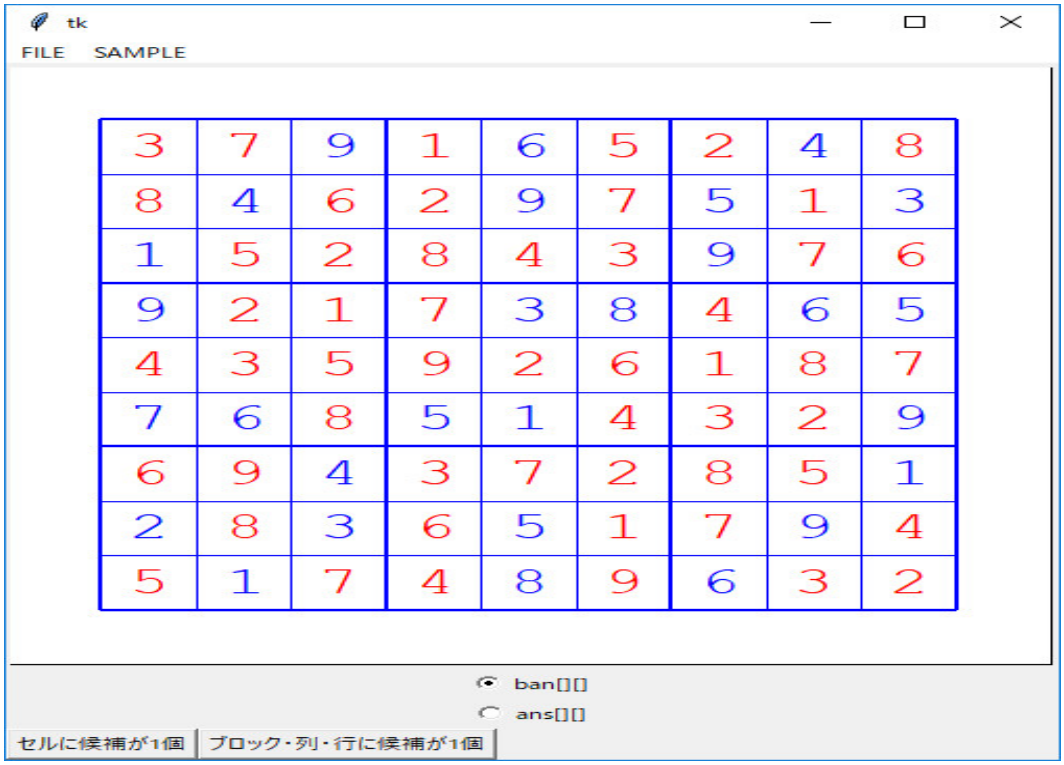
と関数 fun_sample4()

```
def fun_sample4():
    global ban
    ban = [
        [0,0,9,0,6,0,0,4,0],
        [0,4,0,0,9,0,5,0,3],
        [1,0,0,0,0,0,9,0,0],
        [9,0,0,0,3,8,0,6,5],
        [0,0,0,0,0,0,0,0,0],
        [7,6,0,5,1,0,0,0,9],
        [0,0,4,0,0,0,0,0,1],
        [2,0,3,0,5,0,0,9,0],
        [0,1,0,0,8,0,6,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
```

を追加します。実行し、「SAMPLE」 メニューの「Sample4」サブメニューをクリックすると



となります。「ブロック・列・行に候補が 1 個」のボタンを何回かクリックすると



となります。プログラムの全体は

```
from tkinter import *
```

```
import tkinter.simpledialog as sd
import copy
import tkinter.filedialog as fd
import sys, os.path
import win32print
import win32con
import win32gui
import win32ui


board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

path_name = ""
file_name = ""


def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
            if ans[r][c] > 0:
```

```
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
```

```
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
def fun_sample4():
    global ban
    ban = [
        [0,0,9,0,6,0,0,4,0],
        [0,4,0,0,9,0,5,0,3],
        [1,0,0,0,0,0,9,0,0],
        [9,0,0,0,3,8,0,6,5],
        [0,0,0,0,0,0,0,0,0],
        [7,6,0,5,1,0,0,0,9],
        [0,0,4,0,0,0,0,0,1],
        [2,0,3,0,5,0,0,9,0],
        [0,1,0,0,8,0,6,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()

def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
```

```
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                           font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                               font=f2, fill=c3)
                if ss2:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                               font=f2, fill=c3)
                if ss1 and ss2 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                               font=f2, fill=c3)
                elif ss1 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                              font=f2, fill=c3)
                else:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                               font=f2, fill=c3)
    def buttonPress(event):
        x = event.x
        y = event.y
        K = 9;
        s = board_size / (K+2);
        w = h = (board_size - s * 9) / 2;
        i = int((x - w) / s)
        k = int((y - h) / s)
        if i >= 0 and i < 9 and k >= 0 and k < 9:
```

```
        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        ShowBan()

def save_ban():
    global path_name
    filename = fd.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        for x in ban:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        for x in ans:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        f.close()
def open_ban():
    global path_name, ban, ans
    ban = [[0]*9]*9
    ans = [[0]*9]*9
    filename = fd.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        for cnt in range(18):
            line = f.readline()
            s = line.split()
```

```
            if cnt < 9:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ban[cnt] = nlist
            else:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ans[cnt-9] = nlist
            cnt += 1
        f.close()
    ShowBan()

def print_ban():
    K = 9
    s = int(20000 / (K+2))
    w = h = int((20000 - s * 9) / 2)
    PRINTER_NAME = win32print.GetDefaultPrinter()
    hprinter = win32print.OpenPrinter(PRINTER_NAME)
    devmode = win32print.GetPrinter(hprinter, 9)["pDevMode"]
    if devmode == None:
        devmode = win32print.GetPrinter(hprinter, 8)["pDevMode"]
    devmode.PaperSize = win32con.DMPAPER_A4
    devmode.Fields |= win32con.DM_PAPERSIZE
    devmode.Orientation = win32con.DMORIENT_PORTRAIT # 縦
    devmode.Fields |= win32con.DM_ORIENTATION
    hdc = win32gui.CreateDC("WINSPOOL", PRINTER_NAME, devmode)
    dc = win32ui.CreateDCFromHandle(hdc)
    dc.SetMapMode(win32con.MM_HIMETRIC)
    dc.StartDoc("印刷ドキュメント")
    dc.StartPage()
    pen = win32ui.CreatePen(0, 5, 0x666666)
    pen2 = win32ui.CreatePen(0, 30, 0x666666)

    dc.SelectObject(pen)
    MM = 100

    for i in range(10):
        if i % 3 == 0:
            dc.SelectObject(pen2)
```

```
            dc.MoveTo((w, -(h+i*s)))
            dc.LineTo((w+9*s, -(h+i*s)))
            dc.SelectObject(pen)
        else:
            dc.MoveTo((w, -(h+i*s)))
            dc.LineTo((w+9*s, -(h+i*s)))
    for i in range(10):
        if i % 3 == 0:
            dc.SelectObject(pen2)
            dc.MoveTo((w+i*s, -h))
            dc.LineTo((w+i*s, -(h+9*s)))
            dc.SelectObject(pen)
        else:
            dc.MoveTo((w+i*s, -h))
            dc.LineTo((w+i*s, -(h+9*s)))

    PIXELS_PER_INCH = 1440 # 1 インチ毎のピクセル数
    INCH_PER_POINT = 72 # 1 インチ毎のポイント数
    SCALE_FACTOR = int(PIXELS_PER_INCH / INCH_PER_POINT) #  わざわざ計算せず 20 と
指定する場合が多い


    fontdict = {
        "height": SCALE_FACTOR * 60, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font = win32ui.CreateFont(fontdict) # CFont インスタンスを作成
    fontdict2 = {
        "height": SCALE_FACTOR * 40, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font2 = win32ui.CreateFont(fontdict2) # CFont インスタンスを作成
    fontdict3 = {
        "height": SCALE_FACTOR * 20, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font3 = win32ui.CreateFont(fontdict3) # CFont インスタンスを作成

    # フォントをセットすると今までセットされてたフォントを返してくれるので保持しとく。
    oldfont = dc.SelectObject(font)
```

```
# 描画: http://msdn.microsoft.com/ja-jp/library/cc428775.aspx
for k in range(9):
    for i in range(9):
        if ban[k][i] > 0:
            dc.SelectObject(font)
            dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))

        elif ans[k][i] > 0:
            dc.SelectObject(font2)
            dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))
dc.SelectObject(font3)
for k in range(9):
    for i in range(9):
        if ans[k][i] == 0:
            set_cand(k, i)
            ss = ''
            ss1 = ''
            ss2 = ''
            for cnt, n in enumerate(cand[k][i]):
                if cnt == 3:
                    ss1 = ss
                    ss =''
                elif cnt == 6:
                    ss2 = ss
                    ss = ''
                ss += str(n)
            if ss1:
                dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.25)*s), ss1)
            if ss2:
                dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss2)
            if ss1 and ss2 and ss:
                dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
            elif ss1 and ss:
                dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
            else:
                dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss)

# フォントを元に戻す
dc.SelectObject(oldfont)

dc.EndPage()
dc.EndDoc()
```

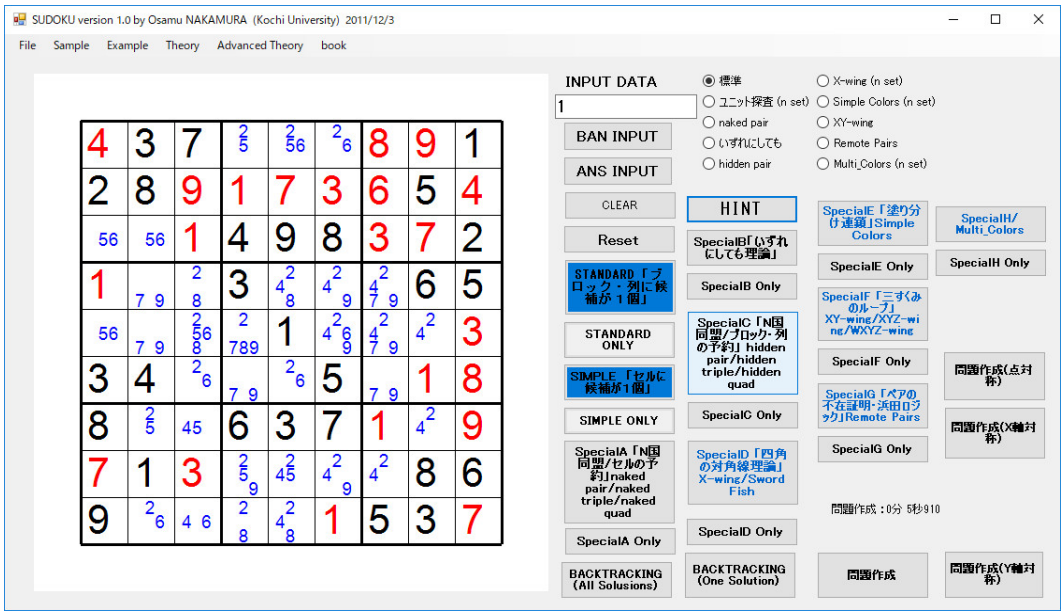
```
    dc.DeleteDC()

def button1_clicked():
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            set_cand(k, i)
            if len(cand[k][i]) == 1:
                ans[k][i] = cand[k][i].pop()
    ShowBan()

def button2_clicked():
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            ## 行のチェック
            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == i: continue
                    if ans[k][j] > 0: continue;
                    set_cand(k, j)
                    if n in cand[k][j]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
            ## 列のチェック
            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == k: continue
                    if ans[j][i] > 0: continue;
                    set_cand(j, i)
                    if n in cand[j][i]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
```

```
            ## ブロックのチェック
            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                kk = k // 3
                ii = i // 3
                for r in range(3*kk, 3*(kk+1)):
                    for c in range(3*ii, 3*(ii+1)):
                        if r == k and c == i: continue
                        if ans[r][c] > 0: continue
                        set_cand(r, c)
                        if n in cand[r][c]:
                            flag = True
                            break
                if not flag:
                    ans[k][i] = n
    ShowBan()

root = Tk()
val = IntVar()
val.set(0)

canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_FILE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="FILE", menu=menu_FILE)
menu_FILE.add_command(label='SaveAs', command=save_ban)
menu_FILE.add_command(label='Open', command=open_ban)
menu_FILE.add_separator()
menu_FILE.add_command(label='Print', command=print_ban)

menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)
menu_SAMPLE.add_command(label='Sample4', command=fun_sample4)

canvas.bind("<ButtonPress-1>", buttonPress)
```

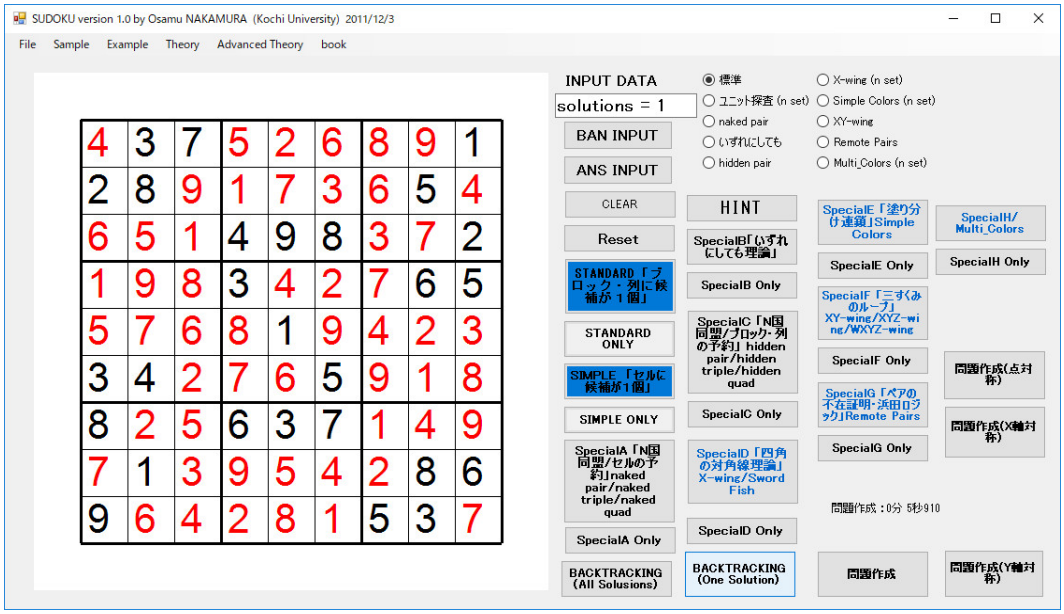
```
r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()

button1 = Button(root, text='セルに候補が１個', command=button1_clicked)
button1.pack(side='left')
button2 = Button(root, text='ブロック･列･行に候補が１個', command=button2_clicked)
button2.pack(side='left')

root.mainloop()
```

です。

書籍やインターネットを探せば、解法に関して、色々の情報が見つかります。後は自分でそれらの解法をプログラミングしていってください。問題をコンピュータに作らせるのも面白いです。私が C++ で作ったプログラムは色々な法則をプログラミングしていますが、それら全部を使っても、それらだけでは解けない問題があります。次の問題はそのような問題の１つで、私の C++ のプログラムが作った問題です。



しかし、バックトラッキングを使えば簡単に解けます。



が、その答えです。

虱潰し探索（バックトラッキング）をプログラミンして、コンピュータに数独を解かせるようにします。

まず、関数 SimplePlay() を

```
def SimplePlay():
    global ans, cand
    FLAG = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0:
                continue
            set_cand(k, i)
            if len(cand[k][i]) == 1:
                ans[k][i] = cand[k][i].pop()
                FLAG = True
    return FLAG
```

と定義し、関数 button1_clicked() を

```
def button1_clicked():
    SimplePlay()
    ShowBan()
```

と修正します。次に、関数 StandardPlay() を

```
def StandardPlay():
    global ans, cand
    FLAG = False
```

```
for k in range(9):
    for i in range(9):
        if ans[k][i] > 0: continue
        ## 行のチェック
        set_cand(k, i)
        S = cand[k][i].copy()
        for n in S:
            flag = False
            for j in range(9):
                if j == i: continue
                if ans[k][j] > 0: continue;
                set_cand(k, j)
                if n in cand[k][j]:
                    flag = True
                    break
            if not flag:
                ans[k][i] = n
                FLAG = True
        ## 列のチェック
        set_cand(k, i)
        S = cand[k][i].copy()
        for n in S:
            flag = False
            for j in range(9):
                if j == k: continue
                if ans[j][i] > 0: continue;
                set_cand(j, i)
                if n in cand[j][i]:
                    flag = True
                    break
            if not flag:
                ans[k][i] = n
                FLAG = True
        ## ブロックのチェック
        set_cand(k, i)
        S = cand[k][i].copy()
        for n in S:
            flag = False
            kk = k // 3
            ii = i // 3
            for r in range(3*kk, 3*(kk+1)):
                for c in range(3*ii, 3*(ii+1)):
                    if r == k and c == i: continue
```

```
                        if ans[r][c] > 0: continue
                        set_cand(r, c)
                        if n in cand[r][c]:
                            flag = True
                            break
                if not flag:
                    ans[k][i] = n
                    FLAG = True
    return FLAG
```

と定義し、関数 button2_clicked() を

```
def button2_clicked():
    StandardPlay()
    ShowBan()
```

と修正します。この修正でも今まで同様動きます。新たに、関数 completeP(), losingP() を

```
def completeP():
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0: return False
    for k in range(9):
        for n in range(1, 10):
            flag = False
            for i in range(9):
                if ans[k][i] == n:
                    flag = True
                    break
            if not flag:
                return False
    for i in range(9):
        for n in range(1, 10):
            flag = False
            for k in range(9):
                if ans[k][i] == n:
                    flag = True
                    break
            if not flag:
                return False
    for kk in range(3):
        for ii in range(3):
            for n in range(1, 10):
                flag = False
                for k in range(3*kk, 3*(kk+1)):
```

```
                    for i in range(3*ii, 3*(ii+1)):
                        if ans[k][i] == n:
                            flag = True
                            break
                if not flag:
                    return False
    return True

def losingP():
    global cand
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
                set_cand(k, i)
                if len(cand[k][i]) == 0:
                    return True
    for k in range(9):
        P = [0]*10
        for i in range(9):
            P[ans[k][i]] = P[ans[k][i]] + 1
        for n in range(1, 10):
            if P[n] > 1:
                return True
    for i in range(9):
        P = [0]*10
        for k in range(9):
            P[ans[k][i]] = P[ans[k][i]] + 1
        for n in range(1, 10):
            if P[n] > 1:
                return True
    for kk in range(3):
        for ii in range(3):
            P = [0]*10
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    P[ans[k][i]] = P[ans[k][i]] + 1
            for n in range(1, 10):
                if P[n] > 1:
                    return True
    return False
```

と定義します。これらの関数を使って、関数 solver() を

```
    global ans, cand
```

```
    if losingP():
        return False
    if completeP():
         return True
    flag = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
                kk = k
                ii = i
                flag = True
                break
        if flag:
            break
    set_cand(kk, ii)
    S = cand[kk][ii].copy()
    for n in S:
        ans[kk][ii] = n
        if solver():
            return True
        ans[kk][ii] = 0
    return False
```

と定義します。そして、ボタンを追加します。

```
button3 = Button(root, text=' 虱潰し探索', command=button3_clicked)
button3.pack(side='left')
```

最後に、関数 button3_clicked() を

```
def button3_clicked():
    global ans, cand
    solver()
    ShowBan()
```

と定義します。このプログラムは解が必ず一つあることを前提にしています。上の問題が解けることを確かめてみましょう。

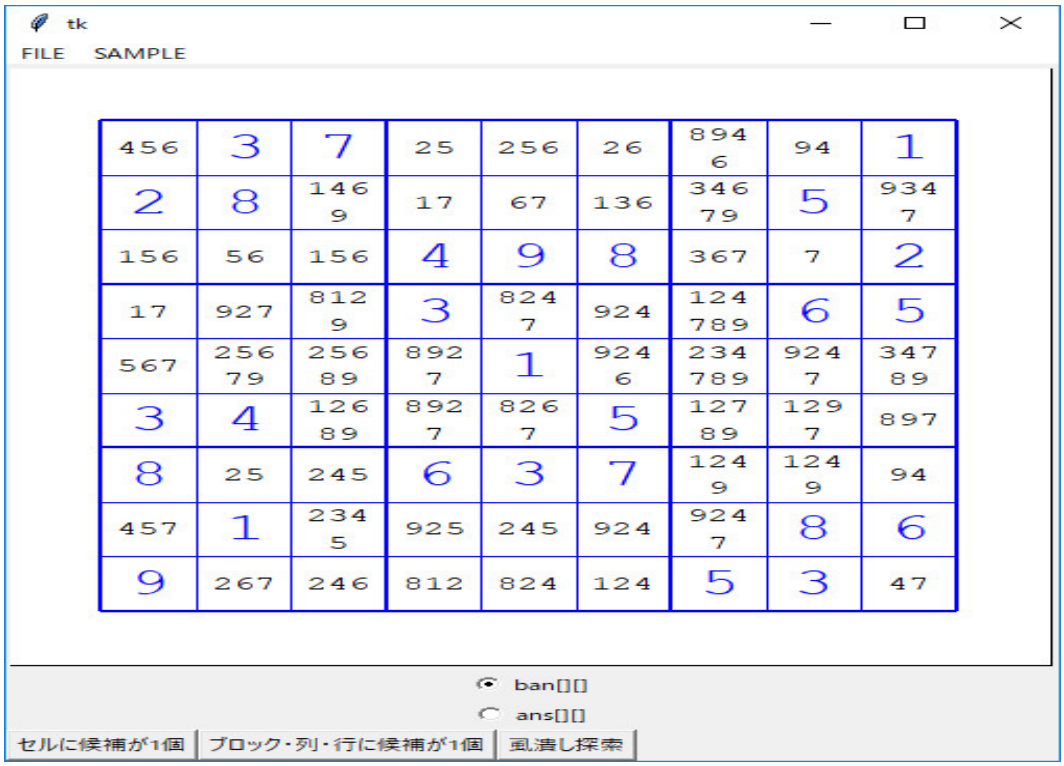
```
menu_SAMPLE.add_command(label='Sample5', command=fun_sample5)
```

と関数 fun_sample5() を

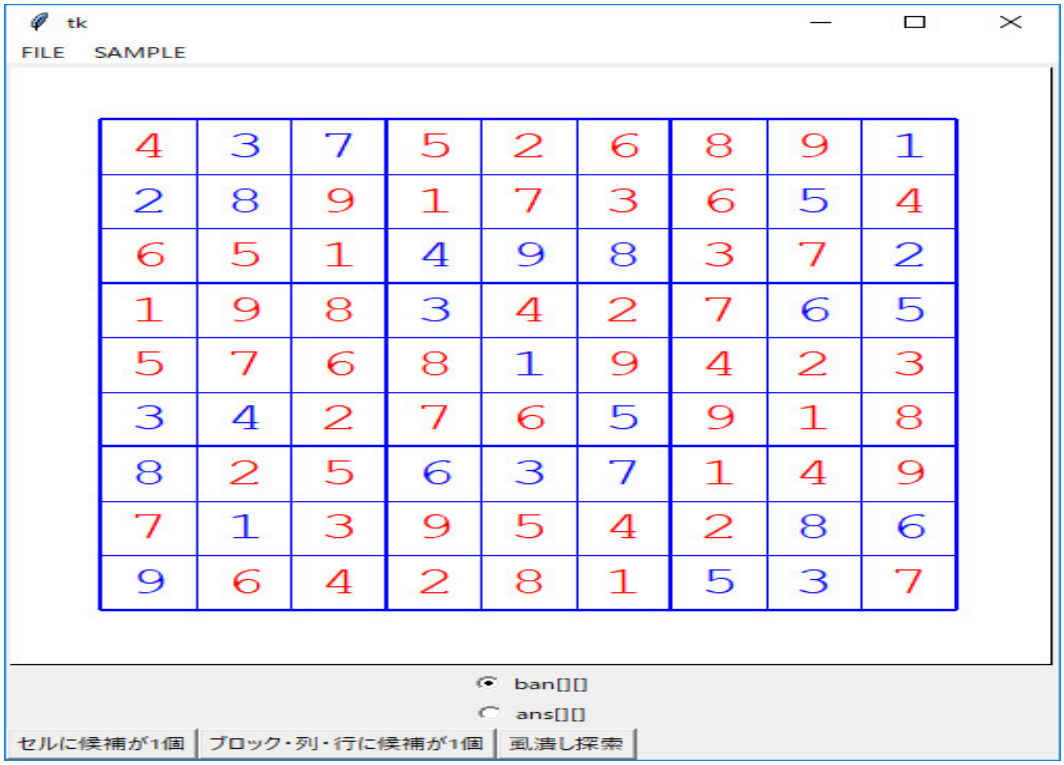
```
def fun_sample5():
    global ban
    ban = [
        [0,3,7,0,0,0,0,0,1],
        [2,8,0,0,0,0,0,5,0],
```

```
        [0,0,0,4,9,8,0,0,2],
        [0,0,0,3,0,0,0,6,5],
        [0,0,0,0,1,0,0,0,0],
        [3,4,0,0,0,5,0,0,0],
        [8,0,0,6,3,7,0,0,0],
        [0,1,0,0,0,0,0,8,6],
        [9,0,0,0,0,0,5,3,0]]
    global ans
    ans = copy.deepcopy(ban)
    ShowBan()
```

と定義します。実行し、「SAMPLE」 メニューの「Sample5」サブメニューをクリックすると



となります。「虱潰し探索」のボタンをクリックすると



となります。これで、時間はかかりますがどんな問題でも解けるようになりました。

もう少し高速に解けるようにしましょう。関数 RegularPlay() を

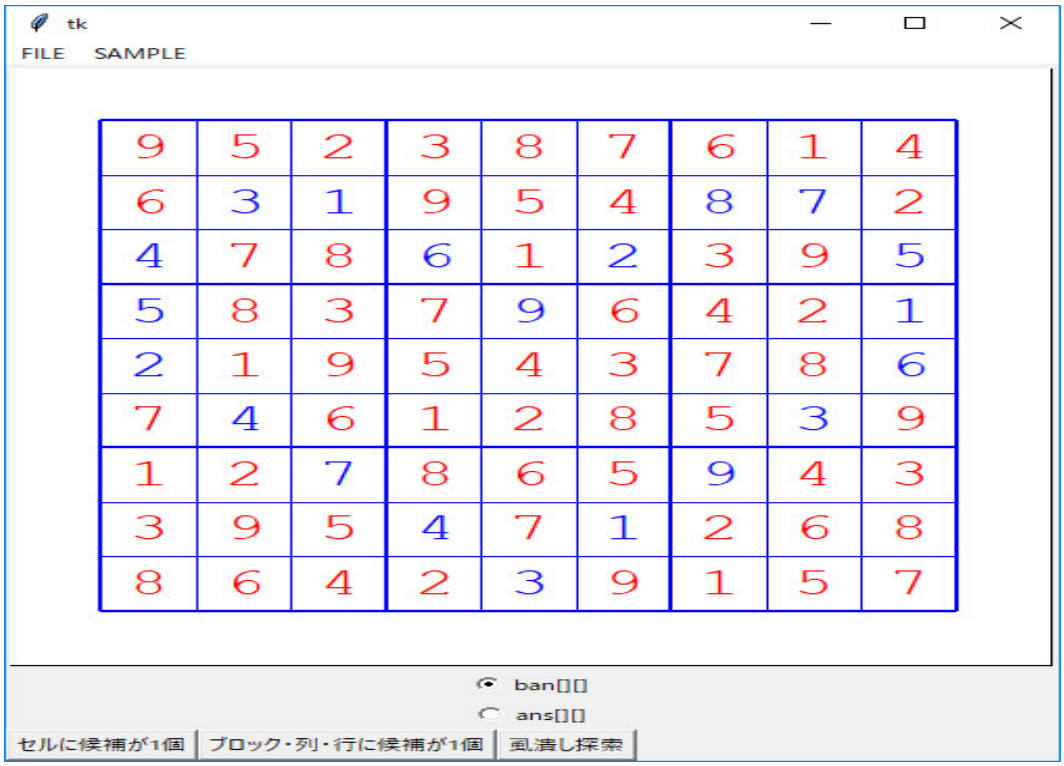
```
def RegularPlay():
    while True:
        while True:
            F = SimplePlay()
            if not F:
                break
        F = StandardPlay()
        if not F:
            break
```

と定義し、関数 solver() を

```
def solver():
    global n_solve, ans, cand
    RegularPlay()
    if losingP():
        return False
    if completeP():
        return True
    flag = False
    for k in range(9):
        for i in range(9):
```

```
            if ans[k][i] == 0:
                kk = k
                ii = i
                flag = True
                break
        if flag:
            break
    set_cand(kk, ii)
    S = cand[kk][ii].copy()
    tempAns = copy.deepcopy(ans)
    for n in S:
        ans[kk][ii] = n
        if solver():
            return True
        ans = copy.deepcopy(tempAns)
        ans[kk][ii] = 0
    return False
```

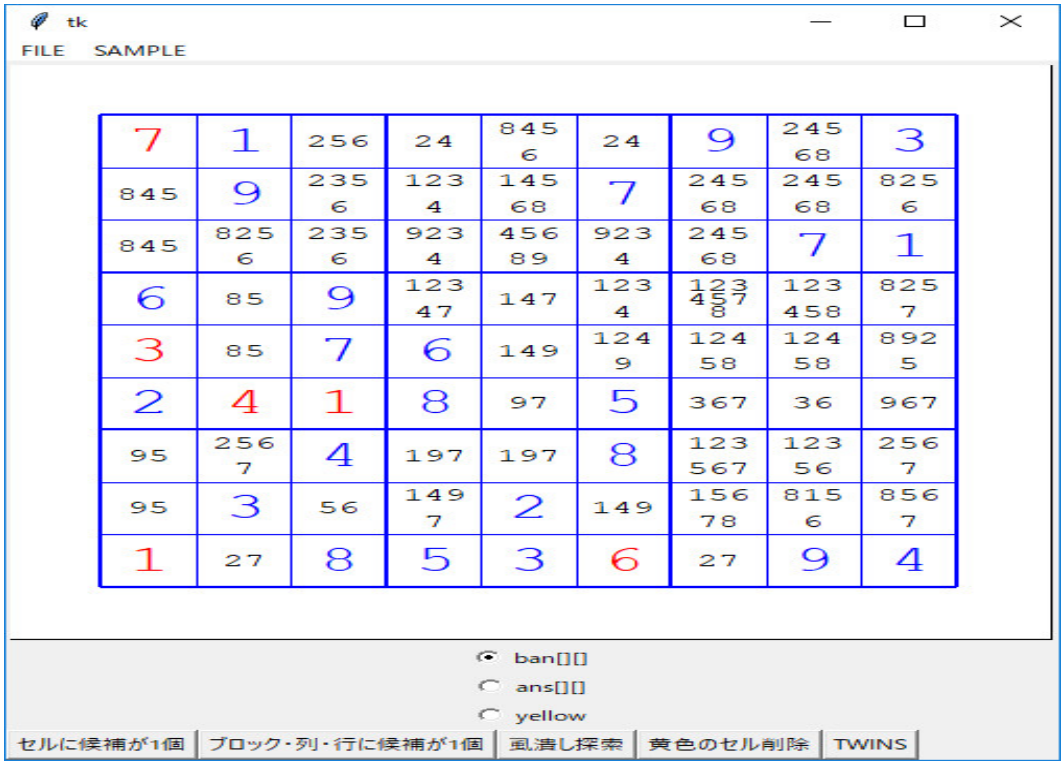
と修正します。実行すると



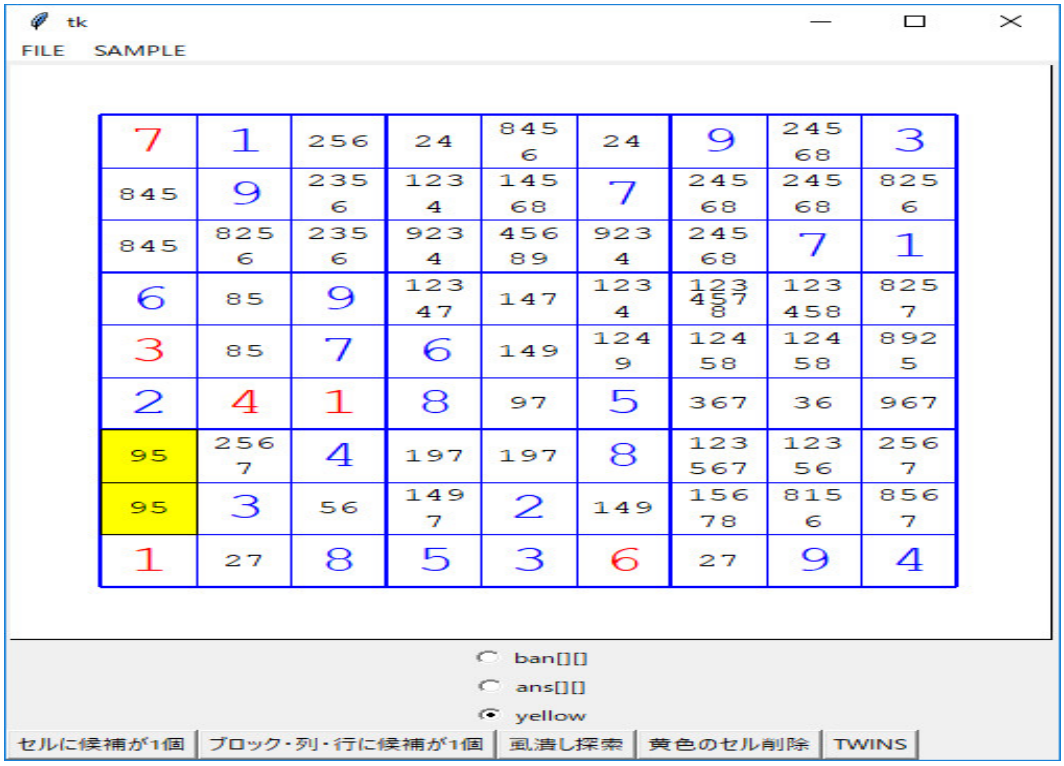
のような問題でも、高速に実行できるようになりました。

Jason Rosenhouse, Laura Taalman 著「Taking Sudoku Scriously : The Math Behind the Worlds Most Popular Pencil Puzzle」に載っている解法の 1 つをプログラミングしてみましょう。

Sample1 の問題で、「虱潰し探索」のボタンを使わなければ、



までしか解けません。ここで、下段左隅のブロックに注目すると



のように、候補が {9,5} のセルが二つあります。これらのセルはどちらかが９で、どちらかが５のはずです。そうだとすると、３の右側のセルは候補{5,6} の内の５はこのセルのために使えない

ので、6 と確定します。これが「TWINS」の法則です。これをプログラミングしてみましょう。
そこで、「TWINS」の法則をプログラミングするために、

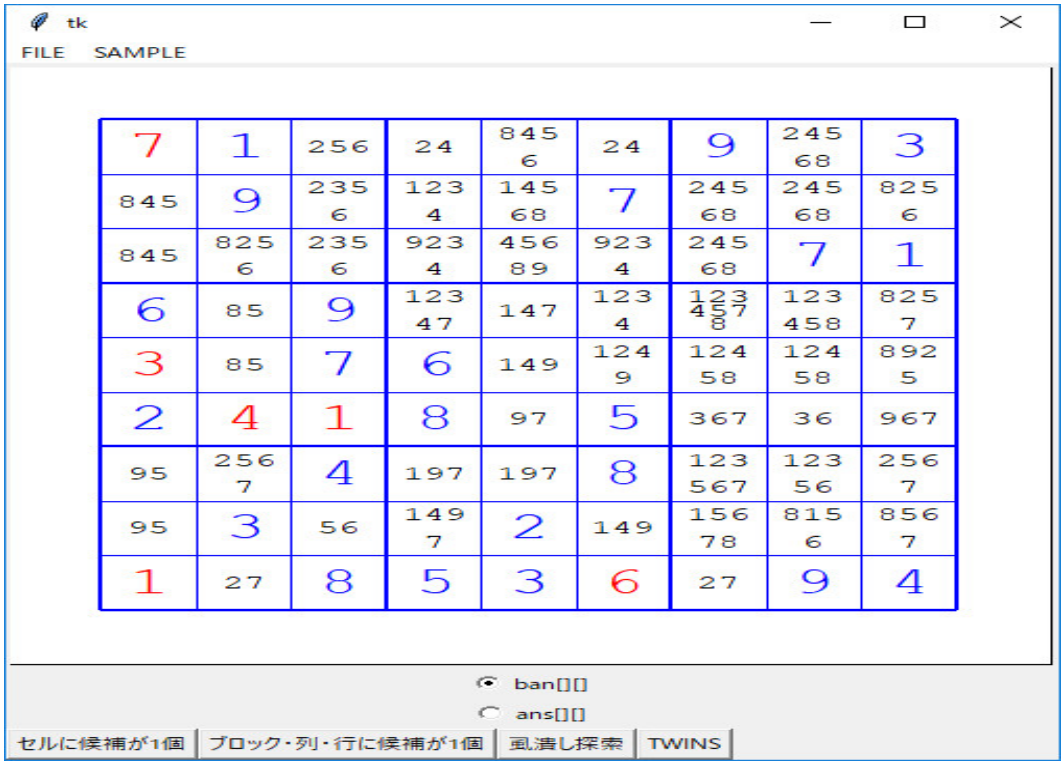
```
button4 = Button(root, text='TWINS', command=button4_clicked)
button4.pack(side='left')
```

を追加し、関数 button4_clicked() を

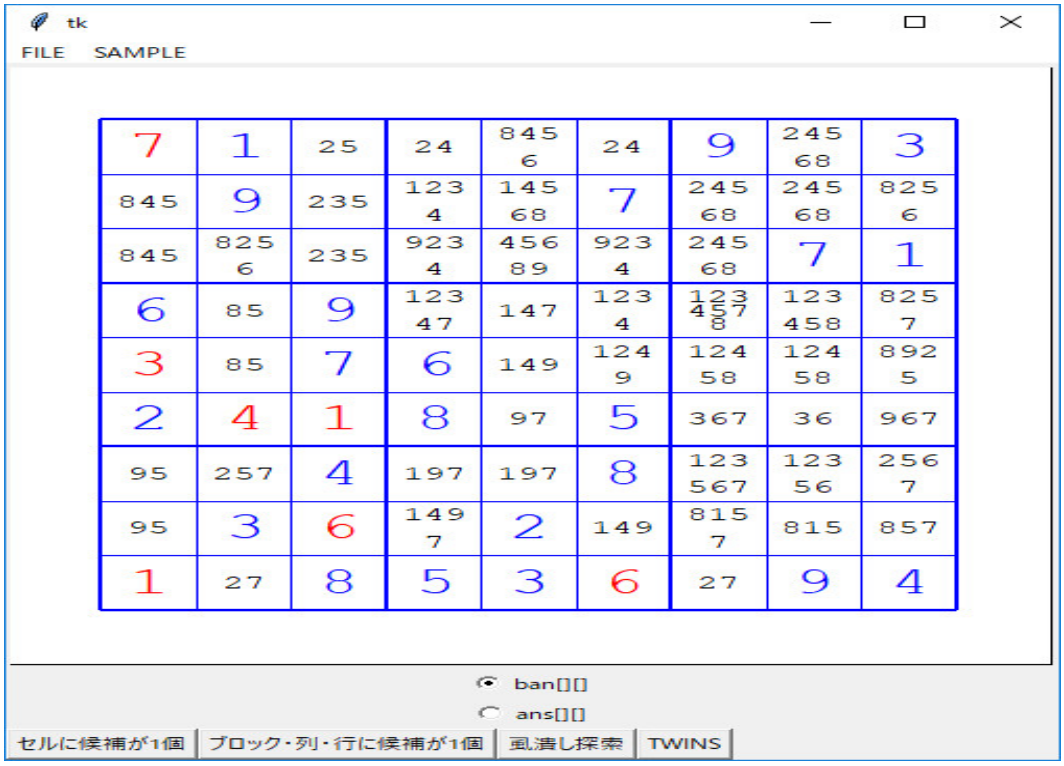
```
def button4_clicked():
    global ans, cand
    # 行の TWINS
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            set_cand(k, i)
            if len(cand[k][i]) == 2:
               for j in range(i+1, 9):
                   if ans[k][j]>0: continue
                   set_cand(k, j)
                   if cand[k][i] == cand[k][j]:
                       for p in range(9):
                           if p == i or p == j: continue
                           if ans[k][p]>0: continue
                           set_cand(k, p)
                           C = cand[k][p] - cand[k][i]
                           if len(C) == 1:
                               ans[k][p] = C.pop()
    # 列の TWINS
    for i in range(9):
        for k in range(9):
            if ans[k][i]>0: continue
            set_cand(k, i)
            if len(cand[k][i]) == 2:
               for j in range(k+1, 9):
                   if ans[j][i]>0: continue
                   set_cand(j, i)
                   if cand[k][i] == cand[j][i]:
                       for p in range(9):
                           if p == k or p == j: continue
                           if ans[p][i]>0: continue
                           set_cand(p, i)
                           C = cand[p][i] - cand[k][i]
                           if len(C) == 1:
                               ans[p][i] = C.pop()
    # ブロックの TWINS
```

```
    for kk in range(3):
        for ii in range(3):
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    if ans[k][i] > 0: continue
                    set_cand(k, i)
                    if len(cand[k][i]) == 2:
                        for p in range(3*kk, 3*(kk+1)):
                            for q in range(3*ii, 3*(ii+1)):
                                if p<k or p ==k and q<=i: continue
                                if ans[p][q] > 0: continue
                                set_cand(p, q)
                                if cand[k][i] == cand[p][q]:
                                   for r in range(3*kk, 3*(kk+1)):
                                        for s in range(3*ii, 3*(ii+1)):
                                            if (r==k and s==i)or(r==p and s==q):
                                                continue
                                            if ans[r][s] > 0: continue
                                            set_cand(r, s)
                                            C = cand[r][s] - cand[k][i]
                                            if len(C) == 1:
                                                ans[r][s] = C.pop()
    ShowBan()
```

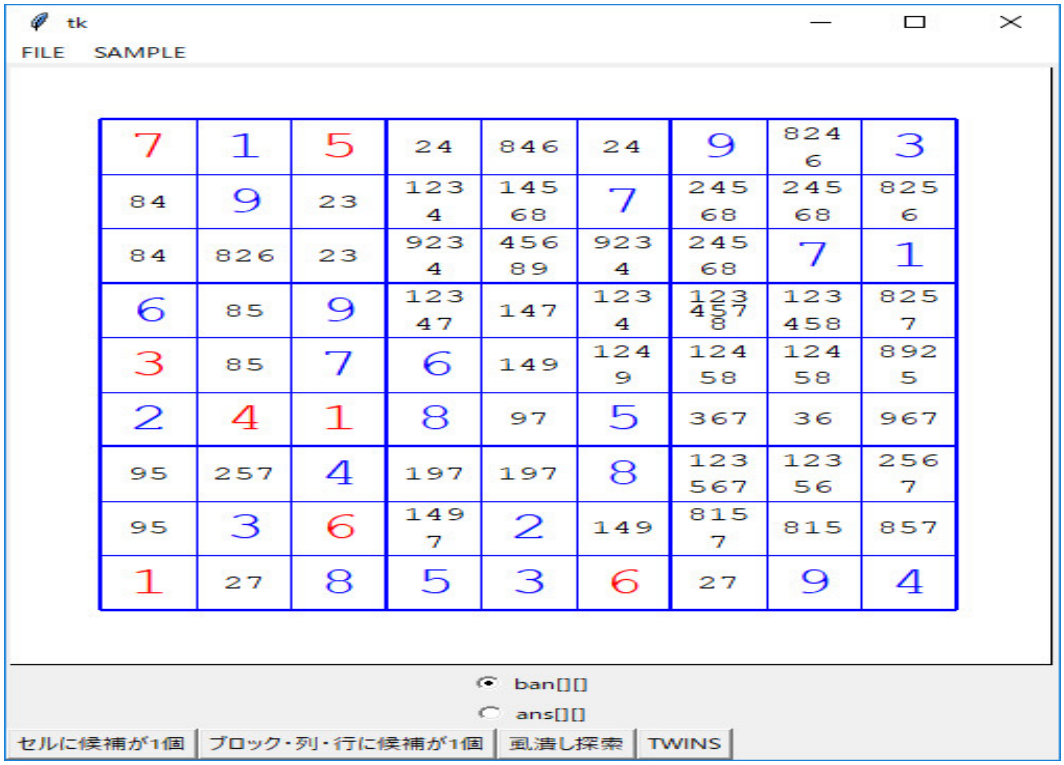
と定義します。実行し、Sample1 を表示し、「セルに候補が 1 個」と「ブロック・行・列に候補が 1 個」のボタンを何回か押し、



とし、「TWINS」のボタンをクリックすると



となり、もう一度「TWINS」のボタンをクリックすると



となります。この局面を見ると、左上隅のブロックで TWINS が {8,4} と {2,3} と二つあるので、9 の下のセルは候補 {8,2,6} のうち 8 と 6 が削除され 6 に確定しますが、このような法則をプログラムするのはめんどくさいですが、「虱潰し探索」を使わず解くためには、1 つ 1 つ見つけた解法をプログラミングしてく必要があります。 これは、「二組の双子」の法則ですが、「双子」の法則があれば、当然、「三つ子 (TRIPLETS)」の法則や「四つ子 (QUADS)」の法則、およびその変形が適用できる局面があることが予想されます。数学ではこのようにして、法則を見つけていきます。

もう一つ、Jason Rosenhouse, Laura Taalman 著「Taking Sudoku Scriously : The Math Behind the Worlds Most Popular Pencil Puzzle」に載っている解法をプログラミングしてみましょう。その解法が有効な例もこの本のもの（三番目の問題）を使います。

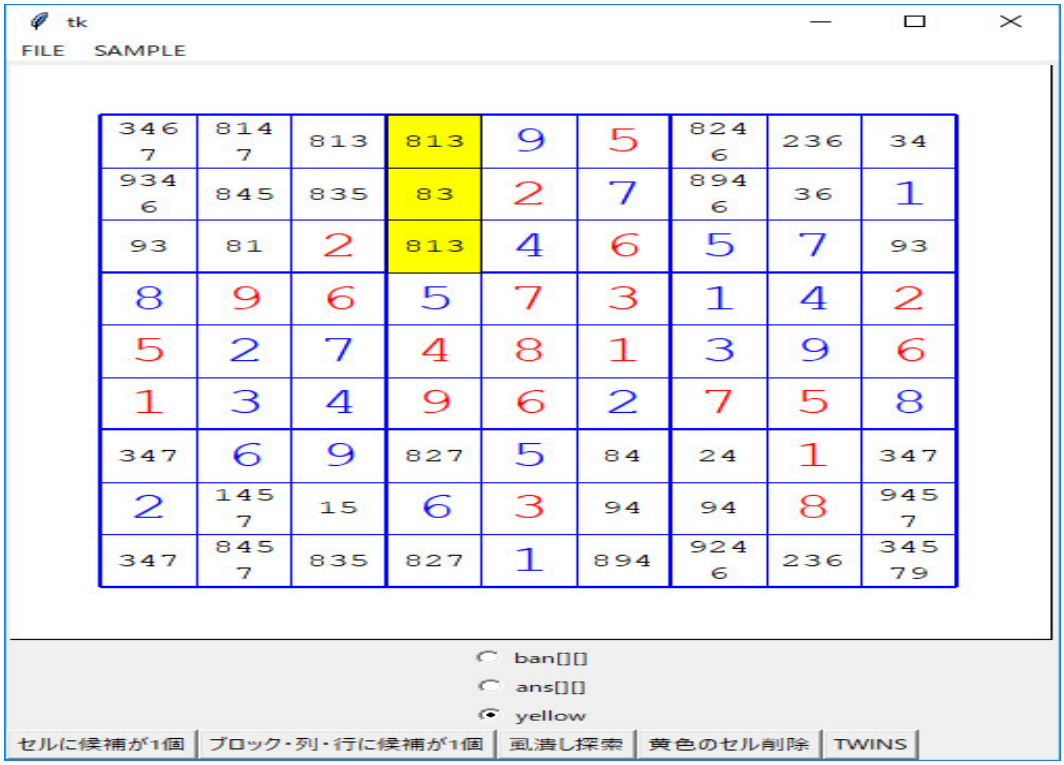
```
menu_SAMPLE.add_command(label='Sample6', command=fun_sample6)
```

を追加し、関数 fun_sample6() を

```
def fun_sample6():
    global ban
    ban = [
        [0,0,0,0,9,0,0,0,0],
        [0,0,0,0,0,7,0,0,1],
        [0,0,0,0,4,0,5,7,0],
        [8,0,0,5,0,0,1,4,0],
        [0,2,7,0,0,0,3,9,0],
        [0,3,4,0,0,2,0,0,8],
```

```
[0,6,9,0,5,0,0,0,0],
[2,0,0,6,0,0,0,0,0],
[0,0,0,0,1,0,0,0,0]]
```

と定義します。実行し、Sample6 を表示し、「セルに候補が 1 個」と「ブロック・列・行に候補が 1 個」を何回かクリックすると



となります。黄色のセルたちが「三つ子」もどきです。この列の他のセルの候補から 8 を削除できます。この局面で

の黄色のセルたちに注目します。まず、第 3 行に「TWINS」 {9,3} があります。どっちか分かりませんが、この行の両端のどちらかが 3 です。第 7 行では、この行の両端だけに 3 が入ることが分かります。従って、対角線のどちらかに 3 が入ります。従って、第 1 列と第 9 列において、黄色のセル以外には 3 は入りません。従って、これらの列の他のセルから 3 を削除できます。従って、第 9 列の一番上のセルが 4 と確定します。この法則は「X-WINGS」の法則と言われています。この法則をプログラミングしてみましょう。しかし、この法則を有用な法則にするためには、今まで作ったプログラムを大幅に変更する必要があります。上の局面で、「**TWINS**」をクリックした時、第 **3** 行の **4** 番目のセルの候補 {8,1,3} から 3 を削除するように修正すべきです。関数 set_cand() はそのようなことを考慮していないので、最初に関数 set_cand() で候補をセットしたら、後は候補の変化だけを修正し、無暗に関数 set_cand() を使って、「**TWINS**」での修正をぶち壊してはいけません。良くできた参考書では、途中で大幅な変更が必要ないように細心の注意を払って、議論を進めていますが、初心者がプログラムを作るときには、大抵初めて作るプログラムを思いつくまま作っているので、このような大幅な方針変更が良く起こります。このような修正をする時、発見の難しいバグが入り込みます。 このような経験を積んでいけば、プログラミングできるようになります。

関数 set_hint() と change_cand(k, i) を

```
def set_hint():
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            set_cand(k, i)

def change_cand(k, i):
```

```
    # ans[k][i] = n としたための cand の変化
    for p in range(9):
        if ans[p][i] > 0: continue
        cand[p][i] = cand[p][i] - {ans[k][i]}
    for p in range(9):
        if ans[k][p] > 0: continue
        cand[k][p] = cand[k][p] - {ans[k][i]}
    kk = k // 3
    ii = i // 3
    for p in range(3*kk, 3*(kk+1)):
        for q in range(3*ii, 3*(ii+1)):
            if ans[p][q] > 0: continue
            cand[p][q] = cand[p][q] - {ans[k][i]}
```

と定義し、関数 fun_sample1() から fun_sample6() までに

```
def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
    ShowBan()
```

や

```
def fun_sample6():
    global ban
    ban = [
        [0,0,0,0,9,0,0,0,0],
        [0,0,0,0,0,7,0,0,1],
        [0,0,0,0,4,0,5,7,0],
        [8,0,0,5,0,0,1,4,0],
        [0,2,7,0,0,0,3,9,0],
        [0,3,4,0,0,2,0,0,8],
```

```
        [0,6,9,0,5,0,0,0,0],
        [2,0,0,6,0,0,0,0,0],
        [0,0,0,0,1,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
    ShowBan()
```

のように

```
    global cand
    set_hint()
```

を追加し、それ以降の関数から、関数 set_cand() をすべて削除します。関数 buttonPress(event) を

```
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
##        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)
        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        change_cand(k, i)
        ShowBan()
```

と修正し、関数 open_ban() を

```
def open_ban():
    global path_name, ban, ans
    ban = [[0]*9]*9
    ans = [[0]*9]*9
    filename = fd.askopenfilename(initialdir=path_name)
    if filename:
```

```
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        for cnt in range(18):
            line = f.readline()
            s = line.split()
            if cnt < 9:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ban[cnt] = nlist
            else:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ans[cnt-9] = nlist
            cnt += 1
        f.close()
    global cand
    set_hint()
    ShowBan()
```

と修正し、関数 `print_ban()` からも `set_cand()` を削除し、関数 SimplePlay() を

```
def SimplePlay():
    global ans, cand
    FLAG = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0:
                continue
##            set_cand(k, i)
            if len(cand[k][i]) == 1:
                ans[k][i] = cand[k][i].pop()
                change_cand(k, i)
                FLAG = True
    return FLAG
```

と修正し、関数 StandardPlay() を

```
def StandardPlay():
    global ans, cand
    FLAG = False
    for k in range(9):
```

```
        for i in range(9):
            if ans[k][i] > 0: continue
            ## 行のチェック
##            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == i: continue
                    if ans[k][j] > 0: continue;
##                    set_cand(k, j)
                    if n in cand[k][j]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
                    change_cand(k, i)
                    FLAG = True
            ## 列のチェック
##            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == k: continue
                    if ans[j][i] > 0: continue;
##                    set_cand(j, i)
                    if n in cand[j][i]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
                    change_cand(k, i)
                    FLAG = True
            ## ブロックのチェック
##            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                kk = k // 3
                ii = i // 3
                for r in range(3*kk, 3*(kk+1)):
                    for c in range(3*ii, 3*(ii+1)):
```

```
                        if r == k and c == i: continue
                        if ans[r][c] > 0: continue
##                        set_cand(r, c)
                        if n in cand[r][c]:
                            flag = True
                            break
                if not flag:
                    ans[k][i] = n
                    change_cand(k, i)
                    FLAG = True
    return FLAG
```

と修正し、関数 losingP() を

```
def losingP():
    global cand
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
##                set_cand(k, i)
                if len(cand[k][i]) == 0:
                    return True
    for k in range(9):
        P = [0]*10
        for i in range(9):
            P[ans[k][i]] = P[ans[k][i]] + 1
        for n in range(1, 10):
            if P[n] > 1:
                return True
    for i in range(9):
        P = [0]*10
        for k in range(9):
            P[ans[k][i]] = P[ans[k][i]] + 1
        for n in range(1, 10):
            if P[n] > 1:
                return True
    for kk in range(3):
        for ii in range(3):
            P = [0]*10
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    P[ans[k][i]] = P[ans[k][i]] + 1
            for n in range(1, 10):
                if P[n] > 1:
```

```
                    return True
    return False
```

と修正し、関数 solver() を

```
def solver():
    global ans, cand
    RegularPlay()
    if losingP():
        return False
    if completeP():
        return True
    flag = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
                kk = k
                ii = i
                flag = True
                break
        if flag:
            break
    tempAns = copy.deepcopy(ans)
    tempCand = copy.deepcopy(cand)
    set_cand(kk, ii)
    S = cand[kk][ii].copy()
    for n in S:
        tempCand2 = copy.deepcopy(cand)
        ans[kk][ii] = n
        change_cand(kk, ii)
        if solver():
            return True
        ans = copy.deepcopy(tempAns)
        cand = copy.deepcopy(tempCand2)
        ans[kk][ii] = 0
    return False
```

と修正し、関数 button4_clicked() を

```
def button4_clicked():
    global ans, cand
    # 行の TWINS
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
```

```
##            set_cand(k, i)
           if len(cand[k][i]) == 2:
              for j in range(i+1, 9):
                  if ans[k][j]>0: continue
##                  set_cand(k, j)
                  if cand[k][i] == cand[k][j]:
                      for p in range(9):
                          if p == i or p == j: continue
                          if ans[k][p]>0: continue
##                          set_cand(k, p)
                          cand[k][p] = cand[k][p] - cand[k][i]
                          if len(cand[k][p]) == 1:
                              ans[k][p] = cand[k][p].pop()
                              change_cand(k, p)
    # 列の TWINS
    for i in range(9):
        for k in range(9):
            if ans[k][i]>0: continue
##            set_cand(k, i)
            if len(cand[k][i]) == 2:
               for j in range(k+1, 9):
                   if ans[j][i]>0: continue
##                   set_cand(j, i)
                   if cand[k][i] == cand[j][i]:
                       for p in range(9):
                           if p == k or p == j: continue
                           if ans[p][i]>0: continue
##                           set_cand(p, i)
                           cand[p][i] = cand[p][i] - cand[k][i]
                           if len(cand[p][i]) == 1:
                               ans[p][i] = cand[p][i].pop()
                               change_cand(p, i)
    # ブロックの TWINS
    for kk in range(3):
        for ii in range(3):
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    if ans[k][i] > 0: continue
##                    set_cand(k, i)
                    if len(cand[k][i]) == 2:
                        for p in range(3*kk, 3*(kk+1)):
                            for q in range(3*ii, 3*(ii+1)):
                                if p<k or p ==k and q<=i: continue
```

```
                                if ans[p][q] > 0: continue
##                                set_cand(p, q)
                                if cand[k][i] == cand[p][q]:
                                   for r in range(3*kk, 3*(kk+1)):
                                        for s in range(3*ii, 3*(ii+1)):
                                            if (r==k and s==i)or(r==p and s==q):
                                                continue
                                            if ans[r][s] > 0: continue
##                                            set_cand(r, s)
                                            cand[r][s] = cand[r][s] - cand[k][i]
                                            if len(cand[r][s]) == 1:
                                                ans[r][s] = cand[r][s].pop()
                                                change_cand(r, s)
    ShowBan()
```

と修正します。最後に

```
button5 = Button(root, text='X-WINGS', command=button5_clicked)
button5.pack(side='left')
```

を追加し、関数 button5_clicked() を

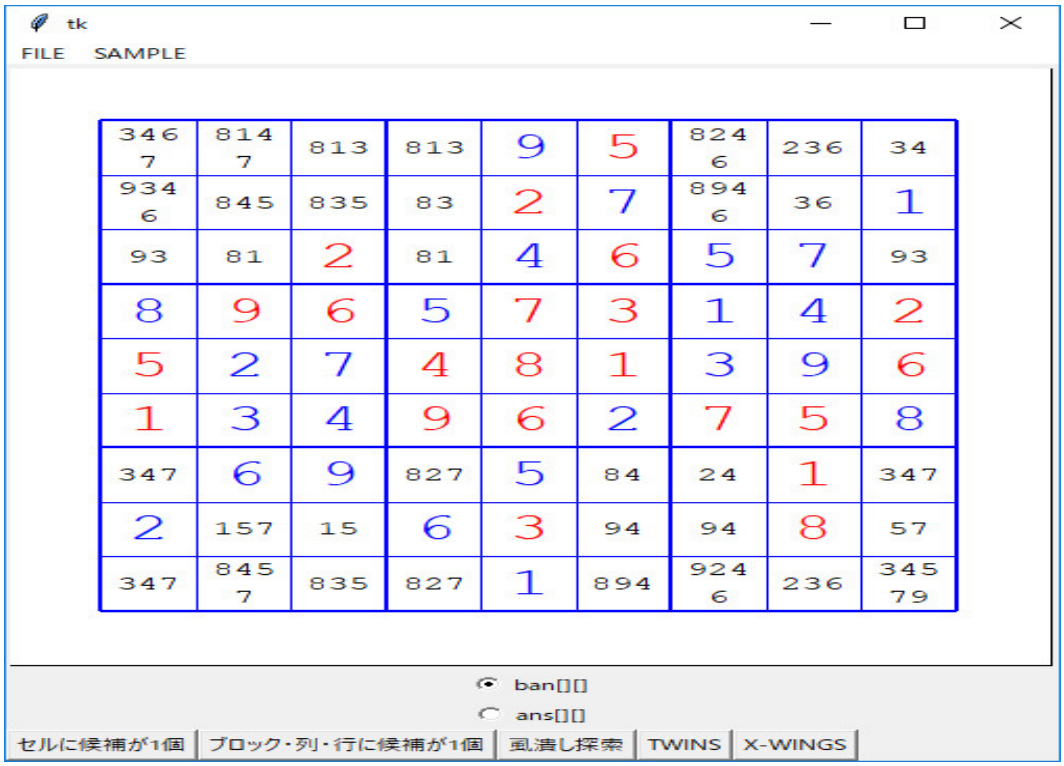
```
def button5_clicked():
    global ans, cand
    # 行の X-WINGS
    for n in range(1,10):
        for k in range(9):
            index = []
            for i in range(9):
                if ans[k][i] > 0: continue;
                if n in cand[k][i]:
                    index.append(i)
            if len(index) == 2:
                for j in range(k+1, 9):
                    index2 = []
                    for p in range(9):
                        if ans[j][p] > 0: continue
                        if n in cand[j][p]:
                            index2.append(p)
                    if len(index2) == 2:
                        if index[0]==index2[0] and index[1]==index2[1]:
                            for q in range(9):
                                if q==k or q==j: continue
                                if ans[q][index[0]] > 0: continue
                                cand[q][index[0]] = cand[q][index[0]] - {n}
```

```
                            if len(cand[q][index[0]]) == 1:
                                ans[q][index[0]] = cand[q][index[0]].pop()
                                change_cand(q, index[0])
                        for q in range(9):
                            if q==k or q==j: continue
                            if ans[q][index[1]] > 0: continue
                            cand[q][index[1]] = cand[q][index[1]] - {n}
                            if len(cand[q][index[1]]) == 1:
                                ans[q][index[1]] = cand[q][index[1]].pop()
                                change_cand(q, index[1])
# 列の X-WINGS
for n in range(1,10):
    for i in range(9):
        index = []
        for k in range(9):
            if ans[k][i] > 0: continue;
            if n in cand[k][i]:
                index.append(k)
        if len(index) == 2:
            for j in range(i+1, 9):
                index2 = []
                for p in range(9):
                    if ans[p][j] > 0: continue
                    if n in cand[p][j]:
                        index2.append(p)
                if len(index2) == 2:
                    if index[0]==index2[0] and index[1]==index2[1]:
                        for q in range(9):
                            if q==i or q==j: continue
                            if ans[index[0]][q] > 0: continue
                            cand[index[0]][q] = cand[index[0]][q] - {n}
                            if len(cand[index[0]][q]) == 1:
                                ans[index[0]][q] = cand[index[0]][q].pop()
                                change_cand(index[0], q)
                        for q in range(9):
                            if q==i or q==j: continue
                            if ans[index[1]][q] > 0: continue
                            cand[index[1]][q] = cand[index[1]][q] - {n}
                            if len(cand[index[1]][q]) == 1:
                                ans[index[1]][q] = cand[index[1]][q].pop()
                                change_cand(index[1], q)

ShowBan()
```

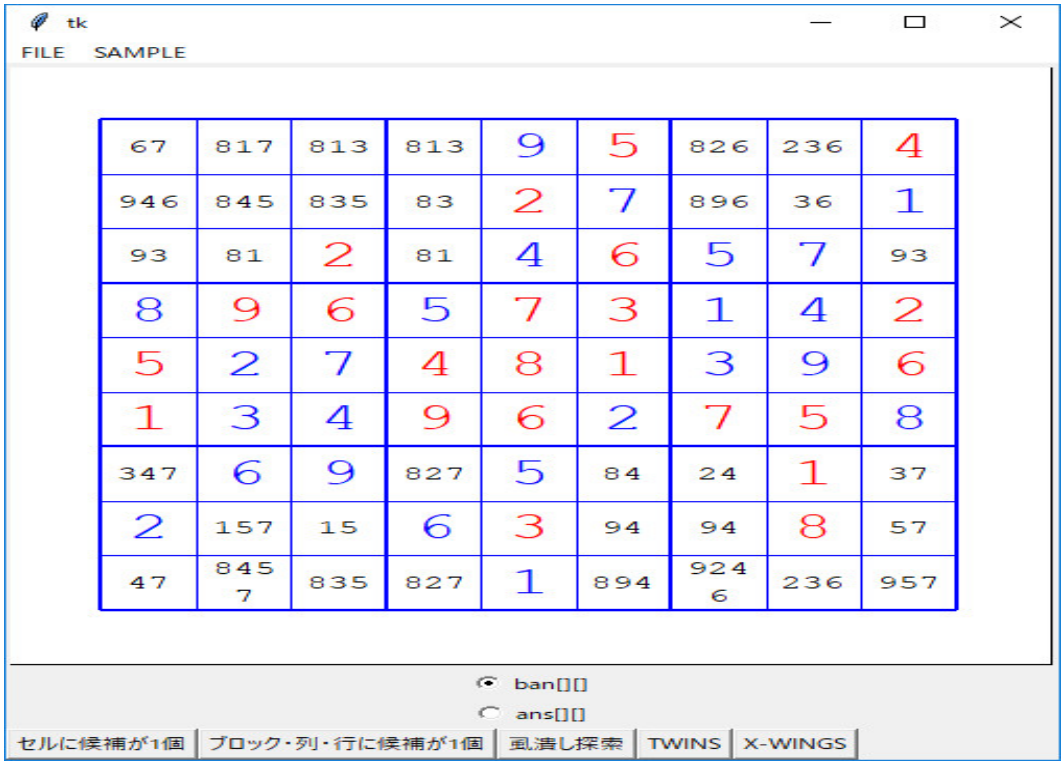
と定義します。

実行し、Sample6 を表示し、「セルに候補が 1 個」と「ブロック・列・行に候補が 1 個」を何回かクリックすると

になり、「TWINS」をクリックすると

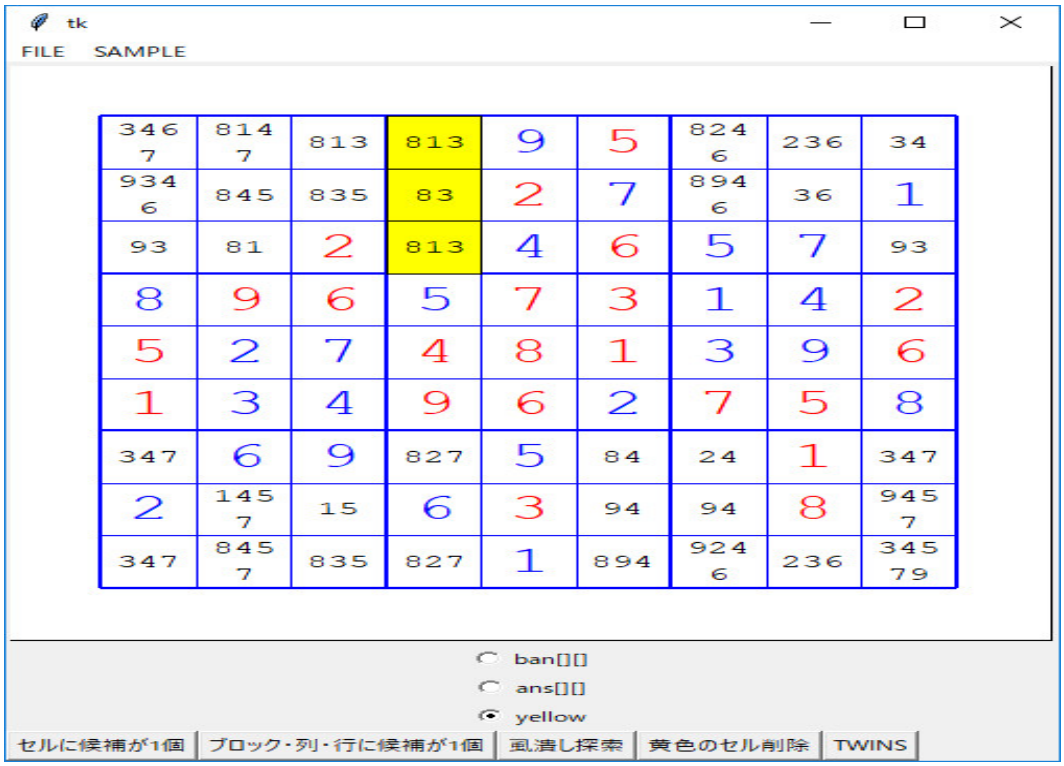


になり、「X-WINGS」をクリックすると



となります。

実行し、Sample6 を表示し、「セルに候補が 1 個」と「ブロック・列・行に候補が 1 個」を何回かクリックすると



となります。黄色のセルたちが「三つ子 (Triplets)」です。この列の他のセルの候補から 8 を削除できます。「三つ子 (TRIPLETS)」の法則をプログラミングしてみましょう。

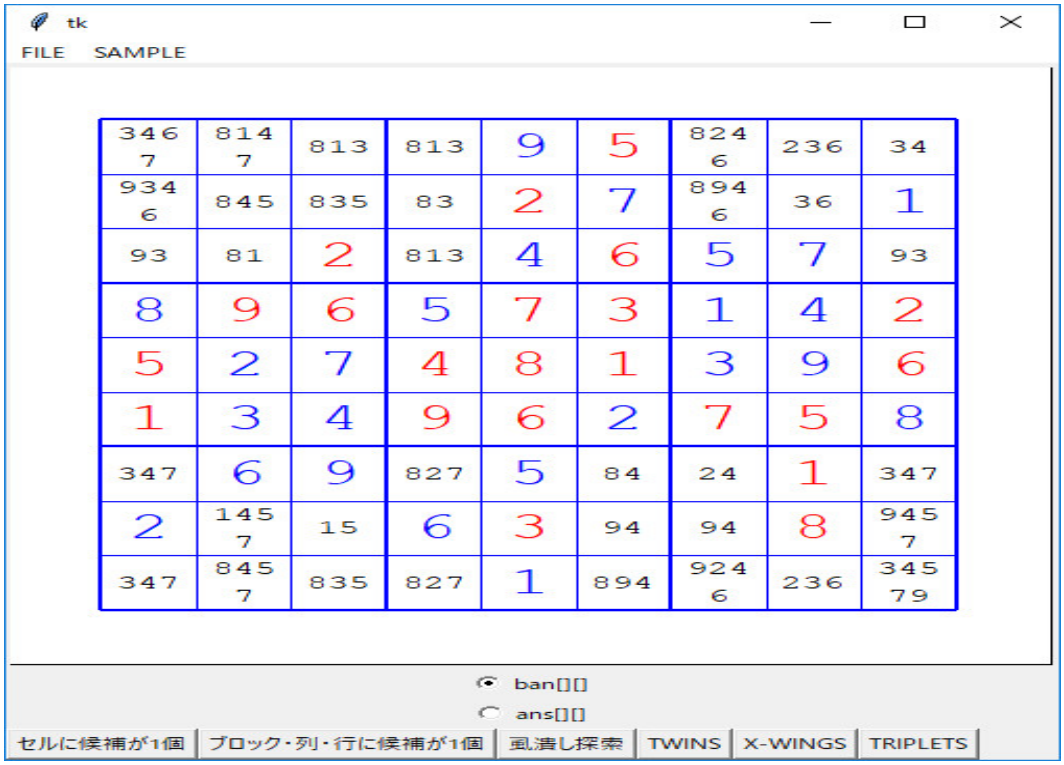
```
button6 = Button(root, text='TRIPLETS', command=button6_clicked)
button6.pack(side='left')
```

を追加し、関数 button6_clicked() を

```
def button6_clicked():
    global ans, cand
    # 行の TRIPLETS
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            if len(cand[k][i]) == 3:
                index = [i]
                for j in range(9):
                   if j == i: continue
                   if ans[k][j]>0: continue
                   if cand[k][j] <= cand[k][i]:
                       index.append(j)
                if len(index) == 3:
                    for p in range(9):
                        if p == i or p == index[1] or p == index[2]: continue
                        if ans[k][p]>0: continue
                        cand[k][p] = cand[k][p] - cand[k][i]
                        if len(cand[k][p]) == 1:
                            ans[k][p] = cand[k][p].pop()
                            change_cand(k, p)
    # 列の TRIPLETS
    for i in range(9):
        for k in range(9):
            if ans[k][i]>0: continue
            if len(cand[k][i]) == 3:
                index = [k]
                for j in range(9):
                   if j == k: continue
                   if ans[j][i]>0: continue
                   if cand[j][i] <= cand[k][i]:
                       index.append(j)
                if len(index) == 3:
                    for p in range(9):
                        if p == k or p == index[1] or p == index[2]: continue
                        if ans[p][i]>0: continue
                        cand[p][i] = cand[p][i] - cand[k][i]
```

```
                    if len(cand[p][i]) == 1:
                        ans[p][i] = cand[p][i].pop()
                        change_cand(p, i)
    # ブロックの TRIPLETS
    for kk in range(3):
        for ii in range(3):
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    if ans[k][i] > 0: continue
                    if len(cand[k][i]) == 3:
                        index = [(k, i)]
                        for p in range(3*kk, 3*(kk+1)):
                            for q in range(3*ii, 3*(ii+1)):
                                if p == k and q == i: continue
                                if ans[p][q] > 0: continue
                                if cand[p][q] <= cand[k][i]:
                                    index.append((p, q))
                        if len(index) == 3:
                            for r in range(3*kk, 3*(kk+1)):
                                for s in range(3*ii, 3*(ii+1)):
                                    if (r,s)==(k,i) or(r,s)==index[1] or \
                                       (r,s)==index[2]:
                                                continue
                                    if ans[r][s] > 0: continue
                                    cand[r][s] = cand[r][s] - cand[k][i]
                                    if len(cand[r][s]) == 1:
                                        ans[r][s] = cand[r][s].pop()
                                        change_cand(r, s)
    ShowBan()
```

と定義します。実行し、Sample6 を表示し、「セルに候補が 1 個」と「ブロック・列・行に候補が 1 個」を何回かクリックすると



となります。「TRIPLETS」をクリックすると

となります。ところで、「TRIPLETS」の法則は、3 つのセルの候補が {a,b}、 {b,c}、 {c,a} でも成立するはずです。これもプログラミングしてみます。関数 button6_clicked() に

```
for k in range(9):
    for i in range(9):
        if ans[k][i]>0: continue
        if len(cand[k][i]) == 2:
             for j in range(i+1, 9):
               if ans[k][j]>0: continue
               if len(cand[k][j]) == 2:
                    if len(cand[k][i] & cand[k][j]) == 1:
                       S = cand[k][i] | cand[k][j]
                       index = [i, j]
                       for p in range(j+1, 9):
                           if ans[k][p] > 0: continue
                           if cand[k][p] <= S:
                               index.append(p)
                       if len(index) == 3:
                          for p in range(9):
                            if p == i or p == j or p == index[2]: continue
                            if ans[k][p]>0: continue
                            cand[k][p] = cand[k][p] - S
                            if len(cand[k][p]) == 1:
                                ans[k][p] = cand[k][p].pop()
                                change_cand(k, p)
for i in range(9):
    for k in range(9):
        if ans[k][i]>0: continue
        if len(cand[k][i]) == 2:
             for j in range(k+1, 9):
               if ans[j][i]>0: continue
               if len(cand[j][i]) == 2:
                    if len(cand[k][i] & cand[j][i]) == 1:
                       S = cand[k][i] | cand[j][i]
                       index = [k, j]
                       for p in range(j+1, 9):
                           if ans[p][i]>0: continue
                           if cand[p][i] <= S:
                               index.append(p)
                       if len(index) == 3:
                            for p in range(9):
                                if p == k or p == j or p == index[2]:
                                    continue
                                if ans[p][i]>0: continue
                                cand[p][i] = cand[p][i] - S
                                if len(cand[p][i]) == 1:
```

```
                                    ans[p][i] = cand[p][i].pop()
                                    change_cand(p, i)
for kk in range(3):
    for ii in range(3):
        for k in range(3*kk, 3*(kk+1)):
            for i in range(3*ii, 3*(ii+1)):
                if ans[k][i] > 0: continue
                if len(cand[k][i]) == 2:
                    for p in range(3*kk, 3*(kk+1)):
                        for q in range(3*ii, 3*(ii+1)):
                            if p<k or p ==k and q<=i: continue
                            if ans[p][q] > 0: continue
                            if len(cand[p][q]) == 2:
                                if len(cand[k][i] & cand[p][q]) == 1:
                                    S = cand[k][i] | cand[p][q]
                                    index=[(k,i),(p, q)]
                                    for r in range(3*kk, 3*(kk+1)):
                                        for s in range(3*ii, 3*(ii+1)):
                                           if r<p or r ==p and s<=q:
                                               continue
                                           if ans[r][s] > 0: continue
                                           if cand[r][s] <= S:
                                               index.append((r,s))
                                    if len(index) == 3:
                                        for r in range(3*kk, 3*(kk+1)):
                                            for s in range(3*ii, 3*(ii+1)):
                                                if (r,s)==(k,i) or \
                                                   (r,s)==(p,q) or \
                                                   (r,s)==index[2]:
                                                    continue
                                                if ans[r][s] > 0: continue
                                                cand[r][s] = cand[r][s] - S
                                                if len(cand[r][s]) == 1:
                                                    ans[r][s]=cand[r][s].pop()
                                                    change_cand(r, s)
```

を追加します。「四つ子 (QUADS)」の法則も同様にプログラミングできます。さらに、「X-WINGS」の法則の拡張である 3 × 3 の「 Swordfish」や 4 × 4 の「 Squirmbag」の法則もあります。これら以外にも数独の法則はたくさん知られています。きりがありませんから、ここでやめます。興味があれば、自分でやってみてください。Jason Rosenhouse, Laura Taalman 著「Taking Sudoku Scriously : The Math Behind the Worlds Most Popular Pencil Puzzle」は面白い本です。数独に関連する数学が幅広く解説されています。ぜひ、数独に興味がわいたら読んでみてください。翻訳もありますが、この本の英語は難しくありません。

プログラムの全体は

```
from tkinter import *
import tkinter.simpledialog as sd
import copy
import tkinter.filedialog as fd
import sys, os.path
import win32print
import win32con
import win32gui
import win32ui

board_size = 500
ban = []
ans = []
cand =[[set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()],
       [set(),set(),set(),set(),set(),set(),set(),set(),set()]]

path_name = ""
file_name = ""

def set_cand(k, i):
    global cand
    numYoko = set()
    for j in range(9):
        if ans[k][j] > 0:
            numYoko.add(ans[k][j])
    numTate = set()
    for j in range(9):
        if ans[j][i] > 0:
            numTate.add(ans[j][i])
    numBlock = set()
    ii = i//3
    kk = k//3
    for r in range(3*kk, 3*(kk+1)):
        for c in range(3*ii, 3*(ii+1)):
```

```
            if ans[r][c] > 0:
                numBlock.add(ans[r][c])
    cand[k][i].clear()
    for n in range(1, 10):
        cand[k][i].add(n)
    cand[k][i] = cand[k][i] - (numYoko | numTate | numBlock)

def set_hint():
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            set_cand(k, i)

def change_cand(k, i):
    # ans[k][i] = n としたための cand の変化
    for p in range(9):
        if ans[p][i] > 0: continue
        cand[p][i] = cand[p][i] - {ans[k][i]}
    for p in range(9):
        if ans[k][p] > 0: continue
        cand[k][p] = cand[k][p] - {ans[k][i]}
    kk = k // 3
    ii = i // 3
    for p in range(3*kk, 3*(kk+1)):
        for q in range(3*ii, 3*(ii+1)):
            if ans[p][q] > 0: continue
            cand[p][q] = cand[p][q] - {ans[k][i]}

def fun_sample1():
    global ban
    ban = [
        [0,1,0,0,0,0,9,0,3],
        [0,9,0,0,0,7,0,0,0],
        [0,0,0,0,0,0,0,7,1],
        [6,0,9,0,0,0,0,0,0],
        [0,0,7,6,0,0,0,0,0],
        [2,0,0,8,0,5,0,0,0],
        [0,0,4,0,0,8,0,0,0],
        [0,3,0,0,2,0,0,0,0],
        [0,0,8,5,3,0,0,9,4]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
```

```
    set_hint()
    ShowBan()
def fun_sample2():
    global ban
    ban = [
        [0,0,0,0,0,6,2,0,1],
        [0,0,0,0,4,0,3,7,0],
        [0,0,3,0,9,2,0,0,6],
        [4,0,5,0,0,0,0,6,0],
        [0,0,0,0,7,0,0,0,0],
        [0,2,0,0,0,0,4,0,5],
        [9,0,0,4,5,0,1,0,0],
        [0,5,4,0,2,0,0,0,0],
        [7,0,1,9,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
    ShowBan()
def fun_sample3():
    global ban
    ban = [
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0],
        [0,0,0,0,0,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
    ShowBan()
def fun_sample4():
    global ban
    ban = [
        [0,0,9,0,6,0,0,4,0],
        [0,4,0,0,9,0,5,0,3],
        [1,0,0,0,0,0,9,0,0],
        [9,0,0,0,3,8,0,6,5],
```

```
        [0,0,0,0,0,0,0,0,0],
        [7,6,0,5,1,0,0,0,9],
        [0,0,4,0,0,0,0,0,1],
        [2,0,3,0,5,0,0,9,0],
        [0,1,0,0,8,0,6,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
    ShowBan()
def fun_sample5():
    global ban
    ban = [
        [0,3,7,0,0,0,0,0,1],
        [2,8,0,0,0,0,0,5,0],
        [0,0,0,4,9,8,0,0,2],
        [0,0,0,3,0,0,0,6,5],
        [0,0,0,0,1,0,0,0,0],
        [3,4,0,0,0,5,0,0,0],
        [8,0,0,6,3,7,0,0,0],
        [0,1,0,0,0,0,0,8,6],
        [9,0,0,0,0,0,5,3,0]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
    ShowBan()
def fun_sample6():
    global ban
    ban = [
        [0,0,0,0,9,0,0,0,0],
        [0,0,0,0,0,7,0,0,1],
        [0,0,0,0,4,0,5,7,0],
        [8,0,0,5,0,0,1,4,0],
        [0,2,7,0,0,0,3,9,0],
        [0,3,4,0,0,2,0,0,8],
        [0,6,9,0,5,0,0,0,0],
        [2,0,0,6,0,0,0,0,0],
        [0,0,0,0,1,0,0,0,0]]
    global ans
    ans = copy.deepcopy(ban)
    global cand
    set_hint()
```

```
    ShowBan()

def ShowBan():
    canvas.create_rectangle(0, 0, board_size, board_size, fill='white')
    K = 9
    s = board_size / (K+2)
    w = h = (board_size - s * 9) / 2
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w, h+i*s, w+9*s, h+i*s, fill='blue')
    for i in range(10):
        if i % 3 == 0:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue', width=2.0)
        else:
            canvas.create_line(w+i*s, h, w+i*s, h+9*s, fill='blue')
    f1, f2 = "courier 24", "courier 12"
    c1, c2, c3 = "blue", "red", "black"
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ban[k][i]),
                                   font=f1, fill=c1)
            elif ans[k][i] > 0:
                canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=str(ans[k][i]),
                                   font=f1, fill=c2)
            else:
##                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.25)*s, text=ss1,
                                       font=f2, fill=c3)
```

```
                if ss2:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss2,
                                       font=f2, fill=c3)
                if ss1 and ss2 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                       font=f2, fill=c3)
                elif ss1 and ss:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.75)*s, text=ss,
                                      font=f2, fill=c3)
                else:
                    canvas.create_text(w+(i+0.5)*s, h+(k+0.5)*s, text=ss,
                                       font=f2, fill=c3)
def buttonPress(event):
    x = event.x
    y = event.y
    K = 9;
    s = board_size / (K+2);
    w = h = (board_size - s * 9) / 2;
    i = int((x - w) / s)
    k = int((y - h) / s)
    if i >= 0 and i < 9 and k >= 0 and k < 9:
##        set_cand(k, i)
        ss = ''
        for n in cand[k][i]:
            ss += str(n)

        result = sd.askinteger("数値入力", ss)
        if (val.get() == 0):
            ban[k][i] = result
            ans[k][i] = result
        else:
            ans[k][i] = result
        change_cand(k, i)
        ShowBan()

def save_ban():
    global path_name
    filename = fd.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        for x in ban:
            ss = ""
```

```
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        for x in ans:
            ss = ""
            for n in x:
                ss += str(n)+" "
            ss += "\n"
            f.write(ss)
        f.close()
def open_ban():
    global path_name, ban, ans
    ban = [[0]*9]*9
    ans = [[0]*9]*9
    filename = fd.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        for cnt in range(18):
            line = f.readline()
            s = line.split()
            if cnt < 9:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ban[cnt] = nlist
            else:
                nlist = []
                for i in range(9):
                    n = int(s[i])
                    nlist += [n]
                ans[cnt-9] = nlist
            cnt += 1
        f.close()
    global cand
    set_hint()
    ShowBan()

def print_ban():
    K = 9
    s = int(20000 / (K+2))
```

```
w = h = int((20000 - s * 9) / 2)
PRINTER_NAME = win32print.GetDefaultPrinter()
hprinter = win32print.OpenPrinter(PRINTER_NAME)
devmode = win32print.GetPrinter(hprinter, 9)["pDevMode"]
if devmode == None:
    devmode = win32print.GetPrinter(hprinter, 8)["pDevMode"]
devmode.PaperSize = win32con.DMPAPER_A4
devmode.Fields |= win32con.DM_PAPERSIZE
devmode.Orientation = win32con.DMORIENT_PORTRAIT # 縦
devmode.Fields |= win32con.DM_ORIENTATION
hdc = win32gui.CreateDC("WINSPOOL", PRINTER_NAME, devmode)
dc = win32ui.CreateDCFromHandle(hdc)
dc.SetMapMode(win32con.MM_HIMETRIC)
dc.StartDoc("印刷ドキュメント")
dc.StartPage()
pen = win32ui.CreatePen(0, 5, 0x666666)
pen2 = win32ui.CreatePen(0, 30, 0x666666)

dc.SelectObject(pen)
MM = 100

for i in range(10):
    if i % 3 == 0:
        dc.SelectObject(pen2)
        dc.MoveTo((w, -(h+i*s)))
        dc.LineTo((w+9*s, -(h+i*s)))
        dc.SelectObject(pen)
    else:
        dc.MoveTo((w, -(h+i*s)))
        dc.LineTo((w+9*s, -(h+i*s)))
for i in range(10):
    if i % 3 == 0:
        dc.SelectObject(pen2)
        dc.MoveTo((w+i*s, -h))
        dc.LineTo((w+i*s, -(h+9*s)))
        dc.SelectObject(pen)
    else:
        dc.MoveTo((w+i*s, -h))
        dc.LineTo((w+i*s, -(h+9*s)))

PIXELS_PER_INCH = 1440 # 1 インチ毎のピクセル数
INCH_PER_POINT = 72 # 1 インチ毎のポイント数
SCALE_FACTOR = int(PIXELS_PER_INCH / INCH_PER_POINT) #  わざわざ計算せず 20 と
```

指定する場合が多い

```
    fontdict = {
        "height": SCALE_FACTOR * 60, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font = win32ui.CreateFont(fontdict) # CFont インスタンスを作成
    fontdict2 = {
        "height": SCALE_FACTOR * 40, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font2 = win32ui.CreateFont(fontdict2) # CFont インスタンスを作成
    fontdict3 = {
        "height": SCALE_FACTOR * 20, # 10 ポイント
        "name": u"MS ゴシック", # "MS ゴシック" 等のフォント名
        "charset": win32con.SHIFTJIS_CHARSET, # シフト JIS 文字セット
    }
    font3 = win32ui.CreateFont(fontdict3) # CFont インスタンスを作成
    # フォントをセットすると今までセットされてたフォントを返してくれるので保持しとく。
    oldfont = dc.SelectObject(font)

    # 描画: http://msdn.microsoft.com/ja-jp/library/cc428775.aspx
    for k in range(9):
        for i in range(9):
            if ban[k][i] > 0:
                dc.SelectObject(font)
                dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))

            elif ans[k][i] > 0:
                dc.SelectObject(font2)
                dc.TextOut(int(w+(i+0.25)*s),-int(h+(k+0.25)*s), str(ans[k][i]))
    dc.SelectObject(font3)
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
##                set_cand(k, i)
                ss = ''
                ss1 = ''
                ss2 = ''
                for cnt, n in enumerate(cand[k][i]):
                    if cnt == 3:
```

```
                        ss1 = ss
                        ss =''
                    elif cnt == 6:
                        ss2 = ss
                        ss = ''
                    ss += str(n)
                if ss1:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.25)*s), ss1)
                if ss2:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss2)
                if ss1 and ss2 and ss:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
                elif ss1 and ss:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.75)*s), ss)
                else:
                    dc.TextOut(int(w+(i+0.5)*s), -int(h+(k+0.5)*s), ss)

    # フォントを元に戻す
    dc.SelectObject(oldfont)

    dc.EndPage()
    dc.EndDoc()
    dc.DeleteDC()

def SimplePlay():
    global ans, cand
    FLAG = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0:
                continue
##            set_cand(k, i)
            if len(cand[k][i]) == 1:
                ans[k][i] = cand[k][i].pop()
                change_cand(k, i)
                FLAG = True
    return FLAG

def button1_clicked():
    SimplePlay()
    ShowBan()

def StandardPlay():
```

```
    global ans, cand
    FLAG = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            ## 行のチェック
##            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == i: continue
                    if ans[k][j] > 0: continue;
##                    set_cand(k, j)
                    if n in cand[k][j]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
                    change_cand(k, i)
                    FLAG = True
            ## 列のチェック
##            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                for j in range(9):
                    if j == k: continue
                    if ans[j][i] > 0: continue;
##                    set_cand(j, i)
                    if n in cand[j][i]:
                        flag = True
                        break
                if not flag:
                    ans[k][i] = n
                    change_cand(k, i)
                    FLAG = True
            ## ブロックのチェック
##            set_cand(k, i)
            S = cand[k][i].copy()
            for n in S:
                flag = False
                kk = k // 3
```

```
                ii = i // 3
                for r in range(3*kk, 3*(kk+1)):
                    for c in range(3*ii, 3*(ii+1)):
                        if r == k and c == i: continue
                        if ans[r][c] > 0: continue
##                        set_cand(r, c)
                        if n in cand[r][c]:
                            flag = True
                            break
                if not flag:
                    ans[k][i] = n
                    change_cand(k, i)
                    FLAG = True
    return FLAG

def button2_clicked():
    StandardPlay()
    ShowBan()

def RegularPlay():
    while True:
        while True:
            F = SimplePlay()
            if not F:
                break
        F = StandardPlay()
        if not F:
            break

def completeP():
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0: return False
    for k in range(9):
        for n in range(1, 10):
            flag = False
            for i in range(9):
                if ans[k][i] == n:
                    flag = True
                    break
            if not flag:
                return False
    for i in range(9):
```

```
        for n in range(1, 10):
            flag = False
            for k in range(9):
                if ans[k][i] == n:
                    flag = True
                    break
            if not flag:
                return False
    for kk in range(3):
        for ii in range(3):
            for n in range(1, 10):
                flag = False
                for k in range(3*kk, 3*(kk+1)):
                    for i in range(3*ii, 3*(ii+1)):
                        if ans[k][i] == n:
                            flag = True
                            break
                if not flag:
                    return False
    return True

def losingP():
    global cand
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
##                set_cand(k, i)
                if len(cand[k][i]) == 0:
                    return True
    for k in range(9):
        P = [0]*10
        for i in range(9):
            P[ans[k][i]] = P[ans[k][i]] + 1
        for n in range(1, 10):
            if P[n] > 1:
                return True
    for i in range(9):
        P = [0]*10
        for k in range(9):
            P[ans[k][i]] = P[ans[k][i]] + 1
        for n in range(1, 10):
            if P[n] > 1:
                return True
```

```
    for kk in range(3):
        for ii in range(3):
            P = [0]*10
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    P[ans[k][i]] = P[ans[k][i]] + 1
            for n in range(1, 10):
                if P[n] > 1:
                    return True
    return False

##n_solve = 0

def solver():
    global ans, cand
    RegularPlay()
    if losingP():
        return False
    if completeP():
        return True
    flag = False
    for k in range(9):
        for i in range(9):
            if ans[k][i] == 0:
                kk = k
                ii = i
                flag = True
                break
        if flag:
            break
    tempAns = copy.deepcopy(ans)
    tempCand = copy.deepcopy(cand)
    set_cand(kk, ii)
    S = cand[kk][ii].copy()
    for n in S:
        tempCand2 = copy.deepcopy(cand)
        ans[kk][ii] = n
        change_cand(kk, ii)
        if solver():
            return True
        ans = copy.deepcopy(tempAns)
        cand = copy.deepcopy(tempCand2)
        ans[kk][ii] = 0
```

```
    return False

def button3_clicked():
    global ans, cand
##    global n_solve
    RegularPlay()
##    n_solve = 0
    solver()
    ShowBan()
def button4_clicked():
    global ans, cand
    # 行の TWINS
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
##            set_cand(k, i)
            if len(cand[k][i]) == 2:
               for j in range(i+1, 9):
                   if ans[k][j]>0: continue
##                   set_cand(k, j)
                   if cand[k][i] == cand[k][j]:
                       for p in range(9):
                           if p == i or p == j: continue
                           if ans[k][p]>0: continue
##                           set_cand(k, p)
                           cand[k][p] = cand[k][p] - cand[k][i]
                           if len(cand[k][p]) == 1:
                               ans[k][p] = cand[k][p].pop()
                               change_cand(k, p)
    # 列の TWINS
    for i in range(9):
        for k in range(9):
            if ans[k][i]>0: continue
##            set_cand(k, i)
            if len(cand[k][i]) == 2:
               for j in range(k+1, 9):
                   if ans[j][i]>0: continue
##                   set_cand(j, i)
                   if cand[k][i] == cand[j][i]:
                       for p in range(9):
                           if p == k or p == j: continue
                           if ans[p][i]>0: continue
##                           set_cand(p, i)
```

```
                            cand[p][i] = cand[p][i] - cand[k][i]
                            if len(cand[p][i]) == 1:
                                ans[p][i] = cand[p][i].pop()
                                change_cand(p, i)
    # ブロックの TWINS
    for kk in range(3):
        for ii in range(3):
            for k in range(3*kk, 3*(kk+1)):
                for i in range(3*ii, 3*(ii+1)):
                    if ans[k][i] > 0: continue
##                    set_cand(k, i)
                    if len(cand[k][i]) == 2:
                        for p in range(3*kk, 3*(kk+1)):
                            for q in range(3*ii, 3*(ii+1)):
                                if p<k or p ==k and q<=i: continue
                                if ans[p][q] > 0: continue
##                                  set_cand(p, q)
                                if cand[k][i] == cand[p][q]:
                                   for r in range(3*kk, 3*(kk+1)):
                                        for s in range(3*ii, 3*(ii+1)):
                                            if (r==k and s==i)or(r==p and s==q):
                                                continue
                                            if ans[r][s] > 0: continue
##                                              set_cand(r, s)
                                            cand[r][s] = cand[r][s] - cand[k][i]
                                            if len(cand[r][s]) == 1:
                                                ans[r][s] = cand[r][s].pop()
                                                change_cand(r, s)
    ShowBan()
def button5_clicked():
    global ans, cand
    # 行の X-WINGS
    for n in range(1,10):
        for k in range(9):
            index = []
            for i in range(9):
                if ans[k][i] > 0: continue;
                if n in cand[k][i]:
                    index.append(i)
            if len(index) == 2:
                for j in range(k+1, 9):
                    index2 = []
                    for p in range(9):
```

```
                    if ans[j][p] > 0: continue
                    if n in cand[j][p]:
                        index2.append(p)
                if len(index2) == 2:
                    if index[0]==index2[0] and index[1]==index2[1]:
                        for q in range(9):
                            if q==k or q==j: continue
                            if ans[q][index[0]] > 0: continue
                            cand[q][index[0]] = cand[q][index[0]] - {n}
                            if len(cand[q][index[0]]) == 1:
                                ans[q][index[0]] = cand[q][index[0]].pop()
                                change_cand(q, index[0])
                        for q in range(9):
                            if q==k or q==j: continue
                            if ans[q][index[1]] > 0: continue
                            cand[q][index[1]] = cand[q][index[1]] - {n}
                            if len(cand[q][index[1]]) == 1:
                                ans[q][index[1]] = cand[q][index[1]].pop()
                                change_cand(q, index[1])
# 列の X-WINGS
for n in range(1,10):
    for i in range(9):
        index = []
        for k in range(9):
            if ans[k][i] > 0: continue;
            if n in cand[k][i]:
                index.append(k)
        if len(index) == 2:
            for j in range(i+1, 9):
                index2 = []
                for p in range(9):
                    if ans[p][j] > 0: continue
                    if n in cand[p][j]:
                        index2.append(p)
                if len(index2) == 2:
                    if index[0]==index2[0] and index[1]==index2[1]:
                        for q in range(9):
                            if q==i or q==j: continue
                            if ans[index[0]][q] > 0: continue
                            cand[index[0]][q] = cand[index[0]][q] - {n}
                            if len(cand[index[0]][q]) == 1:
                                ans[index[0]][q] = cand[index[0]][q].pop()
                                change_cand(index[0], q)
```

```
                            for q in range(9):
                                if q==i or q==j: continue
                                if ans[index[1]][q] > 0: continue
                                cand[index[1]][q] = cand[index[1]][q] - {n}
                                if len(cand[index[1]][q]) == 1:
                                    ans[index[1]][q] = cand[index[1]][q].pop()
                                    change_cand(index[1], q)
    ShowBan()
def button6_clicked():
    global ans, cand
    # 行の TRIPLETS
    for k in range(9):
        for i in range(9):
            if ans[k][i] > 0: continue
            if len(cand[k][i]) == 3:
                index = [i]
                for j in range(9):
                   if j == i: continue
                   if ans[k][j]>0: continue
                   if cand[k][j] <= cand[k][i]:
                       index.append(j)
                if len(index) == 3:
                    for p in range(9):
                        if p == i or p == index[1] or p == index[2]: continue
                        if ans[k][p]>0: continue
                        cand[k][p] = cand[k][p] - cand[k][i]
                        if len(cand[k][p]) == 1:
                            ans[k][p] = cand[k][p].pop()
                            change_cand(k, p)
    for k in range(9):
        for i in range(9):
            if ans[k][i]>0: continue
            if len(cand[k][i]) == 2:
                 for j in range(i+1, 9):
                   if ans[k][j]>0: continue
                   if len(cand[k][j]) == 2:
                        if len(cand[k][i] & cand[k][j]) == 1:
                           S = cand[k][i] | cand[k][j]
                           index = [i, j]
                           for p in range(j+1, 9):
                               if ans[k][p] > 0: continue
                               if cand[k][p] <= S:
                                   index.append(p)
```

```
                        if len(index) == 3:
                           for p in range(9):
                             if p == i or p == j or p == index[2]: continue
                             if ans[k][p]>0: continue
                             cand[k][p] = cand[k][p] - S
                             if len(cand[k][p]) == 1:
                                 ans[k][p] = cand[k][p].pop()
                                 change_cand(k, p)
# 列の TRIPLETS
for i in range(9):
    for k in range(9):
        if ans[k][i]>0: continue
        if len(cand[k][i]) == 3:
            index = [k]
            for j in range(9):
               if j == k: continue
               if ans[j][i]>0: continue
               if cand[j][i] <= cand[k][i]:
                   index.append(j)
            if len(index) == 3:
                for p in range(9):
                    if p == k or p == index[1] or p == index[2]: continue
                    if ans[p][i]>0: continue
                    cand[p][i] = cand[p][i] - cand[k][i]
                    if len(cand[p][i]) == 1:
                        ans[p][i] = cand[p][i].pop()
                        change_cand(p, i)
for i in range(9):
    for k in range(9):
        if ans[k][i]>0: continue
        if len(cand[k][i]) == 2:
             for j in range(k+1, 9):
               if ans[j][i]>0: continue
               if len(cand[j][i]) == 2:
                    if len(cand[k][i] & cand[j][i]) == 1:
                       S = cand[k][i] | cand[j][i]
                       index = [k, j]
                       for p in range(j+1, 9):
                           if ans[p][i]>0: continue
                           if cand[p][i] <= S:
                               index.append(p)
                       if len(index) == 3:
                            for p in range(9):
```

```
                                if p == k or p == j or p == index[2]:
                                    continue
                                if ans[p][i]>0: continue
                                cand[p][i] = cand[p][i] - S
                                if len(cand[p][i]) == 1:
                                    ans[p][i] = cand[p][i].pop()
                                    change_cand(p, i)
# ブロックの TRIPLETS
for kk in range(3):
    for ii in range(3):
        for k in range(3*kk, 3*(kk+1)):
            for i in range(3*ii, 3*(ii+1)):
                if ans[k][i] > 0: continue
                if len(cand[k][i]) == 3:
                    index = [(k, i)]
                    for p in range(3*kk, 3*(kk+1)):
                        for q in range(3*ii, 3*(ii+1)):
                            if p == k and q == i: continue
                            if ans[p][q] > 0: continue
                            if cand[p][q] <= cand[k][i]:
                                index.append((p, q))
                    if len(index) == 3:
                        for r in range(3*kk, 3*(kk+1)):
                            for s in range(3*ii, 3*(ii+1)):
                                if (r,s)==(k,i) or(r,s)==index[1] or \
                                   (r,s)==index[2]:
                                            continue
                                if ans[r][s] > 0: continue
                                cand[r][s] = cand[r][s] - cand[k][i]
                                if len(cand[r][s]) == 1:
                                    ans[r][s] = cand[r][s].pop()
                                    change_cand(r, s)
for kk in range(3):
    for ii in range(3):
        for k in range(3*kk, 3*(kk+1)):
            for i in range(3*ii, 3*(ii+1)):
                if ans[k][i] > 0: continue
                if len(cand[k][i]) == 2:
                    for p in range(3*kk, 3*(kk+1)):
                        for q in range(3*ii, 3*(ii+1)):
                            if p<k or p ==k and q<=i: continue
                            if ans[p][q] > 0: continue
                            if len(cand[p][q]) == 2:
```

```
                                    if len(cand[k][i] & cand[p][q]) == 1:
                                        S = cand[k][i] | cand[p][q]
                                        index=[(k,i),(p, q)]
                                        for r in range(3*kk, 3*(kk+1)):
                                            for s in range(3*ii, 3*(ii+1)):
                                               if r<p or r ==p and s<=q:
                                                   continue
                                               if ans[r][s] > 0: continue
                                               if cand[r][s] <= S:
                                                   index.append((r,s))
                                        if len(index) == 3:
                                            for r in range(3*kk, 3*(kk+1)):
                                                for s in range(3*ii, 3*(ii+1)):
                                                    if (r,s)==(k,i) or \
                                                       (r,s)==(p,q) or \
                                                       (r,s)==index[2]:
                                                        continue
                                                    if ans[r][s] > 0: continue
                                                    cand[r][s] = cand[r][s] - S
                                                    if len(cand[r][s]) == 1:
                                                        ans[r][s]=cand[r][s].pop()
                                                        change_cand(r, s)
    ShowBan()

root = Tk()
val = IntVar()
val.set(0)

canvas = Canvas(root, width=board_size, height=board_size)
canvas.pack()
menu_ROOT = Menu(root)
root.configure(menu=menu_ROOT)
menu_FILE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="FILE", menu=menu_FILE)
menu_FILE.add_command(label='SaveAs', command=save_ban)
menu_FILE.add_command(label='Open', command=open_ban)
menu_FILE.add_separator()
menu_FILE.add_command(label='Print', command=print_ban)

menu_SAMPLE = Menu(menu_ROOT)
menu_ROOT.add_cascade(label="SAMPLE", menu=menu_SAMPLE)
menu_SAMPLE.add_command(label='Sample1', command=fun_sample1)
menu_SAMPLE.add_command(label='Sample2', command=fun_sample2)
```

```
menu_SAMPLE.add_command(label='Sample3', command=fun_sample3)
menu_SAMPLE.add_command(label='Sample4', command=fun_sample4)
menu_SAMPLE.add_command(label='Sample5', command=fun_sample5)
menu_SAMPLE.add_command(label='Sample6', command=fun_sample6)

canvas.bind("<ButtonPress-1>", buttonPress)

r0 = Radiobutton(root, text = 'ban[][]', variable = val, value = 0)
r0.pack()
r1 = Radiobutton(root, text = 'ans[][]', variable = val, value = 1)
r1.pack()

button1 = Button(root, text='セルに候補が1個', command=button1_clicked)
button1.pack(side='left')
button2 = Button(root, text='ブロック･列･行に候補が1個', command=button2_clicked)
button2.pack(side='left')
button3 = Button(root, text='虱潰し探索', command=button3_clicked)
button3.pack(side='left')
button4 = Button(root, text='TWINS', command=button4_clicked)
button4.pack(side='left')
button5 = Button(root, text='X-WINGS', command=button5_clicked)
button5.pack(side='left')
button6 = Button(root, text='TRIPLETS', command=button6_clicked)
button6.pack(side='left')

root.mainloop()
```

となります。

何とか VC++で作ったプログラムと同等以上のものを作りましたが、やはりまとまった書籍の情報がある VC++ の方が楽です。tkinter の情報はインターネットが頼りで、気長に情報を探せば、それを使って、 Python でもこの様にプログラミングすることは可能です。一度作れば、後は、これを雛形に他の鉛筆パズルのプログラムも作れます。興味があれば、それぞれの命令の意味はインターネットで調べて下さい。印刷のプログラミングは Python 3.6 の方が Python 2.7 より楽になりました。親切な人たち（日本人だけではないですが、殆どが英語なので何とかなります）が必要な情報を書き込んでくれています。皆、失敗を繰り返しながら、プログラミングを学びます。あらゆる失敗を経験するとプログラミングできるようになると言う人もいます。

1917/11/3 に Srini Devadas 著「Programming for the Puzzled : Learn to Program While Solving Puzzled (The MIT Press)」が出版されました。youtube でその講義を見ることもできます。14 You Won't Want to Play Sudoku Again に数独のバックトラキングの Python のプログラムが載っています。上に書いた数独のバックトラキングのプログラムは間違ってはいないけど、効率が悪いことに気付きます。「8 人の女王の問題」で、バックトラキングのやり方を学んでいた筈なのに、数独のバックトラキングでも同じことをすれば良いことをすっかり忘れていました。折角、苦労して学んだ技術が身についていませんでした。上のバックトラキングのプログラムのよう

に、間違っていないプログラムは、最初に思いついたアルゴリズムで、デバッグを繰り返していれば、誰でも作り上げることが出来ます。しかし、プログラミングの世界では実行時間の短縮が最大の問題です。色々なプログラムを書いて、時間を測って、どれが本当に良いプログラムであるか、チェックすることを怠っていました。良いプログラムを作るのは、時間のかかる大変な作業です。多分、素人にとってはこの余計とも思われることが出来るかどうかが、プロとアマチュアの差です。私のような本来数学者で、プログラミングを数学の研究のために独学で学んできたものは、兎も角正しく動くプログラムができれば安心して、更に良いアルゴリズムがないかを探求する努力を放棄していました。最初から数独のバックトラキングのプログラムだけを考えていれば、このような失敗はしなかったかもわかりませんが、数独のヒントを表示することから始めたので、プログラムが大きくなり、プログラムを見直すことが大変になり、大事なチェックを怠っていました。本物の技術を学ぶためには、本物の技術を持っている人のもとで学ばなくてはいけません。Srini Devadas 著「Programming for the Puzzled : Learn to Program While Solving Puzzled (The MIT Press)」は良い本だと思います。色々なことに気付かせてくれる本です。ぜひ読んでみてください。

「ひとりにしてくれ」の解法解析の文書も株式会社ニコリの許可が下りましたので、同じところにアップしています。鉛筆パズルに興味が出て来たなら、それも参照してください。