高知大学教育学部の情報数学のテキスト：おまけ

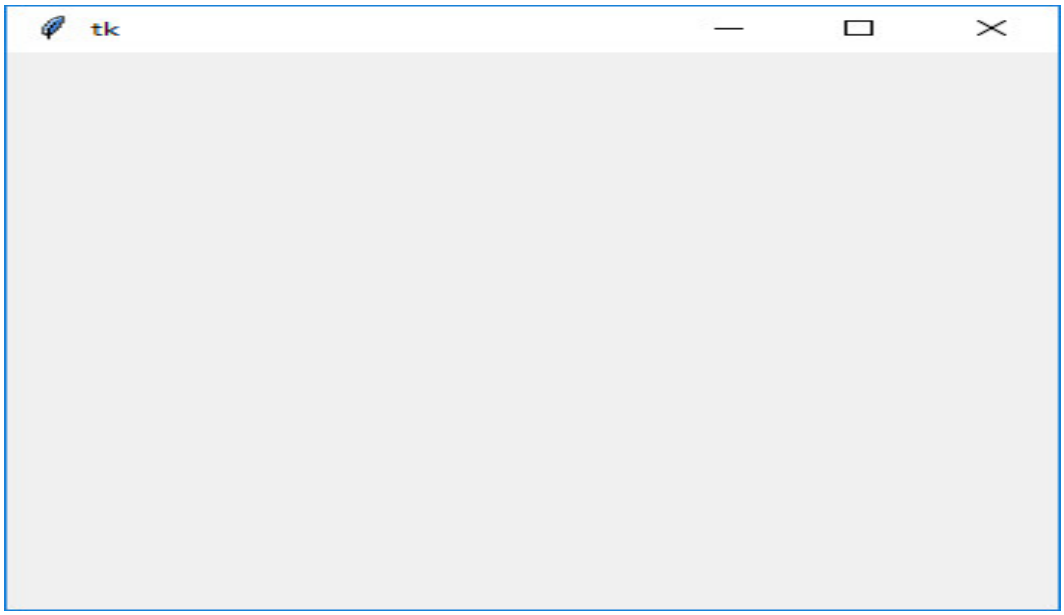
文責　：　高知大学名誉教授　中村　治

＜囲碁の定石を覚えるためのプログラム＞

昔は勝っていた若い人に囲碁で負けるようになりました。仕方がないので囲碁ソフト：天頂の囲碁 6 Zen（趙治勲さんと対戦した Deep Zen Go より 3 子ぐらい弱いそうです）で遊んでみると定石を知らないことをとがめられます。私は元々丸暗記が苦手の上に、老人になって益々囲碁の定石を覚えることが困難になってきました。本当は私はこの年になるまで暗記の方法を知らなかったということで、短期記憶が長期記憶に変化するまで、忘れる前に、時間を置かず、何度でも覚えなおせば良いだけの話だそうです。そこで、囲碁の定石を覚えるためのプログラムを作ってみました。私の講義を受けている学生さんたちの参考になればと思って、素人のプログラマーが、最終の設計図のイメージなしに、作っては考え作っては考え行き当たりばったりにプログラムを作ったその過程が分かるように、プログラミングしながらこの文章を書いています。そのようなことに興味がない方は最後の最終的なプログラムをご覧ください。

```
from tkinter import *
root = Tk()
canvas = Canvas(root, width = 360, height=360)
canvas.pack()
root.mainloop()
```

が基本パターンである。



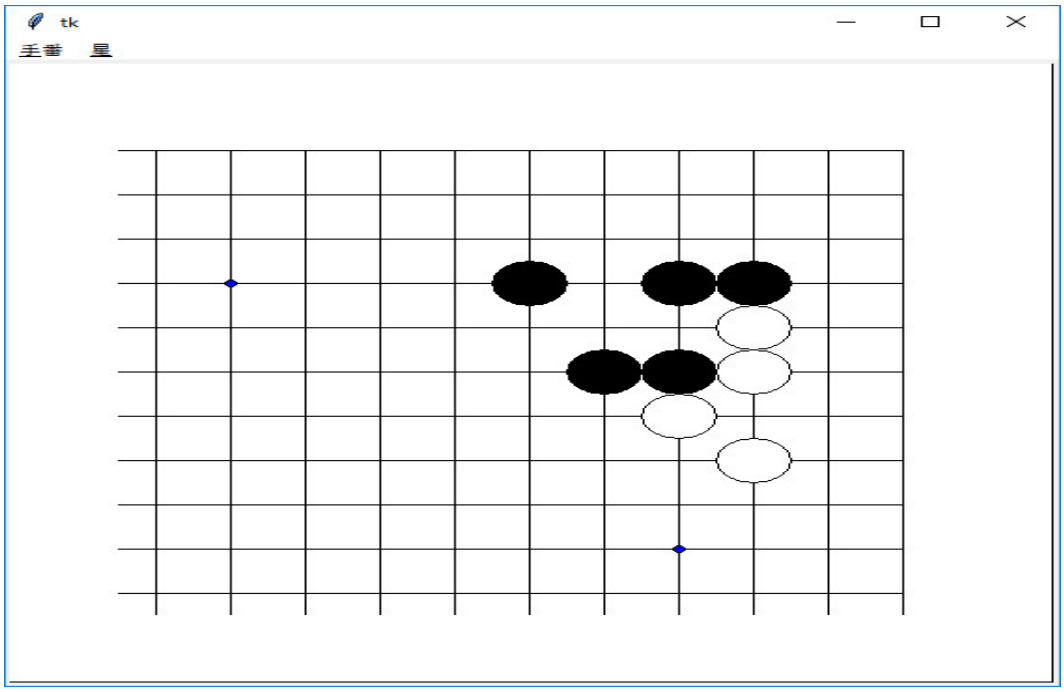
これにメニューを追加する。メニューの基本的使い方は以下のようになる。

```
from tkinter import *

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

# ダミー
def start():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')

    canvas.create_oval(360-20, 200-20, 360+20, 200+20, fill='black')
    canvas.create_oval(400-20, 280-20, 400+20, 280+20, fill='white')
    canvas.create_oval(360-20, 280-20, 360+20, 280+20, fill='black')
    canvas.create_oval(360-20, 320-20, 360+20, 320+20, fill='white')
```

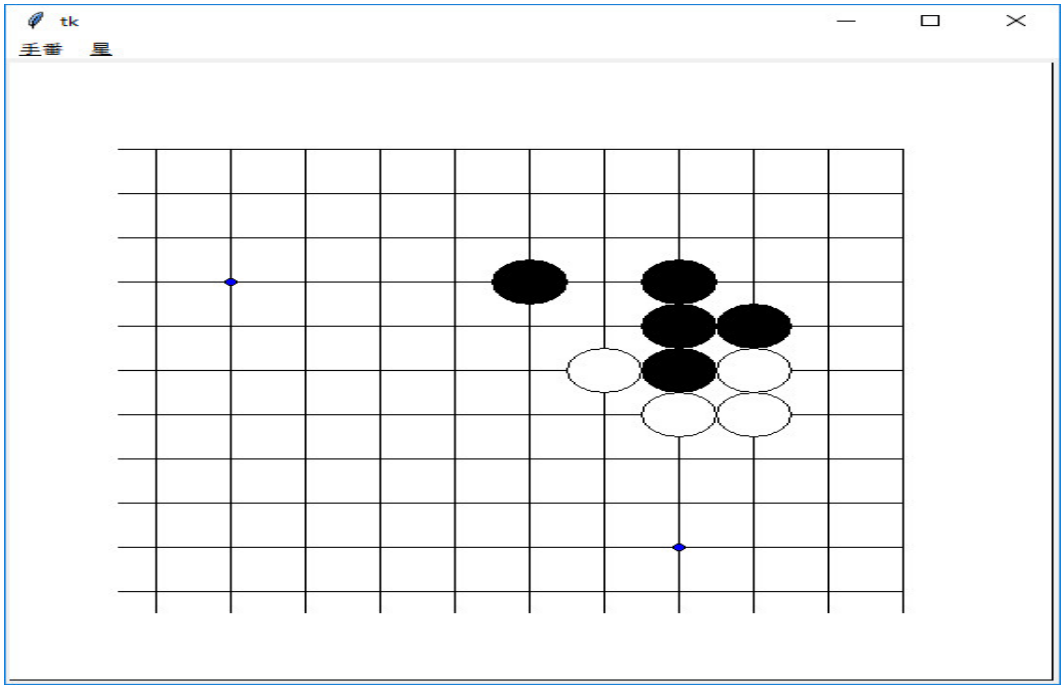
```
    canvas.create_oval(320-20, 280-20, 320+20, 280+20, fill='black')
    canvas.create_oval(400-20, 240-20, 400+20, 240+20, fill='white')
    canvas.create_oval(400-20, 200-20, 400+20, 200+20, fill='black')
    canvas.create_oval(400-20, 360-20, 400+20, 360+20, fill='white')
    canvas.create_oval(280-20, 200-20, 280+20, 200+20, fill='black')

def start2():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')

    if teban.get() == 1:
        canvas.create_oval(360-20, 200-20, 360+20, 200+20, fill='black')
        canvas.create_oval(400-20, 280-20, 400+20, 280+20, fill='white')
        canvas.create_oval(360-20, 280-20, 360+20, 280+20, fill='black')
        canvas.create_oval(360-20, 320-20, 360+20, 320+20, fill='white')
        canvas.create_oval(400-20, 240-20, 400+20, 240+20, fill='black')
        canvas.create_oval(320-20, 280-20, 320+20, 280+20, fill='white')
        canvas.create_oval(360-20, 240-20, 360+20, 240+20, fill='black')
        canvas.create_oval(400-20, 320-20, 400+20, 320+20, fill='white')
        canvas.create_oval(280-20, 200-20, 280+20, 200+20, fill='black')
    else:
        canvas.create_oval(360-20, 200-20, 360+20, 200+20, fill='black')
        canvas.create_oval(400-20, 280-20, 400+20, 280+20, fill='white')
        canvas.create_oval(360-20, 280-20, 360+20, 280+20, fill='black')
        canvas.create_oval(360-20, 320-20, 360+20, 320+20, fill='white')
        canvas.create_oval(400-20, 240-20, 400+20, 240+20, fill='black')
        canvas.create_oval(320-20, 280-20, 320+20, 280+20, fill='white')
        canvas.create_oval(360-20, 240-20, 360+20, 240+20, fill='black')
        canvas.create_oval(320-20, 320-20, 320+20, 320+20, fill='white')
        canvas.create_oval(400-20, 320-20, 400+20, 320+20, fill='black')
        canvas.create_oval(400-20, 360-20, 400+20, 360+20, fill='white')
        canvas.create_oval(440-20, 280-20, 440+20, 280+20, fill='black')
        canvas.create_oval(440-20, 320-20, 440+20, 320+20, fill='white')
```

```
        canvas.create_oval(280-20, 200-20, 280+20, 200+20, fill='black')


# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)

# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)

# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()

root.mainloop()
```

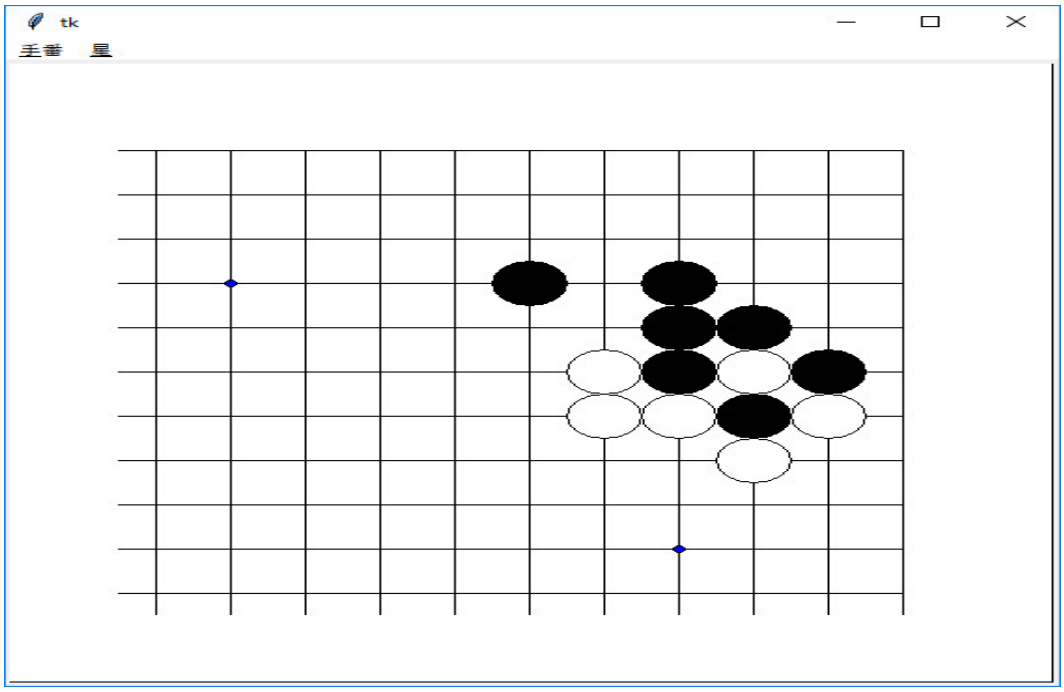
メニューの「星」の「小ゲイマガカリ」の「ツケノビ」をクリックすると



で、「星」の「小ゲイマガカリ」の「ツケオサエ」をクリックするとデフォルトでは teban=1 ですから



となり、メニューの「手番」の「後手」をクリックしてから、「星」の「小ゲイマガカリ」の「ツケオサエ」をクリックすると teban=2 と変化したので



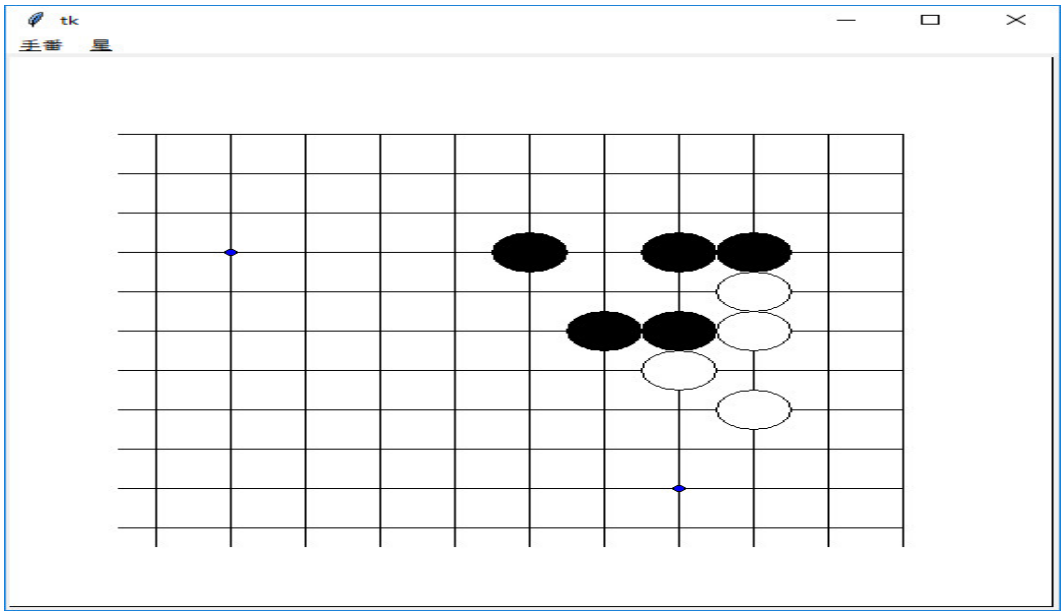
となります。それぞれの基本定石を表示しています。

定石を覚えるためには、定石をアニメーションで見せるようにする方法と先手又は後手をコンピュータが受け持ち、ユーザーが定石の手順通り打つかどうかシミュレーションする方法があります。ここでは後者の方法をとることにします。メニューの「手番」で「先手」を打つか「後手」を打つか決めます。他のメニューでどの定石をシミュレーションするか決めます。その後、マウスで手を選びます。

```
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py

def on_pressed(event):
    sx, sy = get_pos(event.x, event.y)
    if teban.get() == 1:
        canvas.create_oval(sx-20, sy-20, sx+20, sy+20, fill='black')
    else:
        canvas.create_oval(sx-20, sy-20, sx+20, sy+20, fill='white')

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

を追加するとマウスで石を追加できます。

次のステップは



のような定石をどのようなデータ構造で表せばよいか考えることです。
　例えば、

```
[(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,8,'w'),(14,4,'b')]
```

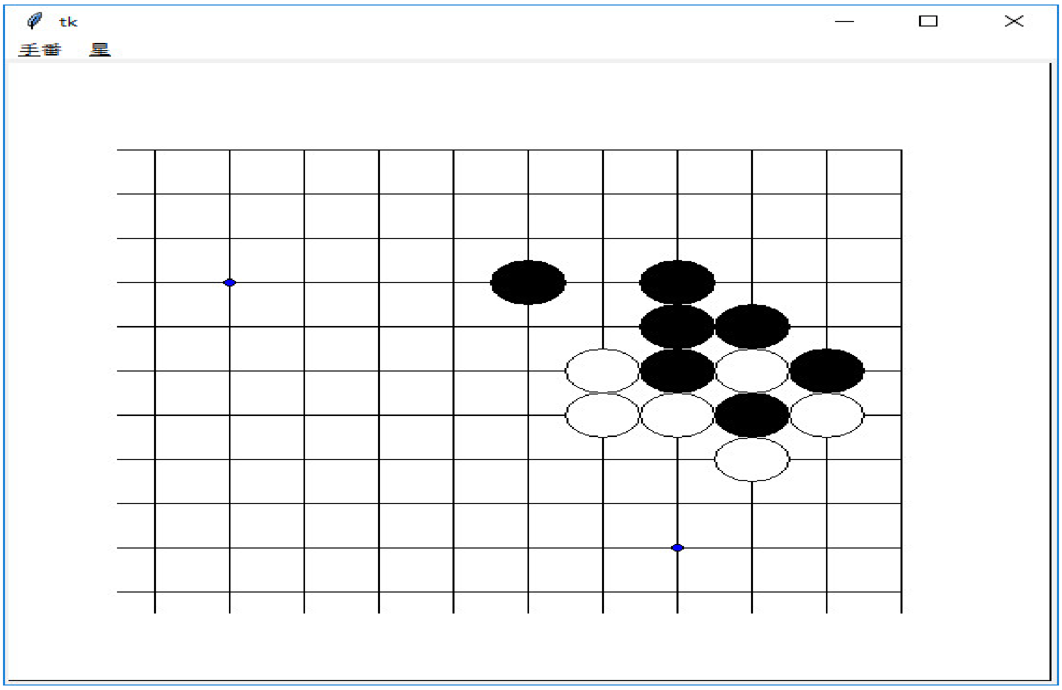
のように表せます。

```
from tkinter import *

root = Tk()


# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0

# ダミー
def start():
    global joseki, num
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
```

```
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')

    joseki = hosi1
    num = 0
    if teban.get() == 2:
        tx, ty, cl = joseki[num]
        px = 40*(tx-9)+80
        py = 40*(ty-1)+80
        canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
        num += 1

def start2(): pass

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)

# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)

# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()

def get_pos(x, y):
```

```
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py

def on_pressed(event):
    global num
    sx, sy = get_pos(event.x, event.y)
    tx, ty, cl = joseki[num]
    px = 40*(tx-9)+80
    py = 40*(ty-1)+80
    print('sx=',sx,'sy=',sy,'px=',px,'py=',py)
    if sx != px or sy != py: return
    if teban.get() == 1 and cl == 'b':
        canvas.create_oval(sx-20, sy-20, sx+20, sy+20, fill='black')
    elif teban.get() == 2 and cl == 'w':
        canvas.create_oval(sx-20, sy-20, sx+20, sy+20, fill='white')
    else: return
    num += 1
    if num >= len(joseki):
        canvas.create_text(10, 10, text = 'OK', font = ('FixedSys', 14))
        return
    tx, ty, cl = joseki[num]
    px = 40*(tx-9)+80
    py = 40*(ty-1)+80
    if teban.get() == 1:
        canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')
    elif teban.get() == 2:
        canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
    num += 1
    if num >= len(joseki):
        canvas.create_text(10, 10, text = 'OK', font = ('FixedSys', 14))

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

# 定石
hosi1 = [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,8,'w'),(14,4,'b')]

root.mainloop()
```

で、一様、星の定石のツケノビ定石の基本形をシミュレーションできますが、



のように、石を取ることもあるので、盤面を記憶しておくことが必要です。

```
from tkinter import *

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
```

```
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
```

```
                                            p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)
def display_board():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
```

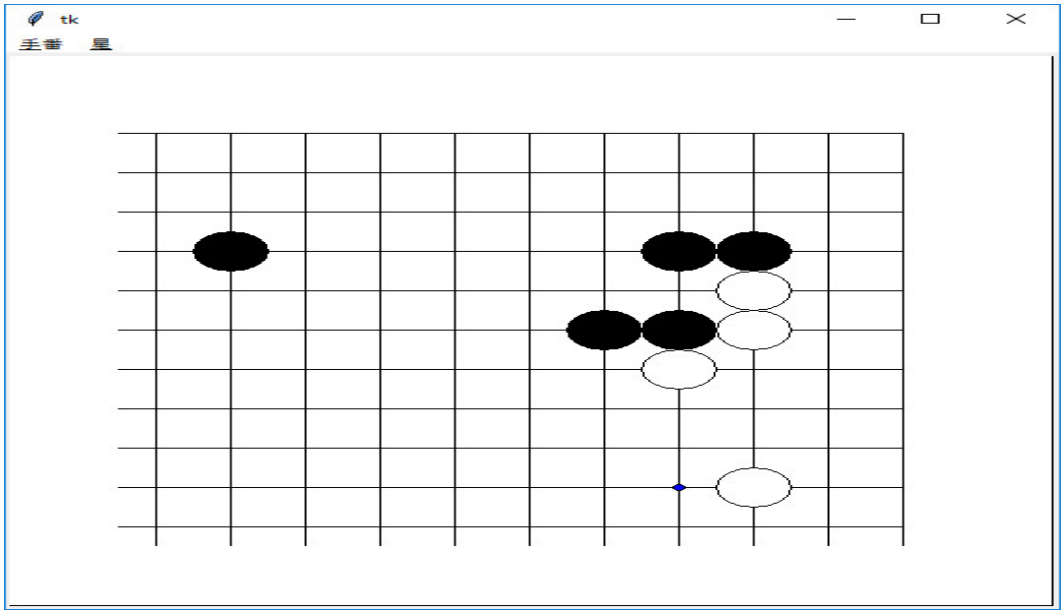
```
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')

# ダミー
def start():
    global joseki, num, board
    joseki = hosi1
    num = 0
    board = basic_board.copy()
    if teban.get() == 2:
        tx, ty, cl = joseki[num]
        put_stone(get_z(tx, ty), get_col(cl))
        num += 1
    display_board()

def start2(): pass

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)

# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)

# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

### ラベル
##Label(root, text = "***** Menu Test *****").pack()

canvas = Canvas(root, width = 560, height=560)
canvas.pack()
```

```
def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py

def on_pressed(event):
    global num, board
    sx, sy = get_pos(event.x, event.y)
    tx, ty, cl = joseki[num]
    px = 40*(tx-9)+80
    py = 40*(ty-1)+80
    print('sx=',sx,'sy=',sy,'px=',px,'py=',py)
    if sx != px or sy != py:
        canvas.create_text(200, 10, text = 'Failure!',
                                font = ('FixedSys', 20))
        return
    if teban.get() == 1 and cl == 'b':
        put_stone(get_z(tx, ty), get_col(cl))
    elif teban.get() == 2 and cl == 'w':
        put_stone(get_z(tx, ty), get_col(cl))
    else: return
    num += 1
    if num >= len(joseki):
        display_board()
        canvas.create_text(200, 10, text = 'Congratulations!',
                                font = ('FixedSys', 20))
        return
    tx, ty, cl = joseki[num]
    if teban.get() == 1:
        put_stone(get_z(tx, ty), get_col(cl))
    elif teban.get() == 2:
        put_stone(get_z(tx, ty), get_col(cl))
    display_board()
    num += 1
    if num >= len(joseki):
        canvas.create_text(200, 10, text = 'Congratulations!',
```

```
                                font = ('FixedSys', 20))

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

# 定石
hosi1 = [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,8,'w'),(14,4,'b')]

root.mainloop()
```

のように修正します。def put_stone(tz, color): は正しい手が打たれるという前提で書いています。

更に、



のような定石もあるし、変化形が色々あります。従って、例えば、

```
[(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,10,'w'),(10,4,'b')]
```

というデータも必要です。この二つのデータを統合して、木構造で表現することも可能ですが、ここでは簡単に、

```
[[(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,8,'w'),(14,4,'b')],
[(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,10,'w'),(10,3,'b')]]
```

のように、リストのリストで表すことにします。変化形があるごとに、そのリストを追加することにします。

次に、このような変化形にも対応できるようにプログラムを修正します。手順を覚えていく history という変数と定石の次の候補を見つける関数 `get_candidate()` を追加します。

```
from tkinter import *
import random

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0
history = []

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
```

```
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
```

```
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)
def display_board():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')
def get_candidate(num):
    candidate = []
    for j_list in joseki:
        flag = True
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                if j_list[num] not in candidate:
```

```
                    candidate.append(j_list[num])
    return candidate
def start():
    global joseki, num, board, history
    random.seed()
    joseki = hosi1
    num = 0
    board = basic_board.copy()
    history = []
    if teban.get() == 2:
        candidate = get_candidate(num)
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        num += 1
    display_board()

def start2(): pass

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()
```

```
def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history
    sx, sy = get_pos(event.x, event.y)
    candidate = get_candidate(num)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if teban.get() == 1 else 'w'
    print('tx=',tx,'ty=',ty,'cl=',cl)
    print('candidate=', candidate)
    if is_candidate(tx, ty, cl,candidate) == False:
        canvas.create_text(200, 10, text = 'Failure!',
                                font = ('FixedSys', 20))
        return
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    print('num=',num)
    candidate = get_candidate(num)
    if candidate == []:
        display_board()
        canvas.create_text(200, 10, text = 'Congratulations!',
                                font = ('FixedSys', 20))
        return
    n = random.randint(0, len(candidate)-1)
    tx, ty, cl = candidate[n]
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    display_board()
```

```
    num += 1
    candidate = get_candidate(num)
    if candidate == []:
        canvas.create_text(200, 10, text = 'Congratulations!',
                           font = ('FixedSys', 20))

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

# 定石
hosi1 = [[(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,8,'w'),(14,4,'b')],
[(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
(17,4,'b'),(17,10,'w'),(10,3,'b')]]

root.mainloop()
```

で、上手くいっているみたいです。定石を増やし、先手用と後手用の定石集を切り替えるようにします。

```
from tkinter import *
import random

root = Tk()
##root.option_add("*font", ("FixedSys", 10))

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0
history = []

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
```

```
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                    p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
```

```
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)


def display_board():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
```

```
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')
def get_candidate(num):
    candidate = []
    for j_list in joseki:
        flag = True
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                if j_list[num] not in candidate:
                    candidate.append(j_list[num])
    return candidate
def basic_start():
    global joseki, num, board, history
    num = 0
    board = basic_board.copy()
    history = []
    if teban.get() == 2:
        candidate = get_candidate(num)
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        num += 1
    display_board()

def start():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi1B
    else:
        joseki = hosi1W
```

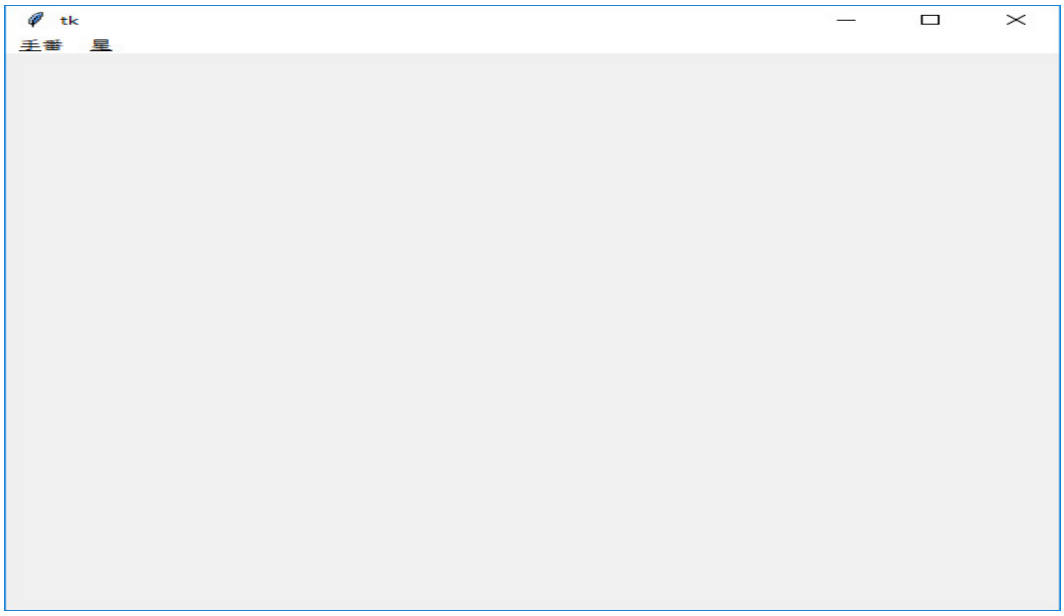
```
    basic_start()

def start2():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi2B
    else:
        joseki = hosi2B
    basic_start()

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()

def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
```

```
            return True
    return False
def on_pressed(event):
    global num, board, history
    sx, sy = get_pos(event.x, event.y)
    candidate = get_candidate(num)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if teban.get() == 1 else 'w'
    if is_candidate(tx, ty, cl,candidate) == False:
        canvas.create_text(200, 10, text = 'Failure!',
                           font = ('FixedSys', 20))
        return
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    candidate = get_candidate(num)
    if candidate == []:
        display_board()
        canvas.create_text(200, 10, text = 'Congratulations!',
                           font = ('FixedSys', 20))
        return
    n = random.randint(0, len(candidate)-1)
    tx, ty, cl = candidate[n]
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    display_board()
    num += 1
    candidate = get_candidate(num)
    if candidate == []:
        canvas.create_text(200, 10, text = 'Congratulations!',
                           font = ('FixedSys', 20))

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

# 定石
##hosi1 = [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
##(17,4,'b'),(17,8,'w'),(14,4,'b')]
```

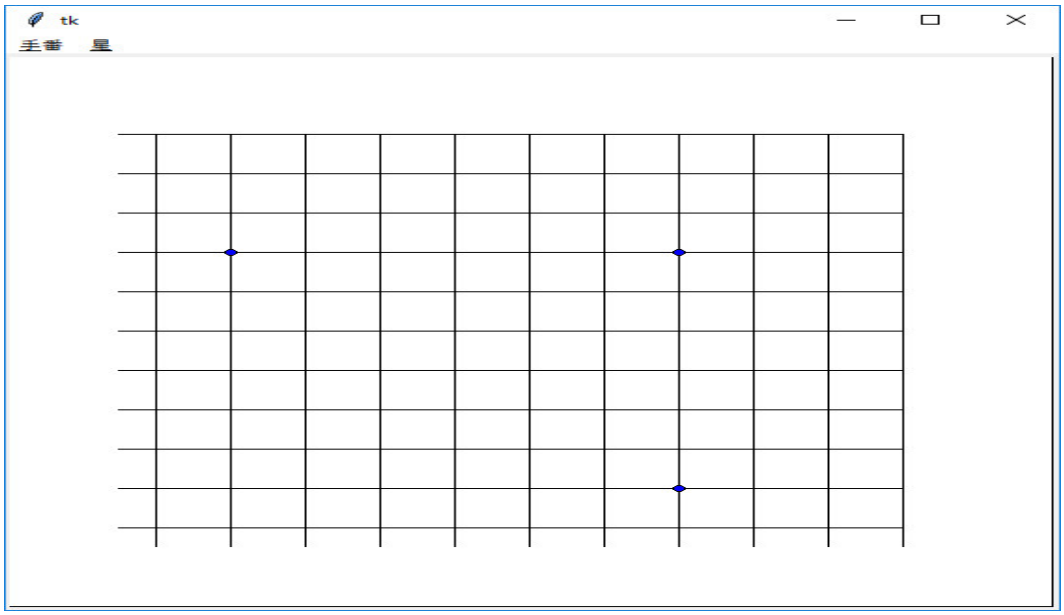
```
hosi1B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(18,2,'w'),
     (17,2,'b'),(17,1,'w'),(15,2,'b'),(16,3,'w'),(17,3,'b'),(16,1,'w'),
     (18,1,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(17,2,'w'),
     (18,2,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(17,7,'b'),(18,7,'w'),
     (17,8,'b'),(18,8,'w'),(14,2,'b'),(18,2,'w'),(17,2,'b'),(17,3,'w'),
     (16,3,'b'),(15,2,'w'),(18,3,'b'),(17,1,'w'),(15,1,'b'),(17,3,'w'),
     (19,5,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(15,2,'w'),(14,2,'b'),(17,2,'w'),
     (17,7,'b'),(18,7,'w'),(17,8,'b'),(18,8,'w'),(18,2,'b'),(18,1,'w'),
     (19,5,'b'),(18,6,'w'),(17,3,'b'),(16,2,'w'),(19,2,'b')],

    ]
hosi1W = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')]
    ]
hosi2B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(17,7,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(18,7,'w'),
     (14,4,'b')],
```

```
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(14,4,'w'),
     (18,8,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(18,6,'w'),(15,7,'b'),(16,8,'w'),(14,6,'b'),(15,8,'w'),
     (15,5,'b'),(18,5,'w'),(17,3,'b')]
    ]

root.mainloop()
```

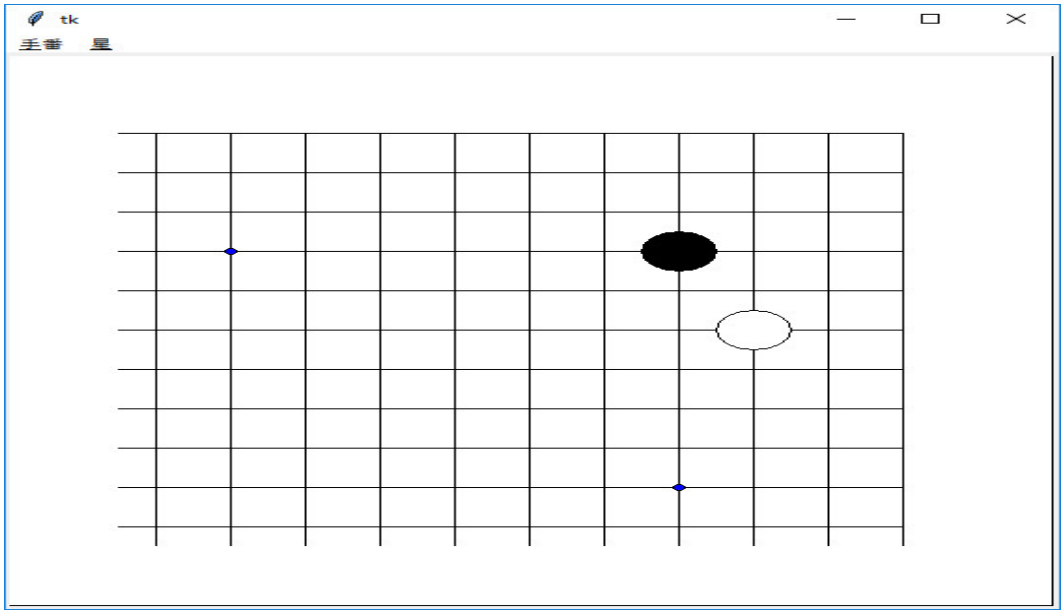
実行すると



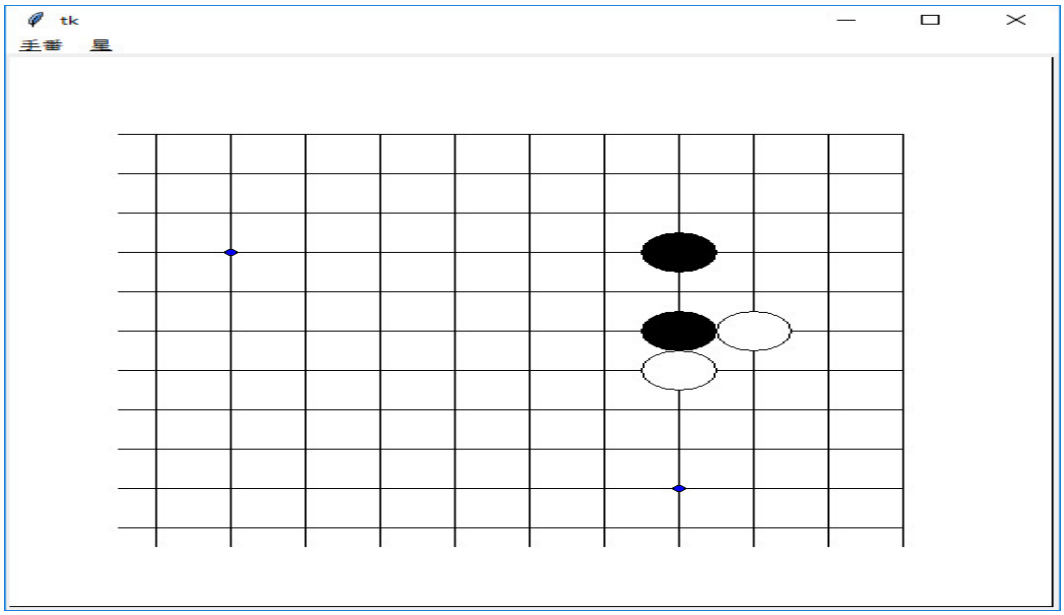
となります。メニューの「星」の「小ゲイマガカリ」の「ツケノビ」を選択すると



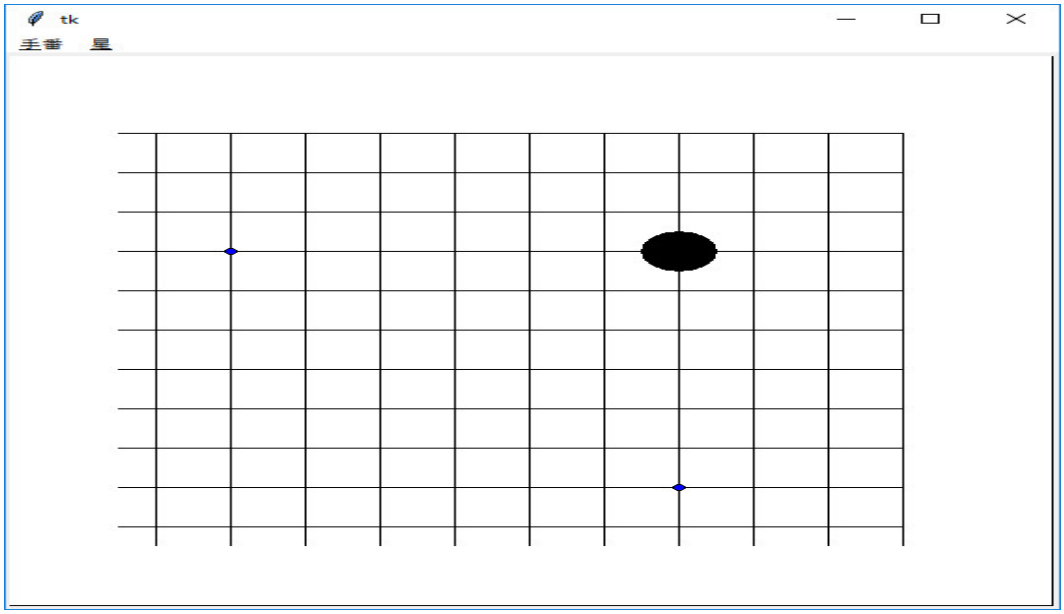
となります。「星」の「小ゲイマガカリ」の「ツケノビ」の定石なので、星の位置をクリックします。



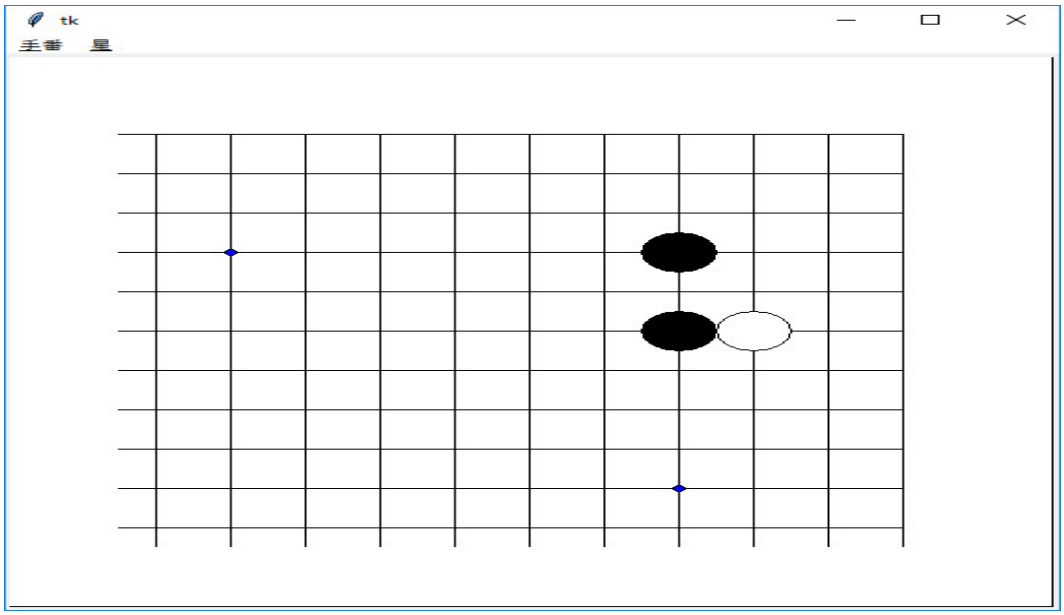
今は先手をユーザーが、後手をコンピュータが受け持っていますから、自動的に白の手はコンピュータが打ちます。続けて黒の手をクリックします。



メニューの「手番」の「後手」をクリックするとコンピュータが先手を受け持ちます。メニューの「星」の「小ゲイマガカリ」の「ツケオサエ」を選択すると



となります。白の手を打つと



となります。

定石のデータを打ち込むだけで大変です。覚えてなくてひどい目にあった定石から順番に暇なときに打ち込んでいけばいいです。しかし、上のような座標のデータを書籍から拾うのは大変です。盤面をクリックしていくと自動的に定石データを作ってくれるプログラムを作りましょう。今作っているプログラムにこの機能を追加することも可能ですが、専用のプログラムを作る方が、プログラムも短くなりますし、間違えてデバッグの泥沼に陥る確率も少なくなります。大部分は、今作っているプログラムをそのまま使えます。必要な修正をすると以下のようなプログラムが考えられます。

```
from tkinter import *
import tkinter.filedialog
import sys, os.path

root = Tk()
##root.option_add("*font", ("FixedSys", 10))

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0
history = []
capture = []
```

```
path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
```

```
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board, capture
    board[tz] = color
    un_col = flip_color(color)
    cap = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)
            if d == 1: cap = cap | 1
            elif d == WIDTH: cap = cap | 2
            elif d == -1: cap = cap | 4
            else: cap = cap | 8
```

```
    capture.append(cap)


def display_board():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')

def set_ban():
    global num, board, history, capture
    new_jyoseki = []
    num = 0
    board = basic_board.copy()
    history = []
    capture = []
    display_board()

def set_joseki():
    global num, history, joseki
    FLAG = True
    for j_list in joseki:
        if len(history) == len(j_list):
            flag = True
            for i in range(len(history)):
```

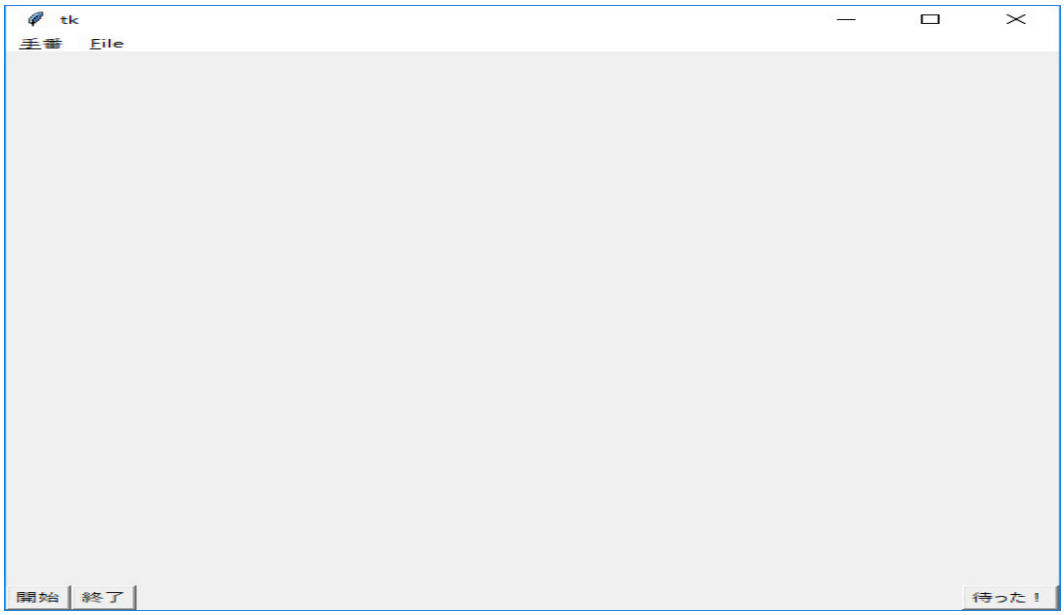
```
                if history[i] != j_list[i]:
                    flag = False
            if flag == True:
                FLAG = False
                break
    if FLAG == True:
        joseki.append(history)
    print('joseki=', joseki)

def cover_stone(tz, color):
    global board
    board[tz] = color
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0:
            take_stone(z, color)
def matta():
     global num, history, board, capture
     cap = capture[-1]
     capture = capture[:-1]
     te = history[-1]
     x, y , cl = te
     history = history[:-1]
     if cap & 8: cover_stone(get_z(x,y)-WIDTH, 1 if cl=='b' else 2)
     elif cap & 4: cover_stone(get_z(x,y)-1, 1 if cl=='b' else 2)
     elif cap & 2: cover_stone(get_z(x,y)+WIDTH, 1 if cl=='b' else 2)
     elif cap & 1: cover_stone(get_z(x,y)+1, 1 if cl=='b' else 2)
     board[get_z(x,y)] = 0
     num -= 1
     display_board()

M_STOP_FLAG = False

def joseki2file():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.asksaveasfilename(initialdir=path_name)
    if filename:
```

```
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        ss = "[\n"
        f.write(ss)
        for k in range(len(joseki)):
            j_list = joseki[k]
            ss = "["
            for i in range(len(j_list)):
                x, y, c = j_list[i]
                ss += "(" + str(x)+"," + str(y)+","+c+")"
                if i < len(j_list)-1:
                    ss += ","
            ss += "]"
            if k < len(joseki)-1:
                ss += ","
            ss += "\n"
            f.write(ss)
        ss = "]\n"
        f.write(ss)
        f.close()
    M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
```

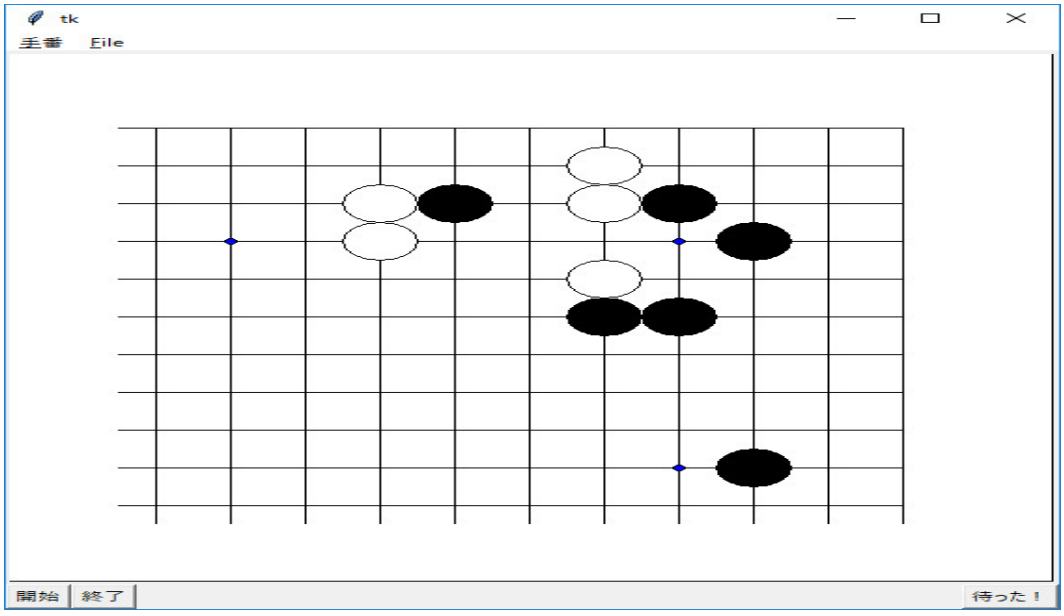
```
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
menufile = Menu(menubar, tearoff = False)
menubar.add_cascade(label="File", underline = 0, menu=menufile)
menuout = Menu(menufile, tearoff = False)
##menufile.add_cascade(label="保存", underline = 0, menu=menuout)
menufile.add_command(label = "保存", under = 0, command = joseki2file)
menufile.add_separator
menufile.add_command(label = "読込", under = 0, command = file2joseki)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()
button1 = Button(root, text = '開始', command = set_ban)
button1.pack(side=LEFT)
button2 = Button(root, text = '終了', command = set_joseki)
button2.pack(side=LEFT)
button3 = Button(root, text = '待った！', command = matta)
button3.pack(side=RIGHT)
```

```
def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history, M_STOP_FLAG
    if M_STOP_FLAG == True: return

    sx, sy = get_pos(event.x, event.y)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if num % 2 == 0 else 'w'
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()

canvas.bind("<ButtonPress-1>", on_pressed)
root.mainloop()
```

で、それなりのものが出来ています。実行すると



となります。左下隅の「開始」のボタンをクリックすると定石を打ち込むことが出来ます。



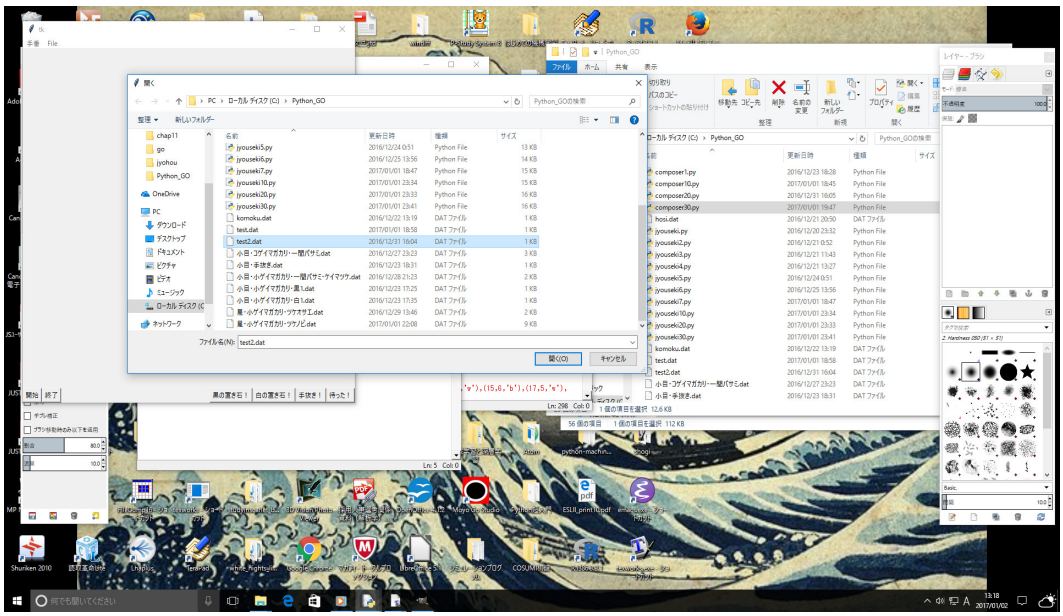
のように定石を順に打ち込むと「終了」のボタンをクリックします。これで一つの定石が joseki に append されます。

```
joseki= [[(17, 4, 'b'), (15, 3, 'w'), (13, 3, 'b'), (15, 5, 'w'),
(16, 6, 'b'), (12, 4, 'w'), (16, 3, 'b'), (15, 2, 'w'), (15, 6, 'b'),
 (12, 3, 'w'), (17, 10, 'b')]]
```

更に「開始」と「終了」を繰り返せば、次々、打ち込んだ定石が joseki に append されます。「待った！」のボタンで待ったをすることが出来ます。メニューの「File」の「保存」で joseki の内容をファイルに保存できます。ファイルに保存した定石は

```
[
[(16,4,b),(17,6,w),(16,6,b),(16,7,w),(15,6,b),(17,5,w),(17,4,b),(17,8,w),(14,4,b)]
[(16,4,b),(17,6,w),(16,6,b),(16,7,w),(15,6,b),(17,5,w),(17,4,b),(17,10,w),(10,3,b)]
]
```

のような形式で保存されています。即ち、一行目は [だけ、最後の行は] だけ、中間には各行に一つの定石が改行なしで保存されています。手で、ファイルを作るときにも、この形式を守ってください。メニューの「File」の「読込」でファイルの内容を joseki にセット出来ます。「開始」と「終了」を繰り返せば、次々、打ち込んだ定石が joseki に append されます。メニューの「手番」は現在は使っていません。メニューの「File」の「読込」で



ファイルを選択する時、マウスでクリックしますが、そのクリックが間接的にソフトのカンバス上で行われたことになると

のようにエラーメッセージが出ます。

```
M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
```

```
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)
def on_pressed(event):
    global num, board, history, M_STOP_FLAG
    if M_STOP_FLAG == True: return

    sx, sy = get_pos(event.x, event.y)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if num % 2 == 0 else 'w'
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()
```

のように、M_STOP_FLAG を使って、対処しているつもりですが、すり抜けて、ファイルを指定するマウスクリックが、石を打ったと解釈されて、on_pressed(event) が実行され、エラーメッセージになります。抜本的な対処法が分かりませんが、このようなエラーメッセージが出ても、ソフトの実行には影響がありませんので無視して実行してください。このことは次の最初に作ったプログラムの修正でも起きます。

次に、この保存している定石集を使って覚えることも出来るように、最初に作ったプログラムを修正します。

```
from tkinter import *
import random
import tkinter.filedialog
import sys, os.path

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)
```

```
joseki = []
num = 0
history = []
path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
```

```
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)

def display_board():
```

```
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')
def get_candidate(num):
    candidate = []
    for j_list in joseki:
        flag = True
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                if j_list[num] not in candidate:
                    candidate.append(j_list[num])
    return candidate
def basic_start():
    global joseki, num, board, history
    num = 0
    board = basic_board.copy()
    history = []
    if teban.get() == 2:
        candidate = get_candidate(num)
        n = random.randint(0, len(candidate)-1)
```

```
            tx, ty, cl = candidate[n]
            put_stone(get_z(tx, ty), get_col(cl))
            history.append((tx, ty, cl))
            num += 1
        display_board()

def start():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi1B
    else:
        joseki = hosi1W
    basic_start()

def start2():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi2B
    else:
        joseki = hosi2B
    basic_start()

M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
```

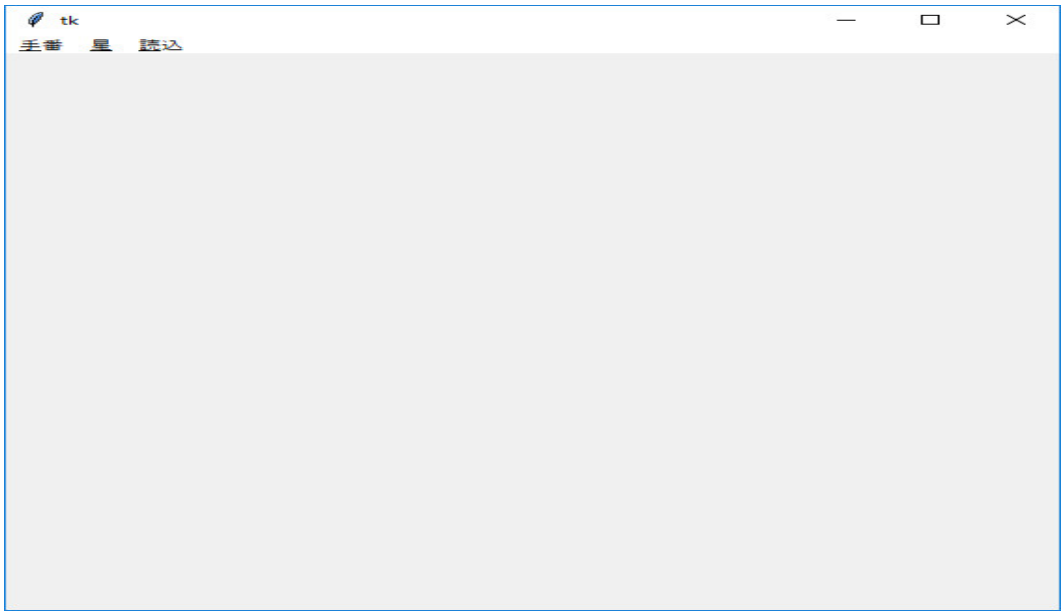
```
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)
    basic_start()

def restart():
    basic_start()

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)
josekiopen = Menu(menubar, tearoff = False)
menubar.add_cascade(label="読込", underline = 0, menu=josekiopen)
josekiopen.add_command(label = "ファイルから定石", under = 0, command = file2joseki)
```

```
josekiopen.add_command(label = "再度挑戦", under = 0, command = restart)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()

def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history
    if M_STOP_FLAG == True: return
    sx, sy = get_pos(event.x, event.y)
    candidate = get_candidate(num)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if teban.get() == 1 else 'w'
    if is_candidate(tx, ty, cl,candidate) == False:
        canvas.create_text(200, 10, text = 'Failure!',
                                font = ('FixedSys', 20))
        return
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    candidate = get_candidate(num)
    if candidate == []:
        display_board()
        canvas.create_text(200, 10, text = 'Congratulations!',
                                font = ('FixedSys', 20))
        return
    n = random.randint(0, len(candidate)-1)
    tx, ty, cl = candidate[n]
```

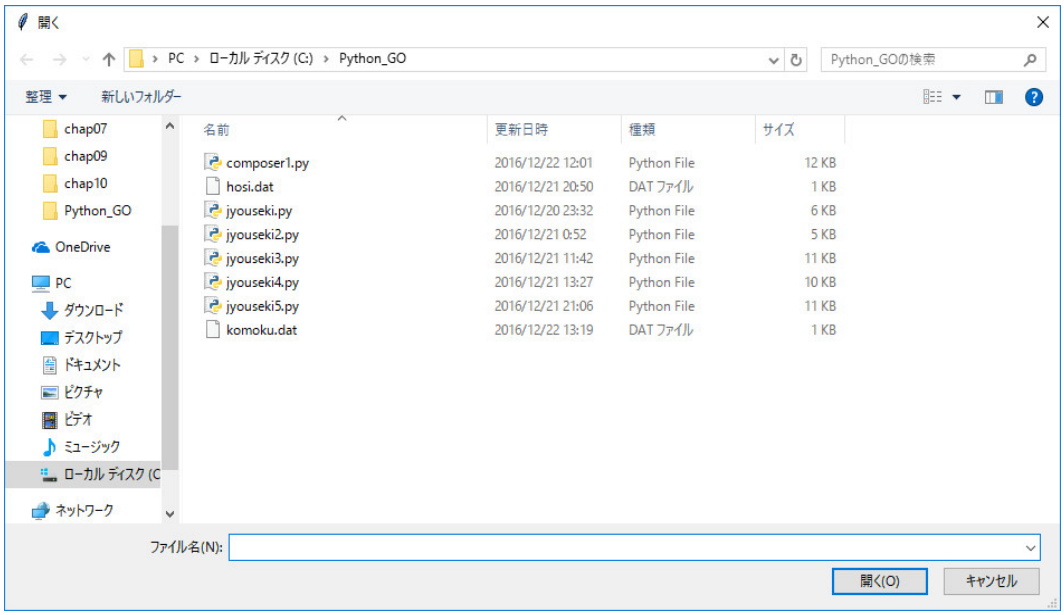
```
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    display_board()
    num += 1
    candidate = get_candidate(num)
    if candidate == []:
        canvas.create_text(200, 10, text = 'Congratulations!',
                                font = ('FixedSys', 20))

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

# 定石
##hosi1 = [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
##(17,4,'b'),(17,8,'w'),(14,4,'b')]

hosi1B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(18,2,'w'),
     (17,2,'b'),(17,1,'w'),(15,2,'b'),(16,3,'w'),(17,3,'b'),(16,1,'w'),
     (18,1,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(17,2,'w'),
     (18,2,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(17,7,'b'),(18,7,'w'),
     (17,8,'b'),(18,8,'w'),(14,2,'b'),(18,2,'w'),(17,2,'b'),(17,3,'w'),
     (16,3,'b'),(15,2,'w'),(18,3,'b'),(17,1,'w'),(15,1,'b'),(17,3,'w'),
     (19,5,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(15,2,'w'),(14,2,'b'),(17,2,'w'),
     (17,7,'b'),(18,7,'w'),(17,8,'b'),(18,8,'w'),(18,2,'b'),(18,1,'w'),
```

```
     (19,5,'b'),(18,6,'w'),(17,3,'b'),(16,2,'w'),(19,2,'b')],

    ]
hosi1W = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')]
    ]
hosi2B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(17,7,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(18,7,'w'),
     (14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(14,4,'w'),
     (18,8,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(18,6,'w'),(15,7,'b'),(16,8,'w'),(14,6,'b'),(15,8,'w'),
     (15,5,'b'),(18,5,'w'),(17,3,'b')]
    ]

root.mainloop()
```

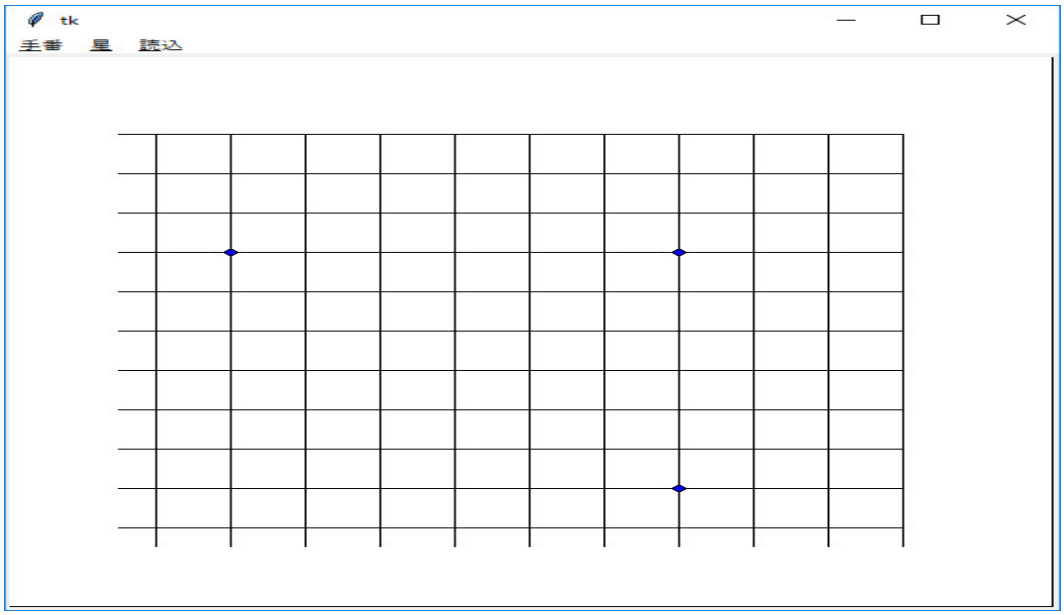
実行すると



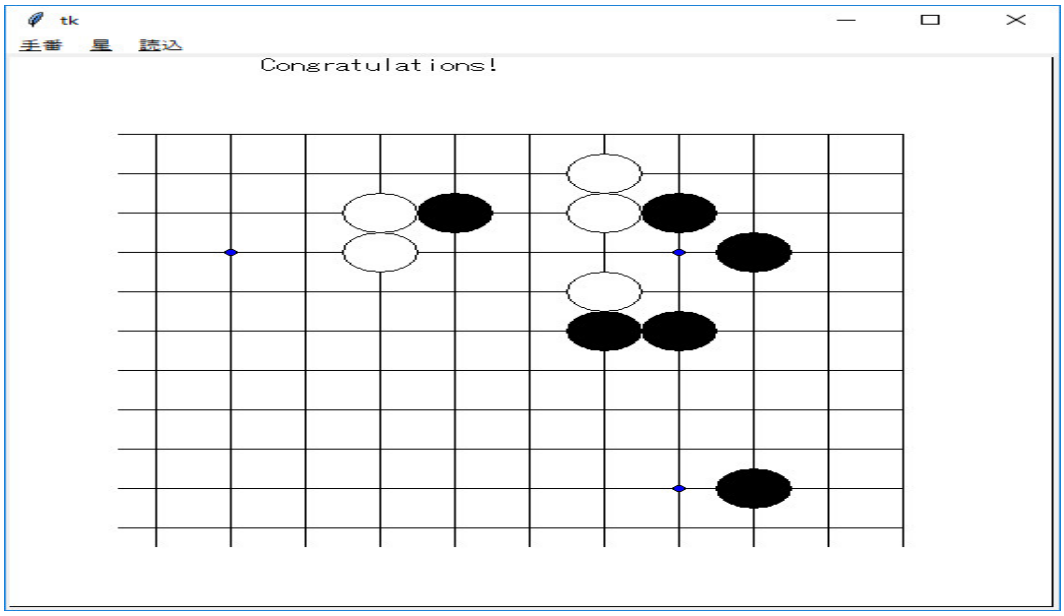
となります。メニューの「読込」の「ファイルから定石」を選択すると



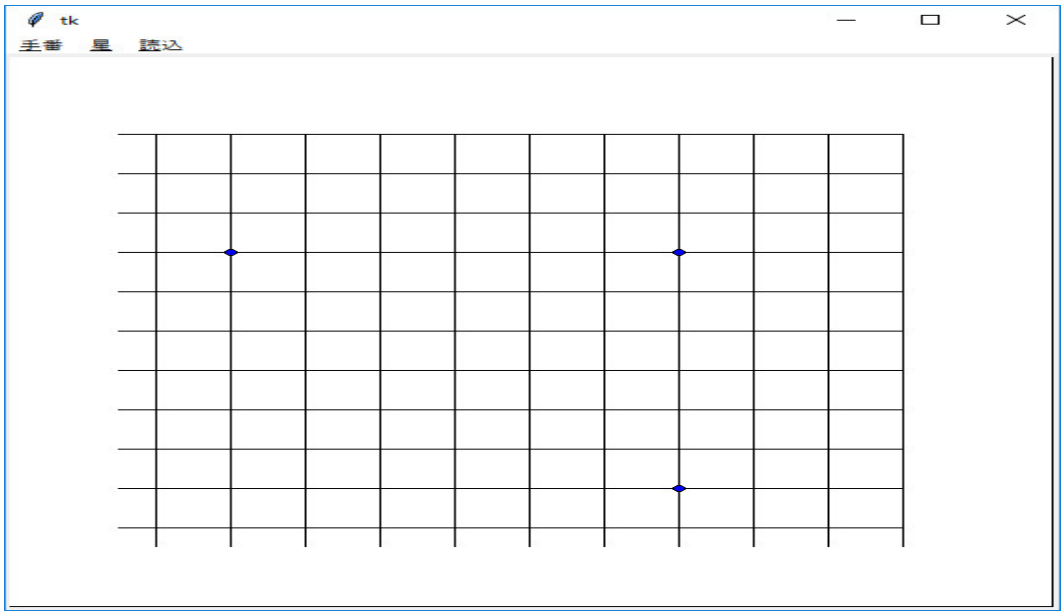
ファイル選択のダイアログボックスが開きますので、保存してあるファイルを読み込みます。



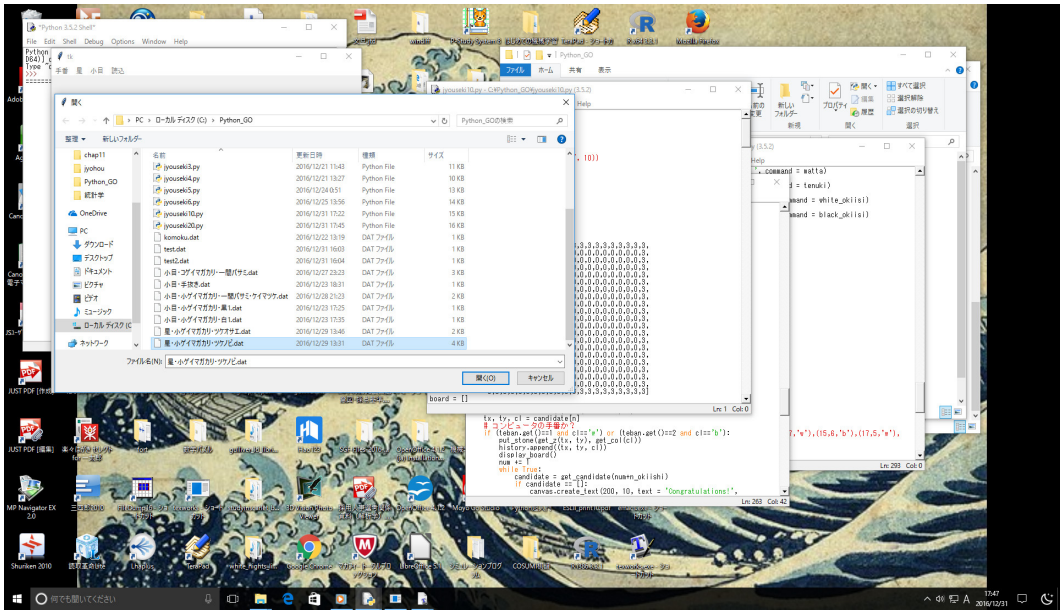
ファイルの定石を練習できます。



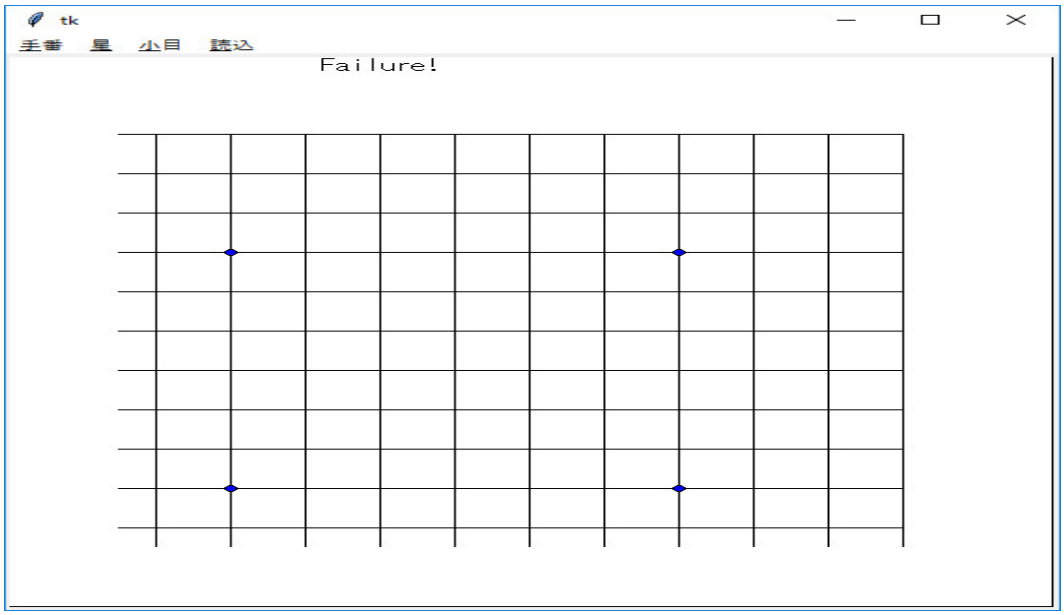
引き続き、この定石集を練習したいときは、メニューの「読込」の「再度挑戦」をクリックすればいいです。



但し、下図のように、メニューの「読込」の「ファイルから定石」を選択した時、

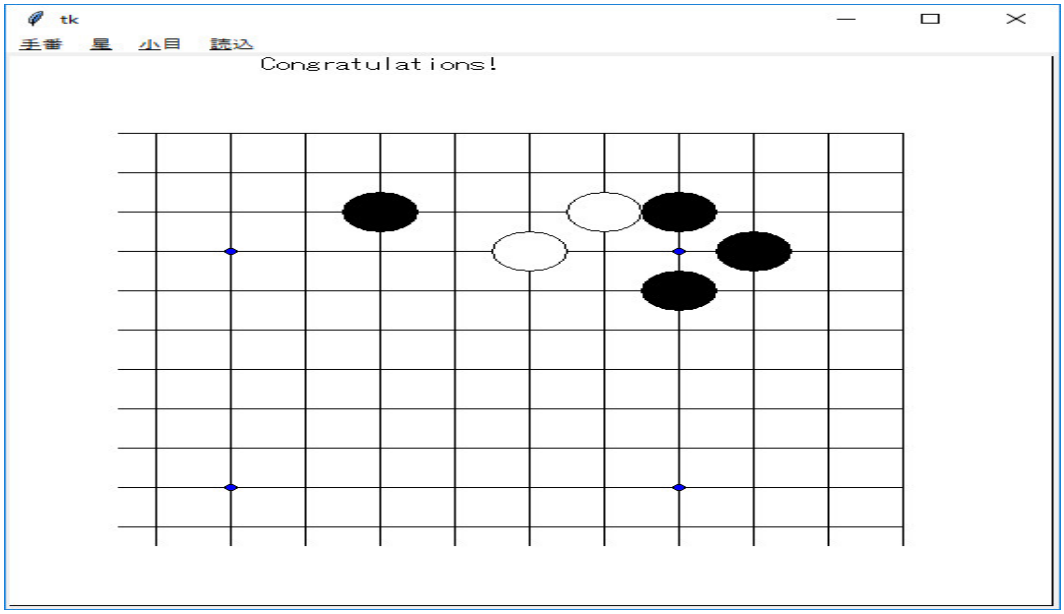


ファイルを指定するマウスのクリックがソフトのキャンバス領域にある場合は

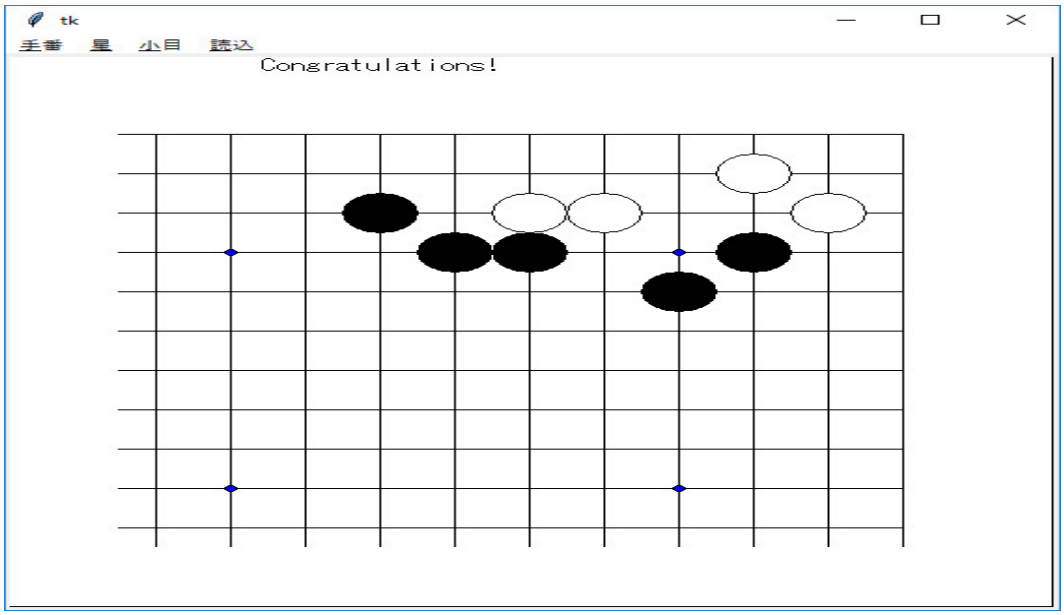


のように、Failure! と表示されますが、気にしないで、実行してください。それぞれのイベントがどのタイミングで実行されているか不明で、これの対策が分かりません。実用上、支障はないです。

これで一様、定石の勉強はできるようになりましたが、手を抜いた時の定石が勉強できるようになっていません。



や



の場合の練習もできるようにしましょう。定石は

```
komoku1B = [
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'b'),(14,4,'w'),(16,3,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'b'),(17,2,'w'),(14,4,'b'),
     (14,3,'w'),(13,4,'b'),(18,3,'w')]
    ]
```

のように単純に表現します。このような表現にも対応できるように二つのプログラムを修正します。

定石をファイルに保存するプログラムは

```
from tkinter import *
import tkinter.filedialog
import sys, os.path

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0
history = []
```

```
capture = []

path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
```

```
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board, capture
    board[tz] = color
    un_col = flip_color(color)
    cap = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)
            if d == 1: cap = cap | 1
            elif d == WIDTH: cap = cap | 2
            elif d == -1: cap = cap | 4
```

```
            else: cap = cap | 8
    capture.append(cap)


def display_board():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')

def set_ban():
    global num, board, history, capture
    new_jyoseki = []
    num = 0
    board = basic_board.copy()
    history = []
    capture = []
    display_board()

def set_joseki():
    global num, history, joseki
    FLAG = True
    for j_list in joseki:
        if len(history) == len(j_list):
            flag = True
```

```
            for i in range(len(history)):
                if history[i] != j_list[i]:
                    flag = False
            if flag == True:
                FLAG = False
                break
    if FLAG == True:
        joseki.append(history)
    print('joseki=', joseki)

def cover_stone(tz, color):
    global board
    board[tz] = color
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0:
            take_stone(z, color)
def matta():
     global num, history, board, capture
     cap = capture[-1]
     capture = capture[:-1]
     te = history[-1]
     x, y , cl = te
     history = history[:-1]
     if cap & 8: cover_stone(get_z(x,y)-WIDTH, 1 if cl=='b' else 2)
     elif cap & 4: cover_stone(get_z(x,y)-1, 1 if cl=='b' else 2)
     elif cap & 2: cover_stone(get_z(x,y)+WIDTH, 1 if cl=='b' else 2)
     elif cap & 1: cover_stone(get_z(x,y)+1, 1 if cl=='b' else 2)
     board[get_z(x,y)] = 0
     num -= 1
     display_board()
def tenuki():
    global num
    num += 1

M_STOP_FLAG = False

def joseki2file():
    global M_STOP_FLAG
```

```
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        ss = "[\n"
        f.write(ss)
        for k in range(len(joseki)):
            j_list = joseki[k]
            ss = "["
            for i in range(len(j_list)):
                x, y, c = j_list[i]
                ss += "(" + str(x)+"," + str(y)+","+c+")"
                if i < len(j_list)-1:
                    ss += ","
            ss += "]"
            if k < len(joseki)-1:
                ss += ","
            ss += "\n"
            f.write(ss)
        ss = "]\n"
        f.write(ss)
        f.close()
    M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
```

```
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
menufile = Menu(menubar, tearoff = False)
menubar.add_cascade(label="File", underline = 0, menu=menufile)
menuout = Menu(menufile, tearoff = False)
##menufile.add_cascade(label="保存", underline = 0, menu=menuout)
menufile.add_command(label = "保存", under = 0, command = joseki2file)
menufile.add_separator
menufile.add_command(label = "読込", under = 0, command = file2joseki)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()
button1 = Button(root, text = '開始', command = set_ban)
button1.pack(side=LEFT)
button2 = Button(root, text = '終了', command = set_joseki)
```

```
button2.pack(side=LEFT)
button3 = Button(root, text = '待った！', command = matta)
button3.pack(side=RIGHT)
button4 = Button(root, text = '手抜き！', command = tenuki)
button4.pack(side=RIGHT)

def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history, M_STOP_FLAG
    if M_STOP_FLAG == True: return

    sx, sy = get_pos(event.x, event.y)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if num % 2 == 0 else 'w'
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()

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

root.mainloop()
```

と修正すればいいです。
　本体のプログラムは

```
from tkinter import *
import random
import tkinter.filedialog
```

```
import sys, os.path

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0
history = []
path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]
```

```
def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
```

```
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)


def display_board():
    canvas.create_rectangle(0, 0, 560, 560, fill='white')
    for k in range(80,520,40):
        canvas.create_line(60,k,480,k)
    for k in range(80,520,40):
        canvas.create_line(k,80,k,500)
    canvas.create_oval(120-3, 200-3, 120+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 200-3, 360+3, 200+3, fill='blue')
    canvas.create_oval(360-3, 440-3, 360+3, 440+3, fill='blue')
    canvas.create_oval(120-3, 440-3, 120+3, 440+3, fill='blue')
    for i in range(9, 20, 1):
        for k in range(1, 12, 1):
            if board[get_z(i,k)] == 1:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 40*(i-9)+80
                py = 40*(k-1)+80
                canvas.create_oval(px-20, py-20, px+20, py+20, fill='white')
def get_candidate(num):
    candidate = []
    for j_list in joseki:
        flag = True
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                if j_list[num] not in candidate:
                    candidate.append(j_list[num])
```

```
    return candidate
def basic_start():
    global joseki, num, board, history
    num = 0
    board = basic_board.copy()
    history = []
    if teban.get() == 2:
        candidate = get_candidate(num)
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        num += 1
    display_board()

def start():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi1B
    else:
        joseki = hosi1W
    basic_start()

def start2():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi2B
    else:
        joseki = hosi2B
    basic_start()
def start3():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = komoku1B
    else:
        joseki = komoku1B
```

```
    basic_start()

M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)
    basic_start()

def restart():
    basic_start()

# メニューバーの生成
```

```
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)
komoku = Menu(menubar, tearoff = False)
menubar.add_cascade(label="小目", underline = 0, menu=komoku)
##komoku.add_cascade(label="コスミ", underline = 0, menu=kakari)
komoku.add_command(label = "コスミ", under = 0, command = start3)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)
josekiopen = Menu(menubar, tearoff = False)
menubar.add_cascade(label="読込", underline = 0, menu=josekiopen)
josekiopen.add_command(label = "ファイルから定石", under = 0, command = file2joseki)
josekiopen.add_command(label = "再度挑戦", under = 0, command = restart)

canvas = Canvas(root, width = 560, height=560)
canvas.pack()
def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+40*int((x-60)/40)
    py = 80+40*int((y-60)/40)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history
    if M_STOP_FLAG == True: return
```

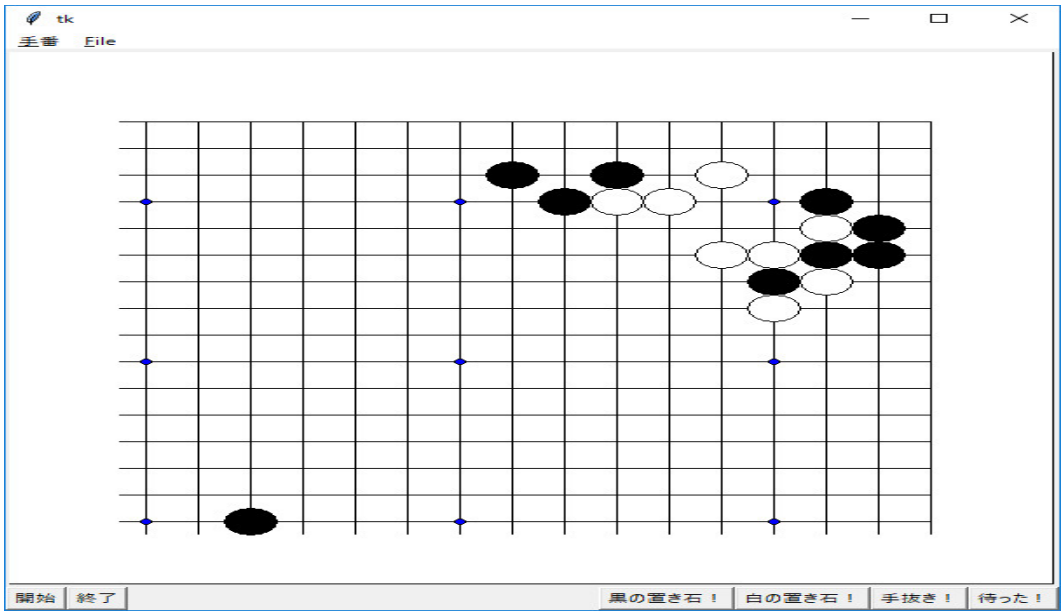
```
canvas.create_text(200, 10, text = '                    ',
                   font = ('FixedSys', 20))
sx, sy = get_pos(event.x, event.y)
candidate = get_candidate(num)
print('candidate=', candidate)
tx = (sx - 80) // 40 + 9
ty = (sy - 80) // 40 + 1
cl = 'b' if teban.get() == 1 else 'w'
if is_candidate(tx, ty, cl, candidate) == False:
    display_board()
    canvas.create_text(200, 10, text = 'Failure!',
                       font = ('FixedSys', 20))
    return
put_stone(get_z(tx, ty), get_col(cl))
history.append((tx, ty, cl))
num += 1
display_board()
candidate = get_candidate(num)
if candidate == []:
    display_board()
    canvas.create_text(200, 10, text = 'Congratulations!',
                       font = ('FixedSys', 20))
    return
n = random.randint(0, len(candidate)-1)
tx, ty, cl = candidate[n]
# コンピュータの手番か？
if (teban.get()==1 and cl=='w') or (teban.get()==2 and cl=='b'):
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    display_board()
    num += 1
    while True:
        candidate = get_candidate(num)
        if candidate == []:
            canvas.create_text(200, 10, text = 'Congratulations!',
                               font = ('FixedSys', 20))
            return
        else:
            # 人間が手抜きしたか？
```

```
                n = random.randint(0, len(candidate)-1)
                tx, ty, cl = candidate[n]
                if (teban.get()==1 and cl=='w') or (teban.get()==2 and cl=='b'):
                    put_stone(get_z(tx, ty), get_col(cl))
                    history.append((tx, ty, cl))
                    display_board()
                    num += 1
                else:
                    break
    else: # コンピュータが手抜きした
        canvas.create_text(200, 10, text = '手抜き！',
                           font = ('FixedSys', 20))
canvas.bind("<ButtonPress-1>", on_pressed)
# 定石
hosi1B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(18,2,'w'),
     (17,2,'b'),(17,1,'w'),(15,2,'b'),(16,3,'w'),(17,3,'b'),(16,1,'w'),
     (18,1,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(17,2,'w'),
     (18,2,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(17,7,'b'),(18,7,'w'),
     (17,8,'b'),(18,8,'w'),(14,2,'b'),(18,2,'w'),(17,2,'b'),(17,3,'w'),
     (16,3,'b'),(15,2,'w'),(18,3,'b'),(17,1,'w'),(15,1,'b'),(17,3,'w'),
     (19,5,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(15,2,'w'),(14,2,'b'),(17,2,'w'),
     (17,7,'b'),(18,7,'w'),(17,8,'b'),(18,8,'w'),(18,2,'b'),(18,1,'w'),
     (19,5,'b'),(18,6,'w'),(17,3,'b'),(16,2,'w'),(19,2,'b')],
```

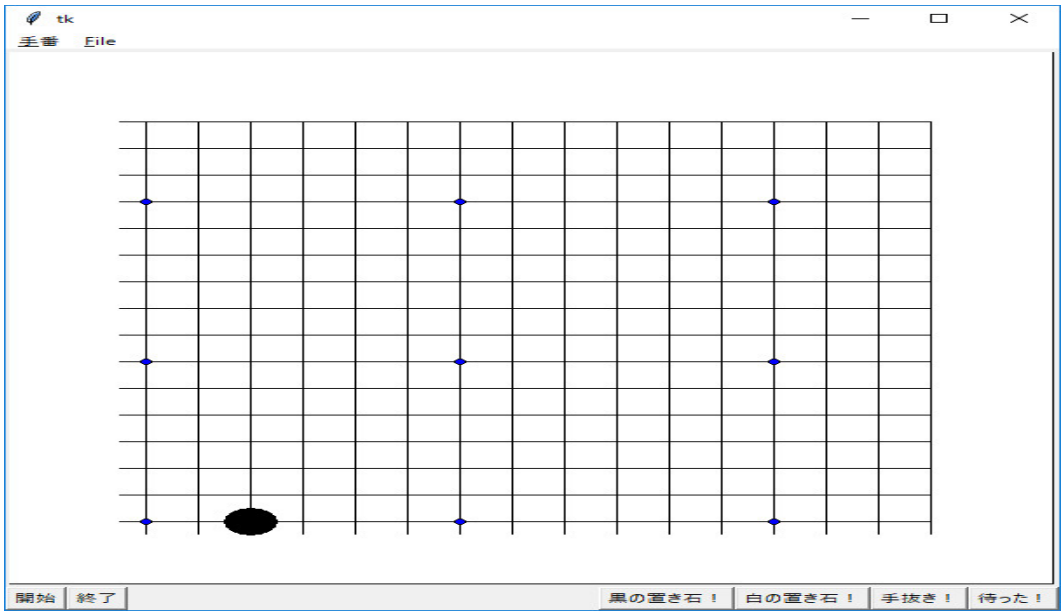
```
    ]
hosi1W = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')]
    ]
hosi2B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(17,7,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(18,7,'w'),
     (14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(14,4,'w'),
     (18,8,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(18,6,'w'),(15,7,'b'),(16,8,'w'),(14,6,'b'),(15,8,'w'),
     (15,5,'b'),(18,5,'w'),(17,3,'b')]
    ]
komoku1B = [
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'b'),(14,4,'w'),(16,3,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'b'),(17,2,'w'),(14,4,'b'),
     (14,3,'w'),(13,4,'b'),(18,3,'w')]
    ]
root.mainloop()
```

と修正すればいいです。

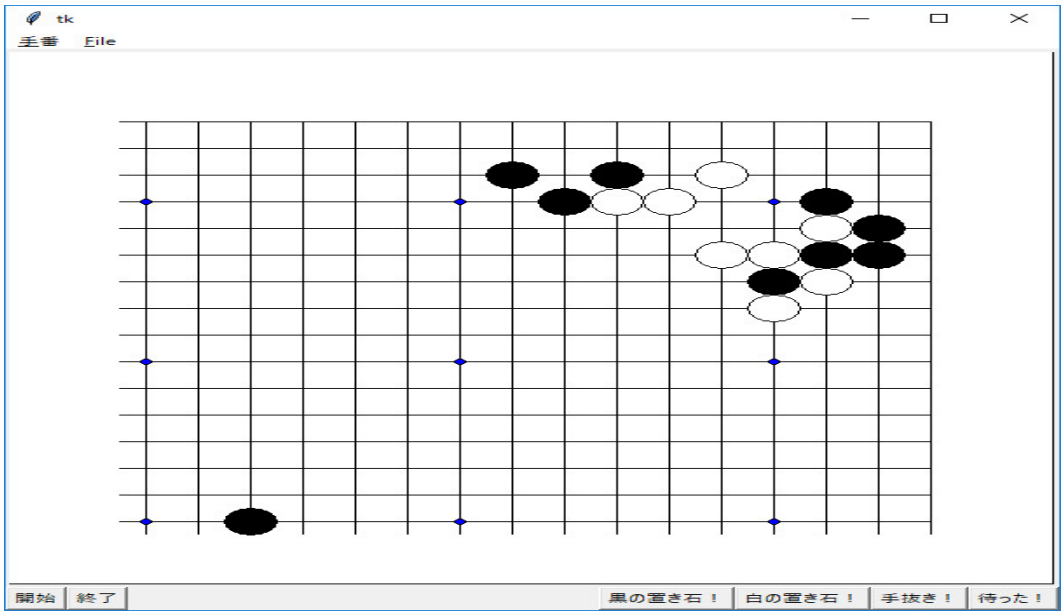
最後に、シチョウが良いかどうかで手が変わる場合



に、



のように前もって置き石をしておくように、プログラムを修正します。「開始」のボタンをクリックして、石を配置するたびに、それが黒石なら「黒の置き石！」、白石なら「白の置き石！」のボタンをクリックすれば、それぞれの置き石として、表示が変わり、座標を負の値として保存されます。すなわち、



とその後打ち込み、保存すれば

```
[
[(-6,-16,b),(17,4,b),(15,3,w),(13,3,b),(13,4,w),(12,4,b),(14,4,w),(11,3,b),(16,6,w)
]
```

のように、置き石は座標を負の値として保存されます。置き石は各定石の最初にだけ任意個数置くことが可能です。ついでに盤面も大きくしています。このようなデータを作るプログラムは

```
from tkinter import *
import tkinter.filedialog
import sys, os.path

root = Tk()

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
num = 0
history = []
capture = []
```

```
path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
```

```
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board, capture
    board[tz] = color
    un_col = flip_color(color)
    cap = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)
            if d == 1: cap = cap | 1
            elif d == WIDTH: cap = cap | 2
            elif d == -1: cap = cap | 4
            else: cap = cap | 8
    capture.append(cap)
```

```
def display_board():
    canvas.create_rectangle(0, 0, 600, 600, fill='white')
    for k in range(80,560,30):
        canvas.create_line(65,k,530,k)
    for k in range(80,540,30):
        canvas.create_line(k,80,k,545)
    canvas.create_oval( 80-3, 170-3,  80+3, 170+3, fill='blue')
    canvas.create_oval(260-3, 170-3, 260+3, 170+3, fill='blue')
    canvas.create_oval(440-3, 170-3, 440+3, 170+3, fill='blue')
    canvas.create_oval( 80-3, 350-3,  80+3, 350+3, fill='blue')
    canvas.create_oval(260-3, 350-3, 260+3, 350+3, fill='blue')
    canvas.create_oval(440-3, 350-3, 440+3, 350+3, fill='blue')
    canvas.create_oval( 80-3, 530-3,  80+3, 530+3, fill='blue')
    canvas.create_oval(260-3, 530-3, 260+3, 530+3, fill='blue')
    canvas.create_oval(440-3, 530-3, 440+3, 530+3, fill='blue')
    for i in range(4, 20, 1):
        for k in range(1, 17, 1):
            if board[get_z(i,k)] == 1:
                px = 30*(i-4)+80
                py = 30*(k-1)+80
                canvas.create_oval(px-15, py-15, px+15, py+15, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 30*(i-4)+80
                py = 30*(k-1)+80
                canvas.create_oval(px-15, py-15, px+15, py+15, fill='white')

def set_ban():
    global num, board, history, capture
    new_jyoseki = []
    num = 0
    board = basic_board.copy()
    history = []
    capture = []
    display_board()

def set_joseki():
    global num, history, joseki
```

```
    FLAG = True
    for j_list in joseki:
        if len(history) == len(j_list):
            flag = True
            for i in range(len(history)):
                if history[i] != j_list[i]:
                    flag = False
            if flag == True:
                FLAG = False
                break
    if FLAG == True:
        joseki.append(history)
    print('joseki=', joseki)

def cover_stone(tz, color):
    global board
    board[tz] = color
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0:
            take_stone(z, color)
def matta():
     global num, history, board, capture
     cap = capture[-1]
     capture = capture[:-1]
     te = history[-1]
     x, y , cl = te
     history = history[:-1]
     if cap & 8: cover_stone(get_z(x,y)-WIDTH, 1 if cl=='b' else 2)
     elif cap & 4: cover_stone(get_z(x,y)-1, 1 if cl=='b' else 2)
     elif cap & 2: cover_stone(get_z(x,y)+WIDTH, 1 if cl=='b' else 2)
     elif cap & 1: cover_stone(get_z(x,y)+1, 1 if cl=='b' else 2)
     board[get_z(x,y)] = 0
     num -= 1
     display_board()
def tenuki():
    global num
    num += 1
def white_okiisi():
```

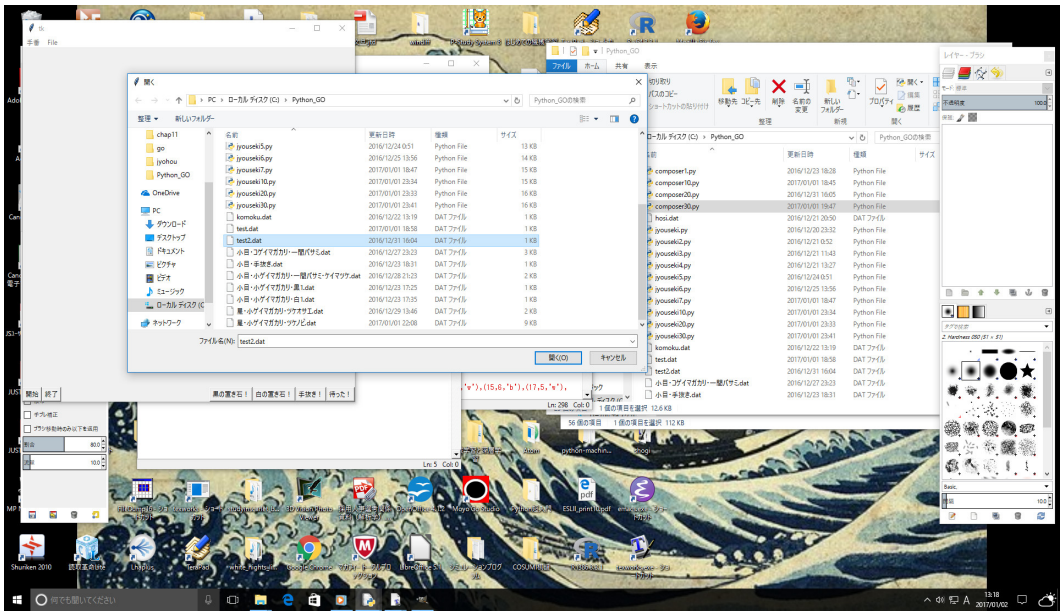
```
    global num, history, board, capture
    x, y , cl = history[-1]
    board[get_z(x,y)] = get_col('w')
    history[-1] =(-x, -y, 'w')
    num = 0
    display_board()
def black_okiisi():
    global num, history, board, capture
    x, y , cl = history[-1]
    board[get_z(x,y)] = get_col('b')
    history[-1] =(-x, -y, 'b')
    num = 0
    display_board()

M_STOP_FLAG = False

def joseki2file():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.asksaveasfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "w")
        ss = "[\n"
        f.write(ss)
        for k in range(len(joseki)):
            j_list = joseki[k]
            ss = "["
            for i in range(len(j_list)):
                x, y, c = j_list[i]
                ss += "(" + str(x)+"," + str(y)+","+c+")"
                if i < len(j_list)-1:
                    ss += ","
            ss += "]"
            if k < len(joseki)-1:
                ss += ","
            ss += "\n"
            f.write(ss)
```

```
            ss = "]\n"
            f.write(ss)
            f.close()
        M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
```

```
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
menufile = Menu(menubar, tearoff = False)
menubar.add_cascade(label="File", underline = 0, menu=menufile)
menuout = Menu(menufile, tearoff = False)
##menufile.add_cascade(label="保存", underline = 0, menu=menuout)
menufile.add_command(label = "保存", under = 0, command = joseki2file)
menufile.add_separator
menufile.add_command(label = "読込", under = 0, command = file2joseki)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)

canvas = Canvas(root, width = 600, height=600)
canvas.pack()
button1 = Button(root, text = '開始', command = set_ban)
button1.pack(side=LEFT)
button2 = Button(root, text = '終了', command = set_joseki)
button2.pack(side=LEFT)
button3 = Button(root, text = '待った！', command = matta)
button3.pack(side=RIGHT)
button4 = Button(root, text = '手抜き！', command = tenuki)
button4.pack(side=RIGHT)
button5 = Button(root, text = '白の置き石!', command = white_okiisi)
button5.pack(side=RIGHT)
button6 = Button(root, text = '黒の置き石!', command = black_okiisi)
button6.pack(side=RIGHT)

def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+30*int((x-65)/30)
    py = 80+30*int((y-65)/30)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
```

```
def on_pressed(event):
    global num, board, history, M_STOP_FLAG
    if M_STOP_FLAG == True: return

    sx, sy = get_pos(event.x, event.y)
    tx = (sx - 80) // 30 + 4
    ty = (sy - 80) // 30 + 1
    cl = 'b' if num % 2 == 0 else 'w'
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()

canvas.bind("<ButtonPress-1>", on_pressed)
root.mainloop()
```

です。メニューの「File」の「読込」で



ファイルを選択する時、マウスでクリックしますが、そのクリックが間接的にソフトのカンバス上で行われたことになると

のようにエラーメッセージが出ます。

```
M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
```

```
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    M_STOP_FLAG = False
    print('joseki=', joseki)
def on_pressed(event):
    global num, board, history, M_STOP_FLAG
    if M_STOP_FLAG == True: return

    sx, sy = get_pos(event.x, event.y)
    tx = (sx - 80) // 40 + 9
    ty = (sy - 80) // 40 + 1
    cl = 'b' if num % 2 == 0 else 'w'
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()
```

のように、M_STOP_FLAG を使って、対処しているつもりですが、すり抜けて、ファイルを指定するマウスクリックが、石を打ったと解釈されて、on_pressed(event) が実行され、エラーメッセージになります。抜本的な対処法が分かりませんが、このようなエラーメッセージが出ても、ソフトの実行には影響がありませんので、無視して実行して下さい。このことは次のプログラムの修正でも起きます。

改良されたデータを使って、定石を覚えるためのプログラムは

```
from tkinter import *
import random
import tkinter.filedialog
import sys, os.path

root = Tk()
##root.option_add("*font", ("FixedSys", 10))

# variable 用のオブジェクト
teban = IntVar()
teban.set(1)
```

```
joseki = []
num = 0
history = []
path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
```

```
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)

def display_board():
    canvas.create_rectangle(0, 0, 600, 600, fill='white')
```

```
    for k in range(80,560,30):
        canvas.create_line(65,k,530,k)
    for k in range(80,540,30):
        canvas.create_line(k,80,k,545)
    canvas.create_oval( 80-3, 170-3,  80+3, 170+3, fill='blue')
    canvas.create_oval(260-3, 170-3, 260+3, 170+3, fill='blue')
    canvas.create_oval(440-3, 170-3, 440+3, 170+3, fill='blue')
    canvas.create_oval( 80-3, 350-3,  80+3, 350+3, fill='blue')
    canvas.create_oval(260-3, 350-3, 260+3, 350+3, fill='blue')
    canvas.create_oval(440-3, 350-3, 440+3, 350+3, fill='blue')
    canvas.create_oval( 80-3, 530-3,  80+3, 530+3, fill='blue')
    canvas.create_oval(260-3, 530-3, 260+3, 530+3, fill='blue')
    canvas.create_oval(440-3, 530-3, 440+3, 530+3, fill='blue')
    for i in range(4, 20, 1):
        for k in range(1, 17, 1):
            if board[get_z(i,k)] == 1:
                px = 30*(i-4)+80
                py = 30*(k-1)+80
                canvas.create_oval(px-15, py-15, px+15, py+15, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 30*(i-4)+80
                py = 30*(k-1)+80
                canvas.create_oval(px-15, py-15, px+15, py+15, fill='white')
    canvas.create_text(200, 10, text = '                    ',
                       font = ('FixedSys', 20))

n_okiishi = 0
def get_candidate(num):
    candidate = []
    for j_list in joseki:
        flag = True
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                if j_list[num] not in candidate:
                    candidate.append(j_list[num])
```

```
    return candidate
def basic_start():
    global joseki, num, board, history, n_okiishi
    num = 0
    n_okiishi = 0
    board = basic_board.copy()
    history = []

    # 定石に置き石があるか？
    n = random.randint(0, len(joseki)-1)
    j_hist = joseki[n]
    i = 0
    while True:
        x,y,cl = j_hist[i]
        if x < 0:
            put_stone(get_z(-x, -y), get_col(cl))
            history.append((x, y, cl))
            i += 1
            n_okiishi += 1
        else:
            break
    if teban.get() == 2:
        candidate = get_candidate(num+n_okiishi)
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        num += 1
    display_board()

def start():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi1B
    else:
        joseki = hosi1W
    basic_start()
```

```
def start2():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi2B
    else:
        joseki = hosi2B
    basic_start()
def start3():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = komoku1B
    else:
        joseki = komoku1B
    basic_start()

M_STOP_FLAG = False

def file2joseki():
    global M_STOP_FLAG
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
```

```
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    print('joseki=', joseki)
    basic_start()
    M_STOP_FLAG = False

def restart():
    basic_start()

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)
komoku = Menu(menubar, tearoff = False)
menubar.add_cascade(label="小目", underline = 0, menu=komoku)
kogeimakakari = Menu(hosi, tearoff = False)
komoku.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kogeimakakari)
kogeimakakari.add_command(label = "コスミ", under = 0, command = start3)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)
josekiopen = Menu(menubar, tearoff = False)
menubar.add_cascade(label="読込", underline = 0, menu=josekiopen)
josekiopen.add_command(label = "ファイルから定石", under = 0, command = file2joseki)
josekiopen.add_command(label = "再度挑戦", under = 0, command = restart)
```

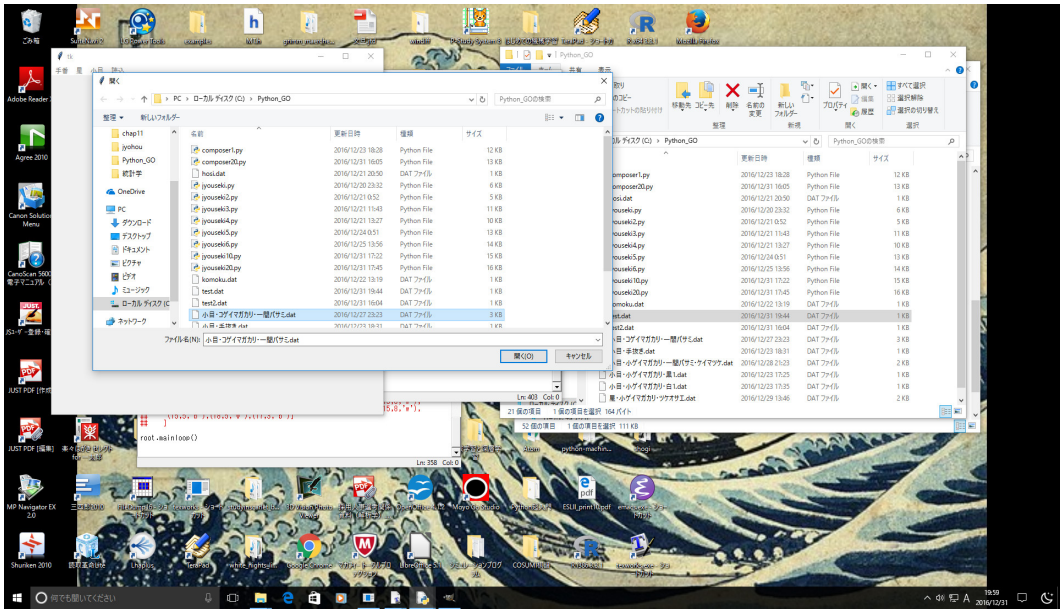
```
canvas = Canvas(root, width = 600, height=600)
canvas.pack()

def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 80+30*int((x-65)/30)
    py = 80+30*int((y-65)/30)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history, n_okiishi
    if M_STOP_FLAG == True:
        sx, sy = get_pos(event.x, event.y)
        return
    canvas.create_text(200, 10, text = '                    ',
                       font = ('FixedSys', 20))
    sx, sy = get_pos(event.x, event.y)
    candidate = get_candidate(num+n_okiishi)
    print('candidate=', candidate)
    tx = (sx - 80) // 30 + 4
    ty = (sy - 80) // 30 + 1
    cl = 'b' if teban.get() == 1 else 'w'
    if is_candidate(tx, ty, cl, candidate) == False:
        display_board()
        canvas.create_text(200, 10, text = 'Failure!',
                           font = ('FixedSys', 20))
        return
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()
    candidate = get_candidate(num+n_okiishi)
    if candidate == []:
        display_board()
```

```
            canvas.create_text(200, 10, text = 'Congratulations!',
                               font = ('FixedSys', 20))
            return
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        # コンピュータの手番か？
        if (teban.get()==1 and cl=='w') or (teban.get()==2 and cl=='b'):
            put_stone(get_z(tx, ty), get_col(cl))
            history.append((tx, ty, cl))
            display_board()
            num += 1
            while True:
                candidate = get_candidate(num+n_okiishi)
                if candidate == []:
                    canvas.create_text(200, 10, text = 'Congratulations!',
                                       font = ('FixedSys', 20))
                    return
                else:
                    # 人間が手抜きしたか？
                    n = random.randint(0, len(candidate)-1)
                    tx, ty, cl = candidate[n]
                    if (teban.get()==1 and cl=='w') or (teban.get()==2 and cl=='b'):
                        put_stone(get_z(tx, ty), get_col(cl))
                        history.append((tx, ty, cl))
                        display_board()
                        num += 1
                    else:
                        break
        else: # コンピュータが手抜きした
            canvas.create_text(200, 10, text = '手抜き！',
                               font = ('FixedSys', 20))

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

# 定石
##hosi1 = [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
##(17,4,'b'),(17,8,'w'),(14,4,'b')]

hosi1B = [
```

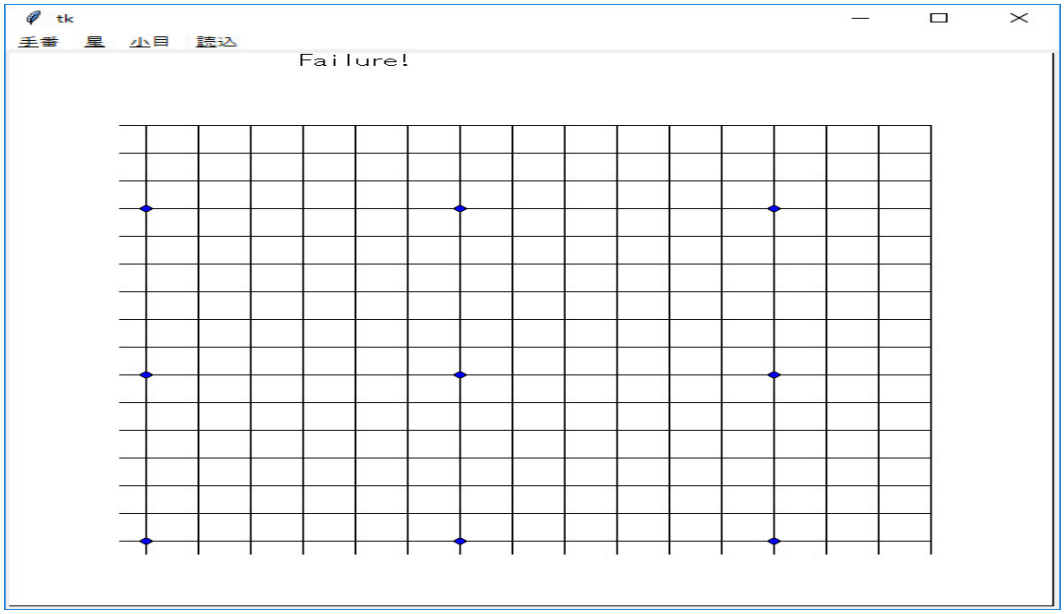
```
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(18,2,'w'),
     (17,2,'b'),(17,1,'w'),(15,2,'b'),(16,3,'w'),(17,3,'b'),(16,1,'w'),
     (18,1,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(17,2,'w'),
     (18,2,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(17,7,'b'),(18,7,'w'),
     (17,8,'b'),(18,8,'w'),(14,2,'b'),(18,2,'w'),(17,2,'b'),(17,3,'w'),
     (16,3,'b'),(15,2,'w'),(18,3,'b'),(17,1,'w'),(15,1,'b'),(17,3,'w'),
     (19,5,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(15,2,'w'),(14,2,'b'),(17,2,'w'),
     (17,7,'b'),(18,7,'w'),(17,8,'b'),(18,8,'w'),(18,2,'b'),(18,1,'w'),
     (19,5,'b'),(18,6,'w'),(17,3,'b'),(16,2,'w'),(19,2,'b')],

    ]
hosi1W = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')]
    ]
hosi2B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(17,7,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(18,7,'w'),
     (14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
```

```
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(14,4,'w'),
     (18,8,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(18,6,'w'),(15,7,'b'),(16,8,'w'),(14,6,'b'),(15,8,'w'),
     (15,5,'b'),(18,5,'w'),(17,3,'b')]
    ]
komoku1B = [
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(17,9,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'w'),(16,3,'b'),(15,4,'w'),
     (16,7,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(13,3,'b'),(15,5,'w'),
     (15,6,'b'),(14,5,'w'),(16,7,'b'),(12,4,'w'),(16,3,'b'),(15,2,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(13,3,'b'),(13,2,'w'),
     (17,10,'b'),(13,4,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(13,3,'b'),(13,2,'w'),
     (12,2,'b'),(12,3,'w'),(14,2,'b'),(13,4,'w'),(14,3,'b'),(14,4,'w'),
     (15,4,'b'),(15,2,'w'),(13,1,'b'),(15,5,'w'),(16,4,'b'),(15,1,'w'),
     (14,1,'b'),(11,2,'w'),(16,3,'b'),(15,6,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(13,4,'w'),(11,3,'b'),(17,2,'w'),
     (17,9,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w'),(16,3,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w'),(11,3,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w'),(12,4,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'b'),(17,2,'w'),(14,4,'b'),
     (14,3,'w'),(13,4,'b'),(18,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'b'),(13,3,'w'),(16,3,'b'),
     (13,5,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'b'),(17,2,'w'),(14,4,'b'),
     (14,3,'w'),(13,4,'b'),(12,2,'w'),(11,2,'b'),(12,3,'w'),(12,4,'b'),
     (18,3,'w'),(10,5,'b')],
    [(17,4,'b'),(15,3,'w'),(16,6,'b'),(11,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,6,'b'),(17,3,'w'),(18,3,'b'),(16,4,'w'),
     (17,5,'b'),(17,2,'w')],
    [(17,4,'b'),(15,3,'w'),(17,7,'b'),(16,6,'w'),(17,6,'b'),(17,3,'w'),
     (18,3,'b'),(16,4,'w'),(17,5,'b'),(17,2,'w')]
    ]

root.mainloop()
```

となります。但し、このプログラムも、下図のように、メニューの「読込」の「ファイルから定石」を選択した時、



ファイルを指定するマウスのクリックがソフトのキャンバス領域にある場合は



のように、Failure! と表示されますが、気にしないで、実行してください。それぞれのイベントがどのタイミングで実行されているか不明で、これの対策が分かりません。実用上、支障はないです。

課題：ランダムに定石を選ぶのではなく、系統的にすべての定石をチェックできるように修正するには、例えば、

```
josekiopen2 = Menu(menubar, tearoff = False)
menubar.add_cascade(label="読込（すべてチェック）", underline = 0, menu=josekiopen2)
josekiopen2.add_command(label = "ファイルから定石", under = 0, command = file2joseki
josekiopen2.add_command(label = "再度挑戦", under = 0, command = restart2)
```

とメニューを増やし、

```
all_joseki = []
index_joseki = 0
def file2joseki2():
    global M_STOP_FLAG, all_joseki, index_joseki
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        all_joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            all_joseki.append(j_item)
        f.close()
    print('joseki=', all_joseki)
    print('len(joseki)=', len(all_joseki))
    # all_joseki を並べ替える
    for i in range(len(all_joseki)):
```

```
        n = random.randint(0, len(all_joseki)-1)
        temp_list = []
        for k in range(n+1, len(all_joseki)):
            temp_list.append(all_joseki[k])
        for k in range(0, n+1):
            temp_list.append(all_joseki[k])
        all_joseki = temp_list.copy()
    joseki = [all_joseki[index_joseki]]
    basic_start()
    M_STOP_FLAG = False

def restart2():
    global index_joseki, joseki
    index_joseki += 1
    if index_joseki >= len(all_joseki):
        index_joseki = 0
    joseki = [all_joseki[index_joseki]]
    basic_start()
```

と修正すればいいです。1 つの定石のデータが多くなって、候補手を選ぶのが遅く感じたら、グローバル変数 current_joseki を追加し、def get_candidate(num): と def basic_start(): を次のように修正すればいいです。

```
current_joseki = []

def get_candidate(num):
    global current_joseki
    candidate = []
    tmp_joseki = []
    for j_list in current_joseki:
        flag = True
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                tmp_joseki.append(j_list)
                if j_list[num] not in candidate:
                    candidate.append(j_list[num])
```

```
    current_joseki = tmp_joseki.copy()
    return candidate
def basic_start():
    global joseki, num, board, history, n_okiishi, current_joseki
    num = 0
    n_okiishi = 0
    board = basic_board.copy()
    history = []
    current_joseki = joseki.copy()
    # 定石に置き石があるか？
    n = random.randint(0, len(current_joseki)-1)
    j_hist = current_joseki[n]
    i = 0
    while True:
        x,y,cl = j_hist[i]
        if x < 0:
            put_stone(get_z(-x, -y), get_col(cl))
            history.append((x, y, cl))
            i += 1
            n_okiishi += 1
        else:
            break
    if teban.get() == 2:
        candidate = get_candidate(num+n_okiishi)
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        num += 1
    display_board()
```

最終的に私が作ったプログラムは以下のようになります。

```
from tkinter import *
import random
import tkinter.filedialog
import sys, os.path

root = Tk()
##root.option_add("*font", ("FixedSys", 10))
```

```
# variable 用のオブジェクト
teban = IntVar()
teban.set(1)

joseki = []
current_joseki = []
num = 0
history = []
path_name = ""
file_name = ""

basic_board = [3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,3,
               3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3,3]
board = []
B_SIZE = 19
WIDTH = 21
dir4 = [1, WIDTH, -1, -WIDTH]

def get_z(x, y):
```

```
    return y * WIDTH + x
def flip_color(col):
    return 3 - col
def count_liberty_sub(tz, color, p_liberty, p_stone, check_board):
    check_board[tz] = 1
    p_stone += 1
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if check_board[z]: continue
        if board[z] == 0:
            check_board[z] = 1
            p_liberty += 1
        if board[z] == color:
            p_liberty, p_stone = count_liberty_sub(z, color,p_liberty,
                                                   p_stone, check_board)
    return p_liberty, p_stone
def count_liberty(tz, p_liberty, p_stone):
    p_liberty = 0
    p_stone = 0
    check_board = [0] * 21 * 21
    p_liberty, p_stone = count_liberty_sub(tz, board[tz], p_liberty,
                                           p_stone, check_board)
    return p_liberty, p_stone
def take_stone(tz, color):
    global board
    board[tz] = 0
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == color:
            take_stone(z, color)
def put_stone(tz, color):
    global board
    board[tz] = color
    un_col = flip_color(color)
    for d in [1, WIDTH, -1, -WIDTH]:
        z = tz + d
        if board[z] == 0 or board[z] == 3 or board[z] == color: continue
        liberty = 0
        stone = 0
```

```
        liberty, stone = count_liberty(z, liberty, stone)
        if liberty == 0:
            take_stone(z, un_col)

def display_board():
    canvas.create_rectangle(0, 0, 600, 600, fill='lightgreen')
    for k in range(30,600,30):
        canvas.create_line(30,k,570,k)
    for k in range(30,600,30):
        canvas.create_line(k,30,k,570)
    canvas.create_oval(120-3, 120-3, 120+3, 120+3, fill='blue')
    canvas.create_oval(300-3, 120-3, 300+3, 120+3, fill='blue')
    canvas.create_oval(480-3, 120-3, 480+3, 120+3, fill='blue')
    canvas.create_oval(120-3, 300-3, 120+3, 300+3, fill='blue')
    canvas.create_oval(300-3, 300-3, 300+3, 300+3, fill='blue')
    canvas.create_oval(480-3, 300-3, 480+3, 300+3, fill='blue')
    canvas.create_oval(120-3, 480-3, 120+3, 480+3, fill='blue')
    canvas.create_oval(300-3, 480-3, 300+3, 480+3, fill='blue')
    canvas.create_oval(480-3, 480-3, 480+3, 480+3, fill='blue')
    for i in range(1, 20, 1):
        for k in range(1, 20, 1):
            if board[get_z(i,k)] == 1:
                px = 30*(i-1)+30
                py = 30*(k-1)+30
                canvas.create_oval(px-15, py-15, px+15, py+15, fill='black')
            elif board[get_z(i,k)] == 2:
                px = 30*(i-1)+30
                py = 30*(k-1)+30
                canvas.create_oval(px-15, py-15, px+15, py+15, fill='white')
    canvas.create_text(200, 10, text = '                    ',
                       font = ('FixedSys', 20))

n_okiishi = 0
def get_candidate(num):
    global current_joseki
    candidate = []
    tmp_joseki = []
    for j_list in current_joseki:
        flag = True
```

```
        if num < len(j_list):
            for i in range(num):
                if j_list[i] != history[i]:
                    flag = False
                    break
            if flag:
                tmp_joseki.append(j_list)
                if j_list[num] not in candidate:
                    candidate.append(j_list[num])
    current_joseki = tmp_joseki.copy()
    return candidate
def basic_start():
    global joseki, num, board, history, n_okiishi, current_joseki
    num = 0
    n_okiishi = 0
    board = basic_board.copy()
    history = []
    current_joseki = joseki.copy()
    # 定石に置き石があるか？
    n = random.randint(0, len(current_joseki)-1)
    j_hist = current_joseki[n]
    i = 0
    while True:
        x,y,cl = j_hist[i]
        if x < 0:
            put_stone(get_z(-x, -y), get_col(cl))
            history.append((x, y, cl))
            i += 1
            n_okiishi += 1
        else:
            break
    if teban.get() == 2:
        candidate = get_candidate(num+n_okiishi)
        n = random.randint(0, len(candidate)-1)
        tx, ty, cl = candidate[n]
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        num += 1
    display_board()
```

```
def start():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi1B
    else:
        joseki = hosi1W
    basic_start()

def start2():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = hosi2B
    else:
        joseki = hosi2B
    basic_start()
def start3():
    global joseki
    random.seed()
    if teban.get() == 1:
        joseki = komoku1B
    else:
        joseki = komoku1B
    basic_start()

M_STOP_FLAG = False
def file2joseki():
    global M_STOP_FLAG, joseki
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        joseki = []
        for line in f:
            line = line[:-1]
```

```
            if line == '[' or line == ']': continue
            j_item = []
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            joseki.append(j_item)
        f.close()
    print('joseki=', joseki)
    print('len(joseki)=', len(joseki))
    basic_start()
    M_STOP_FLAG = False

def restart():
    basic_start()

all_joseki = []
index_joseki = 0
def file2joseki2():
    global M_STOP_FLAG, all_joseki, index_joseki
    M_STOP_FLAG = True
    global path_name, joseki
    filename = tkinter.filedialog.askopenfilename(initialdir=path_name)
    if filename:
        path_name = os.path.dirname(filename)
        f = open(filename, "r")
        all_joseki = []
        for line in f:
            line = line[:-1]
            if line == '[' or line == ']': continue
            j_item = []
```

```
            j_list = line.split(",")
            for k in range(len(j_list)//3):
                sx = j_list[3*k]
                sy = j_list[3*k+1]
                sc = j_list[3*k+2]
                if k == 0:
                    x = int(sx[2:])
                else:
                    x = int(sx[1:])
                y = int(sy)
                c = sc[0]
                j_item.append((x,y,c))
            all_joseki.append(j_item)
        f.close()
    print('joseki=', all_joseki)
    print('len(joseki)=', len(all_joseki))
    # all_joseki を並べ替える
    for i in range(len(all_joseki)):
        n = random.randint(0, len(all_joseki)-1)
        temp_list = []
        for k in range(n+1, len(all_joseki)):
            temp_list.append(all_joseki[k])
        for k in range(0, n+1):
            temp_list.append(all_joseki[k])
        all_joseki = temp_list.copy()
    joseki = [all_joseki[index_joseki]]
    basic_start()
    M_STOP_FLAG = False

def restart2_1():
    global index_joseki, joseki
    index_joseki += 1
    if index_joseki >= len(all_joseki):
        index_joseki = 0
    joseki = [all_joseki[index_joseki]]
    basic_start()
def restart2_2():
    global index_joseki, joseki
    joseki = [all_joseki[index_joseki]]
```

```
    basic_start()

# メニューバーの生成
menubar = Menu(root)
root.configure(menu = menubar)
# メニューの設定
tebans = Menu(menubar, tearoff = False)
menubar.add_cascade(label="手番", underline = 0, menu=tebans)
hosi = Menu(menubar, tearoff = False)
menubar.add_cascade(label="星", underline = 0, menu=hosi)
kakari = Menu(hosi, tearoff = False)
hosi.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kakari)
kakari.add_command(label = "ツケノビ", under = 0, command = start)
kakari.add_command(label = "ツケオサエ", under = 0, command = start2)
komoku = Menu(menubar, tearoff = False)
menubar.add_cascade(label="小目", underline = 0, menu=komoku)
kogeimakakari = Menu(hosi, tearoff = False)
komoku.add_cascade(label="小ゲイマガカリ", underline = 0, menu=kogeimakakari)
kogeimakakari.add_command(label = "コスミ", under = 0, command = start3)
# Labels
tebans.add_radiobutton(label = '先手', variable = teban, value = 1)
tebans.add_radiobutton(label = '後手', variable = teban, value = 2)
josekiopen = Menu(menubar, tearoff = False)
menubar.add_cascade(label="読込（ランダム）", underline = 0, menu=josekiopen)
josekiopen.add_command(label = "ファイルから定石", under = 0, command = file2joseki)
josekiopen.add_command(label = "再度挑戦", under = 0, command = restart)
josekiopen2 = Menu(menubar, tearoff = False)
menubar.add_cascade(label="読込（すべてチェック）", underline = 0, menu=josekiopen2)
josekiopen2.add_command(label = "ファイルから定石", under = 0, command = file2joseki
josekiopen2.add_command(label = "次の定石", under = 0, command = restart2_1)
josekiopen2.add_command(label = "再度挑戦", under = 0, command = restart2_2)

canvas = Canvas(root, width = 600, height=600)
canvas.pack()

def get_col(cl):
    return 1 if cl == 'b' else 2
def get_pos(x, y):
    px = 30+30*int((x-15)/30)
```

```
    py = 30+30*int((y-15)/30)
    return px, py
def is_candidate(tx, ty, cl, candidate):
    for j in candidate:
        if j == (tx, ty, cl):
            return True
    return False
def on_pressed(event):
    global num, board, history, n_okiishi
    if M_STOP_FLAG == True:
        sx, sy = get_pos(event.x, event.y)
        return
    canvas.create_text(200, 10, text = '                    ',
                       font = ('FixedSys', 20))
    sx, sy = get_pos(event.x, event.y)
    candidate = get_candidate(num+n_okiishi)
    print('candidate=', candidate)
    tx = (sx - 30) // 30 + 1
    ty = (sy - 30) // 30 + 1
    cl = 'b' if teban.get() == 1 else 'w'
    if is_candidate(tx, ty, cl, candidate) == False:
        display_board()
        canvas.create_text(200, 10, text = 'Failure!',
                           font = ('FixedSys', 20))
        return
    put_stone(get_z(tx, ty), get_col(cl))
    history.append((tx, ty, cl))
    num += 1
    display_board()
    candidate = get_candidate(num+n_okiishi)
    if candidate == []:
        display_board()
        canvas.create_text(200, 10, text = 'Congratulations!',
                           font = ('FixedSys', 20))
        return
    n = random.randint(0, len(candidate)-1)
    tx, ty, cl = candidate[n]
    # コンピュータの手番か？
    if (teban.get()==1 and cl=='w') or (teban.get()==2 and cl=='b'):
```

```
        put_stone(get_z(tx, ty), get_col(cl))
        history.append((tx, ty, cl))
        display_board()
        num += 1
        while True:
            candidate = get_candidate(num+n_okiishi)
            if candidate == []:
                canvas.create_text(200, 10, text = 'Congratulations!',
                                   font = ('FixedSys', 20))
                return
            else:
                # 人間が手抜きしたか？
                n = random.randint(0, len(candidate)-1)
                tx, ty, cl = candidate[n]
                if (teban.get()==1 and cl=='w') or (teban.get()==2 and cl=='b'):
                    put_stone(get_z(tx, ty), get_col(cl))
                    history.append((tx, ty, cl))
                    display_board()
                    num += 1
                else:
                    break
    else: # コンピュータが手抜きした
        canvas.create_text(200, 10, text = '手抜き！',
                           font = ('FixedSys', 20))

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

# 定石
##hosi1 = [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
##(17,4,'b'),(17,8,'w'),(14,4,'b')]

hosi1B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(18,2,'w'),
```

```
     (17,2,'b'),(17,1,'w'),(15,2,'b'),(16,3,'w'),(17,3,'b'),(16,1,'w'),
     (18,1,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(17,7,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(14,2,'b'),(17,2,'w'),
     (18,2,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(16,2,'w'),(17,7,'b'),(18,7,'w'),
     (17,8,'b'),(18,8,'w'),(14,2,'b'),(18,2,'w'),(17,2,'b'),(17,3,'w'),
     (16,3,'b'),(15,2,'w'),(18,3,'b'),(17,1,'w'),(15,1,'b'),(17,3,'w'),
     (19,5,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(16,5,'w'),(15,5,'b'),(15,4,'w'),(18,4,'b'),(18,5,'w'),
     (14,4,'b'),(15,3,'w'),(14,3,'b'),(15,2,'w'),(14,2,'b'),(17,2,'w'),
     (17,7,'b'),(18,7,'w'),(17,8,'b'),(18,8,'w'),(18,2,'b'),(18,1,'w'),
     (19,5,'b'),(18,6,'w'),(17,3,'b'),(16,2,'w'),(19,2,'b')],

    ]
hosi1W = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,8,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(15,6,'b'),(17,5,'w'),
     (17,4,'b'),(17,10,'w'),(10,3,'b')]
    ]
hosi2B = [
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(17,7,'w'),(14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(18,7,'w'),
     (14,4,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(15,7,'w'),(17,7,'b'),(17,8,'w'),(18,6,'b'),(14,4,'w'),
     (18,8,'b')],
    [(16,4,'b'),(17,6,'w'),(16,6,'b'),(16,7,'w'),(17,5,'b'),(15,6,'w'),
     (16,5,'b'),(18,6,'w'),(15,7,'b'),(16,8,'w'),(14,6,'b'),(15,8,'w'),
     (15,5,'b'),(18,5,'w'),(17,3,'b')]
    ]
komoku1B = [
```

```
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(17,9,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'w'),(16,3,'b'),(15,4,'w'),
     (16,7,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(13,3,'b'),(15,5,'w'),
     (15,6,'b'),(14,5,'w'),(16,7,'b'),(12,4,'w'),(16,3,'b'),(15,2,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(13,3,'b'),(13,2,'w'),
     (17,10,'b'),(13,4,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'w'),(13,3,'b'),(13,2,'w'),
     (12,2,'b'),(12,3,'w'),(14,2,'b'),(13,4,'w'),(14,3,'b'),(14,4,'w'),
     (15,4,'b'),(15,2,'w'),(13,1,'b'),(15,5,'w'),(16,4,'b'),(15,1,'w'),
     (14,1,'b'),(11,2,'w'),(16,3,'b'),(15,6,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(13,4,'w'),(11,3,'b'),(17,2,'w'),
     (17,9,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w'),(16,3,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w'),(11,3,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(14,4,'b'),(13,3,'w'),(12,4,'b')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(12,3,'b'),(17,2,'w'),(14,4,'b'),
     (14,3,'w'),(13,4,'b'),(18,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'b'),(13,3,'w'),(16,3,'b'),
     (13,5,'w')],
    [(17,4,'b'),(15,3,'w'),(16,5,'b'),(11,3,'b'),(17,2,'w'),(14,4,'b'),
     (14,3,'w'),(13,4,'b'),(12,2,'w'),(11,2,'b'),(12,3,'w'),(12,4,'b'),
     (18,3,'w'),(10,5,'b')],
    [(17,4,'b'),(15,3,'w'),(16,6,'b'),(11,3,'w')],
    [(17,4,'b'),(15,3,'w'),(16,6,'b'),(17,3,'w'),(18,3,'b'),(16,4,'w'),
     (17,5,'b'),(17,2,'w')],
    [(17,4,'b'),(15,3,'w'),(17,7,'b'),(16,6,'w'),(17,6,'b'),(17,3,'w'),
     (18,3,'b'),(16,4,'w'),(17,5,'b'),(17,2,'w')]
    ]

root.mainloop()
```

更に、英単語の学習ソフトのように、間違いを記録しておいて、間違えた定石を優先的に表示するようにすれば、もっと役に立つと思います。後は、藤沢秀行さんの「藤沢秀行囲碁教室２：初段になれる囲碁基本定石」土屋書店や林海峰さんの「定石の急所完全版」マイナビや高尾紳路さんの「基本定石辞典（上・下巻）」日本棋院のような本を見ながら、メニューの定石の追加や定石集データファイルを増やしていけばいいです。素人が作ったソフトですが、お役に立つなら、ご活用ください。また、いいアイデアを思いついたら、自分なりに修正して、より良

いソフトに改良してください。他人に使ってもらう凝ったソフトは C++ で作るのが普通ですが、Python ならこのように手軽にプロトタイプが作れます。将棋も囲碁も勝てれば面白いです。英単語も定石も忘れる前に時間を置かず、定期的に、覚えるまで復習すればいいそうです（西澤ロイ著「頑張らない英単語記憶法」）。数学もプログラミングも外国語も趣味で自分が楽しいと思うことだけをやる分には楽しいです。一度だけの人生です。楽しいことをしましょう！知ることは楽しいことです。知らないということは悲しいことです。