高知大学教育学部の情報数学のテキスト

文責　：　高知大学名誉教授　中村　治

数独のヒントを表示してくれるプログラム (Ruby/tk 版)

数独のヒントを表示してくれるプログラムも Pthon と人気を二分する Ruby で作ってみました。

Ruby で **window** を表示するには、普通、**Ruby/Tk** か **Ruby/Gtk** を使うみたいです。プログラミングを始めるときは、プログラミング言語や関係するソフトをインストールして、環境を整えるのが、私のような素人には、一番難しいです。

Ruby/Gtk は **GTK+ for Windows Runtime Environment** と最新の **ruby25-x64** をインストールし、コマンドラインで

```
$ gem install gtk2
```

を実行すれば、多分、良いです。私のデスクトップパソコンとノートパソコンで、**Ruby/Gtk** が利用可能になりました。

Ruby/Tk は私のデスクトップパソコンでは、**ActiveTcl** と最新の **ruby25-x64** をインストールし、コマンドラインで

```
$ gem install tk
```

を実行すれば、**Ruby/Tk** が利用可能になりました。しかし、私のノートパソコンでは、

```
$ gem install tk
```

を実行しても、エラーが表示され上手くいきません。インターネットで色々調べればわかると思いますが、原因の追究はしていません。**ActiveTcl** と昔の **ruby22-x64** をインストールし、但し、**ruby22-x64** をインストールする時、「**Tcl/Tk** サポートをインストールする」にチェックを入れてインストールすると **Ruby/Tk** が利用可能になりました。 **ruby22-x64** で **Ruby/Gtk** が使える方法は調べていません。

```
$ gem install gtk2
```

が可能なのは、最近の **ruby** だけみたいです。従って、ノートパソコンでは、**Ruby/Tk** を使う時と **Ruby/Gtk** を使う時で、**ruby** を使い分けなくてはならず不便ですが、今はほとんどデスクトップパソコンを使っています。

ここでは、情報量が多い **Ruby/Tk** を使って、**Python Tkinter** で作ったような「数独」のヒントを表示してくれるプログラムを作ってみます。**Ruby/Tk** であれば、**Python Tkinter** のプログラムの真似をして作れるので楽です。

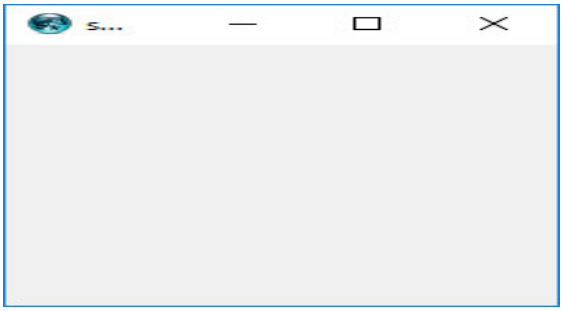
まず、ウィンドウを作ります。

```
require 'tk'
Tk.mainloop
```

という Ruby のプログラムをエディタで作ります。私は Atom というフリーソフトを Ruby プログラミングのためには使っています。sudoku.rb という名前で保存し、コマンドラインで、sudoku.rb を保存したフォルダに移動し、

```
$ ruby sudoku.rb
```

とすれば、



と何もないウインドを表示します。

```
require 'tk'
```

で、Ruby/Tk を使うことを Ruby に教えます。

```
Tk.mainloop
```

は無限ループで、ウインドウ上でマウス入力などを受け付けれるようにします。メソッド mainloop() の括弧は省略してもいいです。

Python/Tkinter と異なり、 Ruby/Tk では、メインウインドを生成しなくてもいいです。そのため、

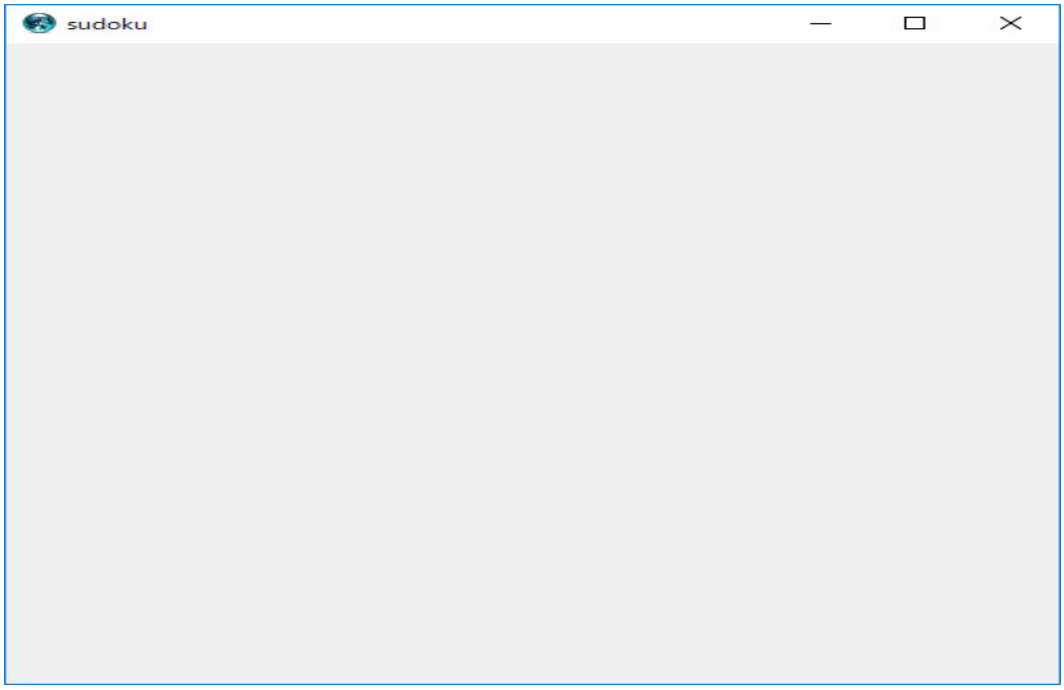
```
Tk.mainloop
```

のように、クラスメソッドを使っていますが、あまり気にしなくていいです。

次に、このウインドにキャンバスを貼り付けます。

```
require 'tk'
BOARD_SIZE = 500
canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack
Tk.mainloop
```

と変更し、実行すると



となります。

```
BOARD_SIZE = 500
```

で、盤のサイズを定義し、

```
canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack
```

で、500 × 500 のキャンバスを生成し、canvas という名前を付け、メソッド pack() で、メインウインドに張り付けています。これは、二つに分けて

```
canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE)
canvas.pack
```

でも良いです。メソッド pack() の括弧は省略してもいいです。
　このプログラムはブロックを使って

```
require 'tk'
BOARD_SIZE = 500
canvas = TkCanvas.new() do
  width BOARD_SIZE
  height BOARD_SIZE
  pack
end
Tk.mainloop
```

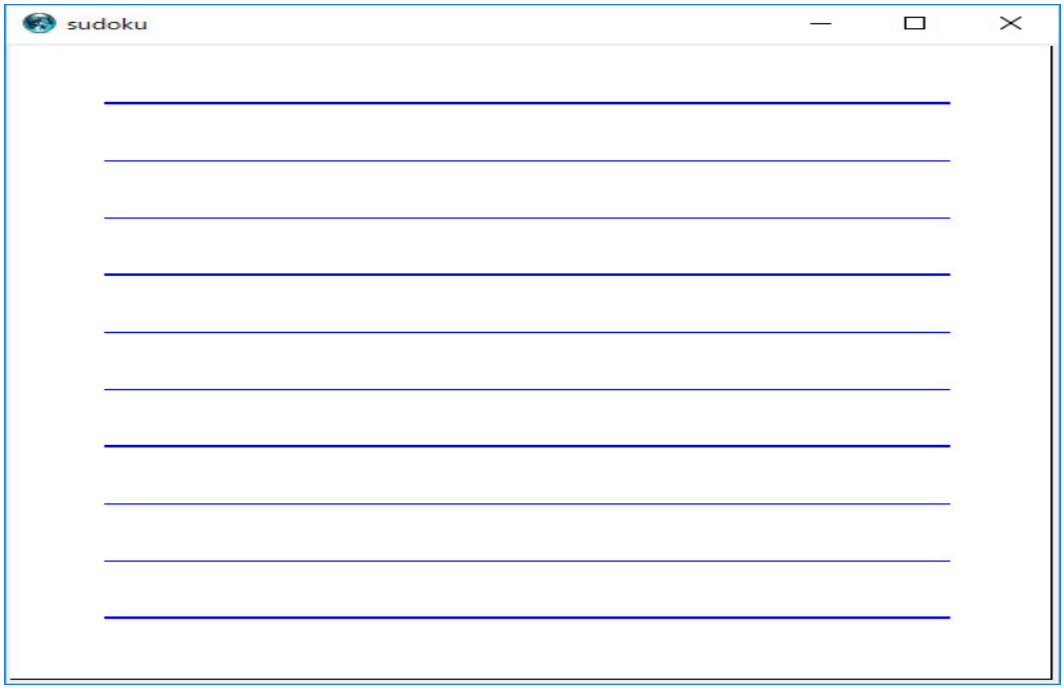
や

```
require 'tk'
BOARD_SIZE = 500
canvas = TkCanvas.new() {
  width BOARD_SIZE
  height BOARD_SIZE
  pack
}
Tk.mainloop
```

と書くこともでき、Ruby は色々な書き方があり、初心者は戸惑いますが、単なる慣れの問題です。
キャンバスに線を描いて、「数独」の盤を描きます。

```
TkcRectangle.new(canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
  fill 'white'
end
K = 9
s = BOARD_SIZE/(K+2)
w = h = (BOARD_SIZE-s*9)/2
for i in 0..9
  if i % 3 == 0
    TkcLine.new(canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new(canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
    end
  end
end
```

を追加します。実行すると



となります。

```
TkcRectangle.new(canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
  fill 'white'
end
```

で、キャンバス canvas にキャンバスと同じ大きさの矩形を描き、白で塗りつぶしています。

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

で、盤を描くための cell の大きさや盤の左上隅の座標を計算しています。メソッド `to_i()` は整数値に変換します。

```
    TkcLine.new(canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
      width 2.0
    end
```

で、キャンバス canvas に青色のサイズ 2.0 の線を描いています。

```
for i in 0..9
    ...
end
```

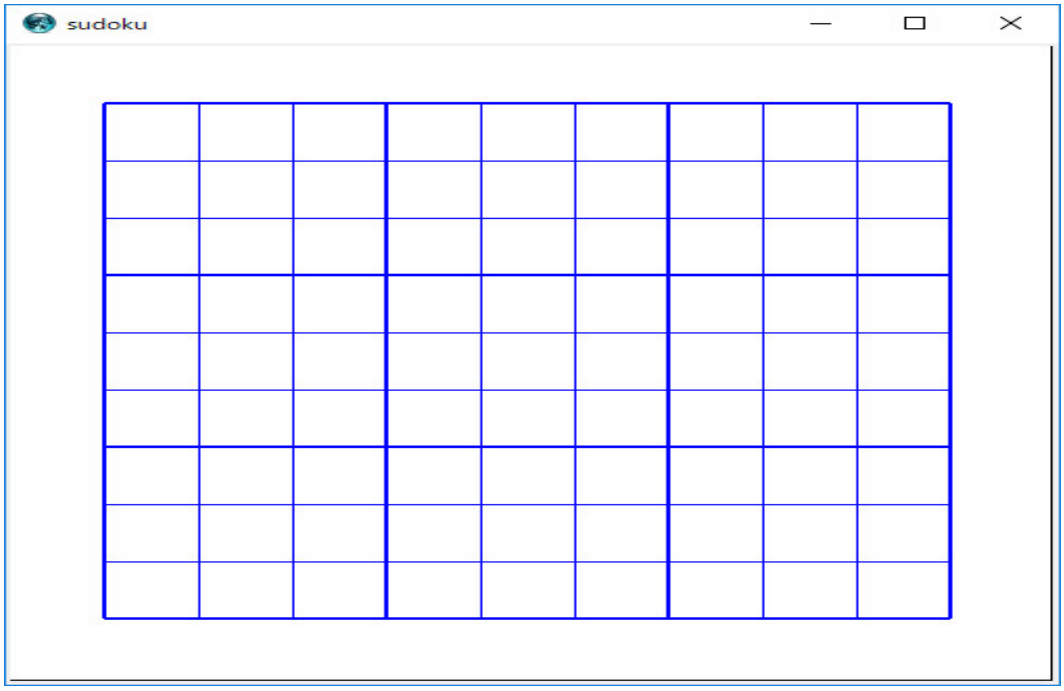
は、Python の

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

と同じです。

```
for i in 0..9
  if i % 3 == 0
    TkcLine.new(canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new(canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
    end
  end
end
```

を追加し、実行すると



となります。

プログラムの全体は

```
require 'tk'
BOARD_SIZE = 500
canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack
TkcRectangle.new(canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
  fill 'white'
```

```
end
K = 9
s = BOARD_SIZE/(K+2)
w = h = (BOARD_SIZE-s*9)/2
for i in 0..9
  if i % 3 == 0
    TkcLine.new(canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new(canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
    end
  end
end
for i in 0..9
  if i % 3 == 0
    TkcLine.new(canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new(canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
    end
  end
end

Tk.mainloop
```

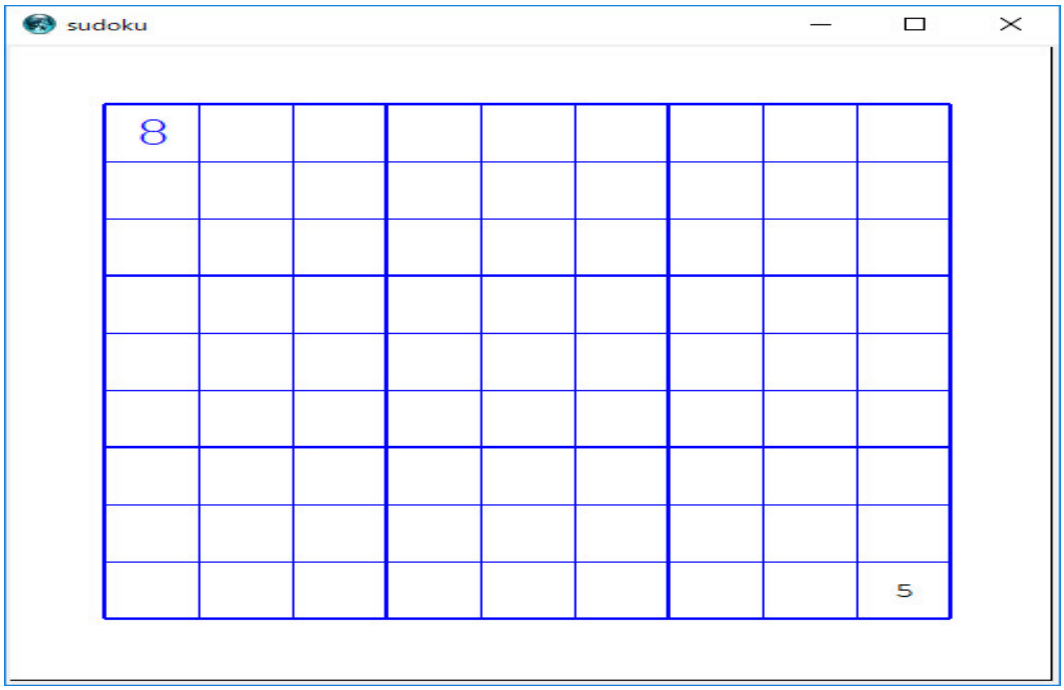
となります。

次に、数字をキャンバスに描きます。

```
f1 = 'courier 24'
c1 = "blue"
i = 0
k = 0
TkcText.new(canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
  text 8.to_s
  font f1
  fill c1
end
f3 = 'courier 12'
```

```
c3 = "black"
i = 8
k = 8
TkcText.new(canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
  text 5.to_s
  font f3
  fill c3
end
```

を追加し、実行すると



となります。

```
TkcText.new(canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
  text 8.to_s
  font f1
  fill c1
end
```

で、キャンバスに文字列を表示します。

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

```
ban = [
[0,1,0,0,0,0,9,0,3],
[0,9,0,0,0,7,0,0,0],
[0,0,0,0,0,0,0,7,1],
```

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

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

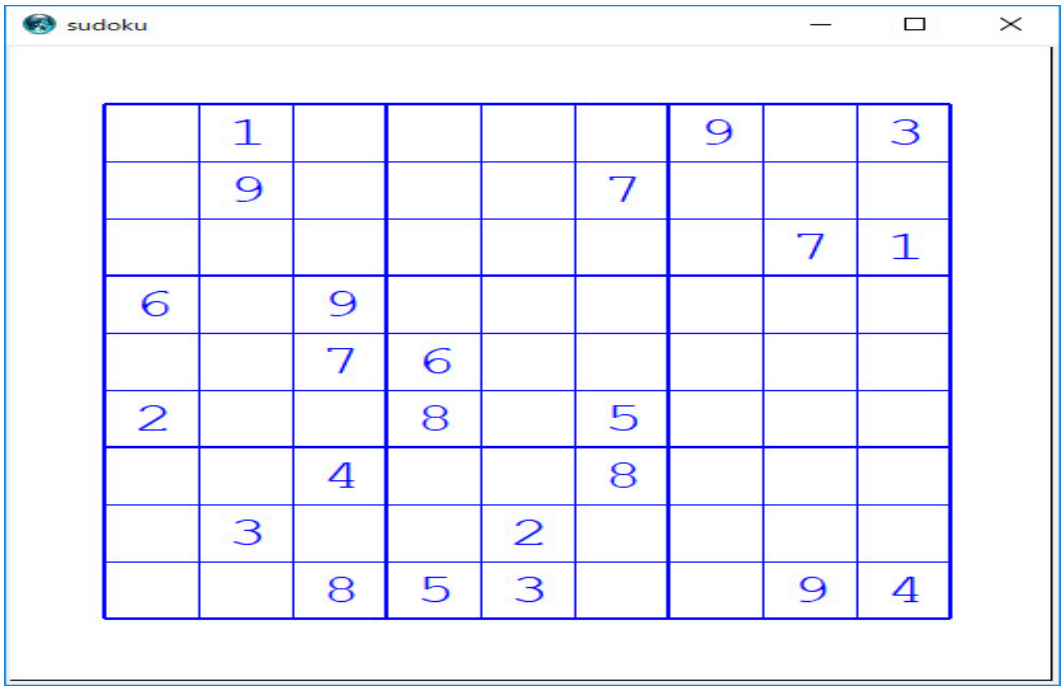
```
f1 = 'courier 24'
c1 = "blue"
i = 0
k = 0
TkcText.new(canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
  text 8.to_s
  font f1
  fill c1
end
f3 = 'courier 12'
c3 = "black"
i = 8
k = 8
TkcText.new(canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
  text 5.to_s
  font f3
  fill c3
end
```

の代わりに

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

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

とし、実行すると



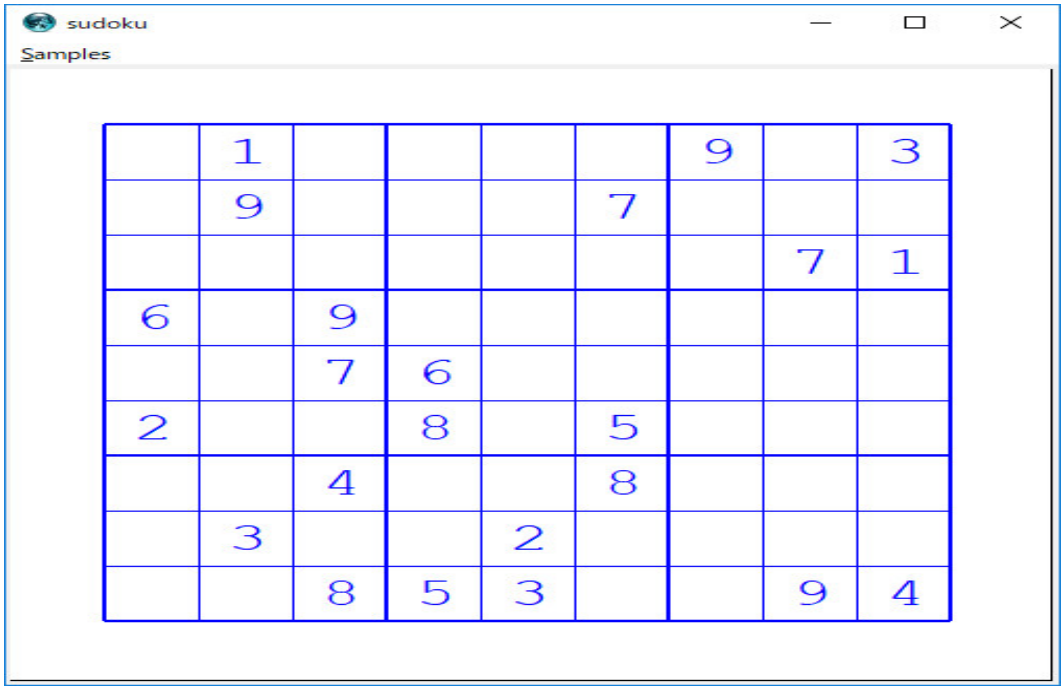
となります。

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

```
def callback
  print "called \n"
end

menubar = TkMenu.new
Tk.root.configure(menu: menubar)
```

```
samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {callback})
```

です。これを追加し、実行すると



となります。「Samples」というメニューができています。「Samples」をクリックすると、「Sample1」というサブメニューを表示します。

```
samples.add_command(label: "Sample1", command: proc {callback})
```

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

```
def callback
  print "called \n"
end
```

と定義されていて、これは called と表示するだけです。メニューの Sample1 を選択すれば、コマンドラインに called と表示される。これをメニューの Sample1 を選択すれば、いま画面に表示されている問題が表示されるようにします。

```
samples.add_command(label: "Sample1", command: proc {callback})
```

を

```
samples.add_command(label: "Sample1", command: proc {f_sample1})
```

と変え、

```
def callback
  print "called \n"
end
```

を

```
$ban = []
def f_sample1
  $ban = [
  [0,1,0,0,0,0,9,0,3],
  [0,9,0,0,0,7,0,0,0],
  [0,0,0,0,0,0,0,7,1],
  [6,0,9,0,0,0,0,0,0],
  [0,0,7,6,0,0,0,0,0],
  [2,0,0,8,0,5,0,0,0],
  [0,0,4,0,0,8,0,0,0],
  [0,3,0,0,2,0,0,0,0],
  [0,0,8,5,3,0,0,9,4]]
  showban()
end
```

と変え、関数 showban() を

```
def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
      end
    end
  end
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
```

```
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
      end
    end
  end
  f1, f2 = "courier 24", "courier 12"
  c1, c2, c3 = "blue", "red", "black"
  for k in 0...9
    for i in 0...9
      if $ban[k][i] > 0
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ($ban[k][i]).to_s
          font f1
          fill c1
        end
      end
    end
  end
end
```

と定義します。

```
$canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack
```

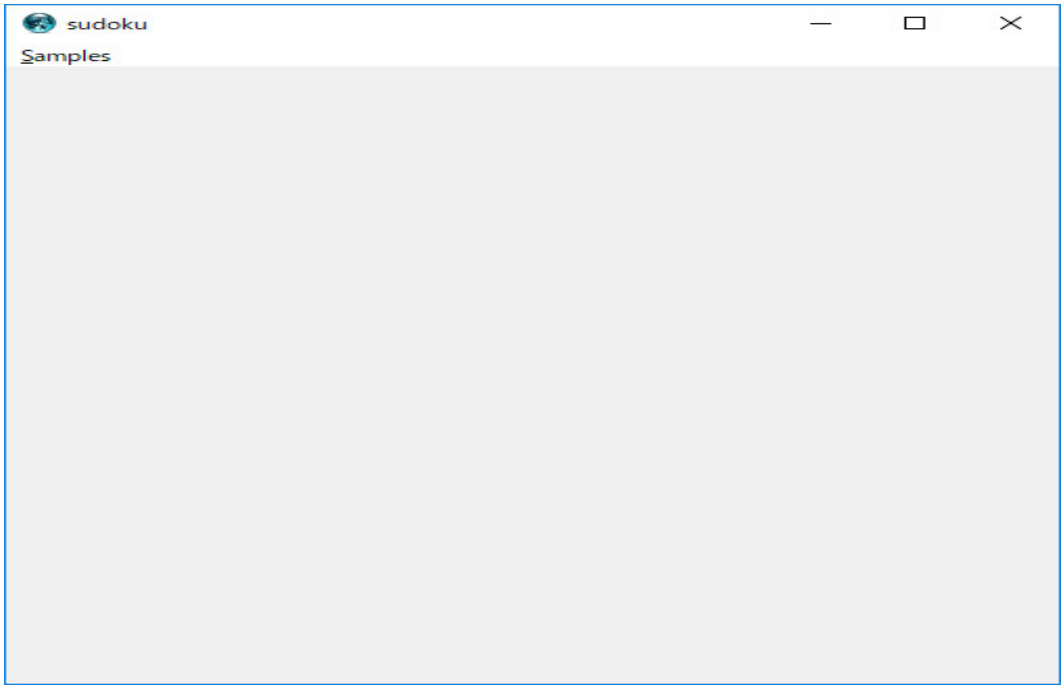
と canvas をグローバル変数 $canvas に変えます。全体のプログラムは

```
require 'tk'
BOARD_SIZE = 500
$canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack

def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
```

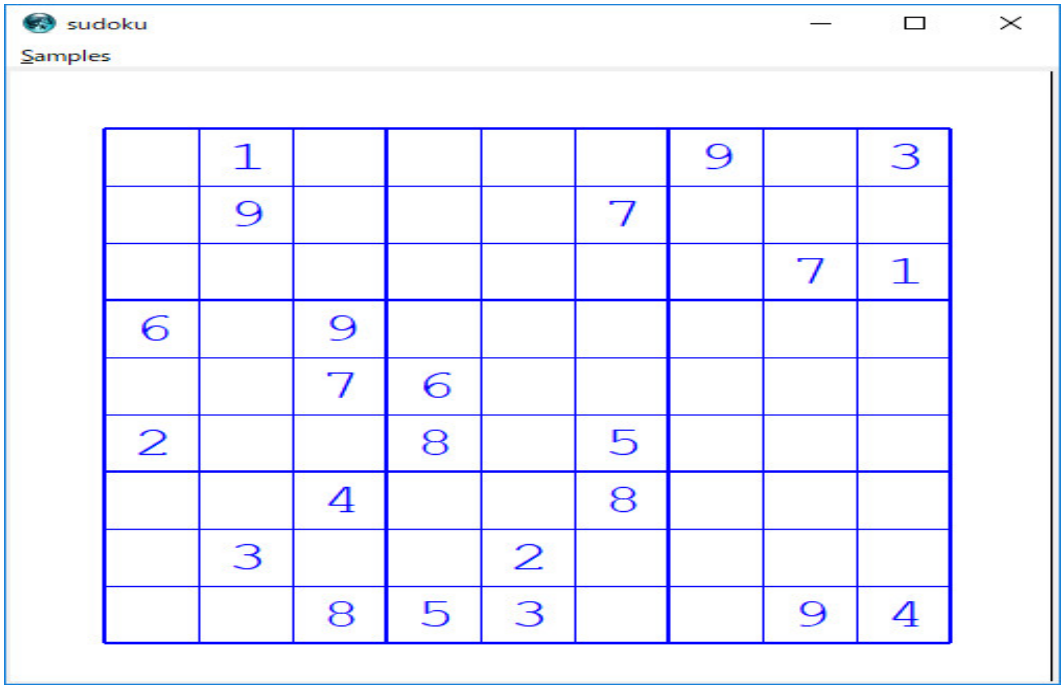
```
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
      end
    end
  end
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
      end
    end
  end
  f1, f2 = "courier 24", "courier 12"
  c1, c2, c3 = "blue", "red", "black"
  for k in 0...9
    for i in 0...9
      if $ban[k][i] > 0
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ($ban[k][i]).to_s
          font f1
          fill c1
        end
      end
    end
  end
end

$ban = []
def f_sample1
  $ban = [
  [0,1,0,0,0,0,9,0,3],
  [0,9,0,0,0,7,0,0,0],
  [0,0,0,0,0,0,0,7,1],
  [6,0,9,0,0,0,0,0,0],
  [0,0,7,6,0,0,0,0,0],
```

```
  [2,0,0,8,0,5,0,0,0],
  [0,0,4,0,0,8,0,0,0],
  [0,3,0,0,2,0,0,0,0],
  [0,0,8,5,3,0,0,9,4]]
  showban()
end

menubar = TkMenu.new
Tk.root.configure(menu: menubar)
samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})

Tk.mainloop
```

となります。実行すると



となります。「Samples」をクリックすると、「Sample1」 というサブメニューを表示します。「Sample1」というサブメニューをクリックすると `f_sample1` という関数が実行され ます。



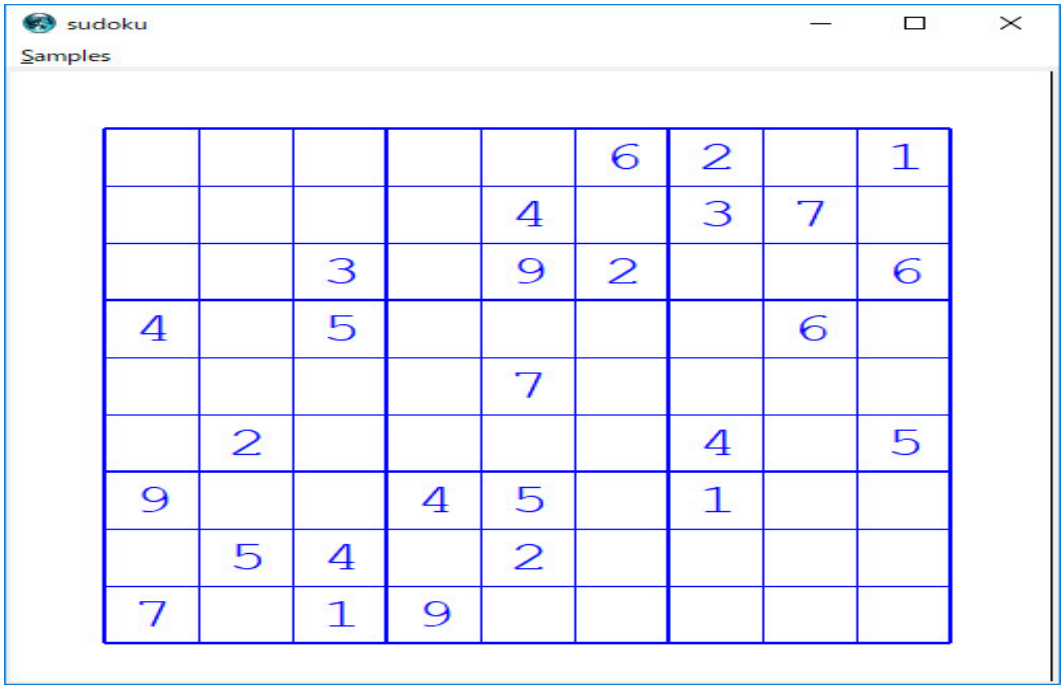
と、問題が表示されます。もう一つ問題を表示できるようにします。

```
samples.add_command(label: "Sample2", command: proc {f_sample2})
```

を追加し、

```
def f_sample2
  $ban = [
    [0,0,0,0,0,6,2,0,1],
    [0,0,0,0,4,0,3,7,0],
    [0,0,3,0,9,2,0,0,6],
    [4,0,5,0,0,0,0,6,0],
    [0,0,0,0,7,0,0,0,0],
    [0,2,0,0,0,0,4,0,5],
    [9,0,0,4,5,0,1,0,0],
    [0,5,4,0,2,0,0,0,0],
    [7,0,1,9,0,0,0,0,0]]
  showban()
end
```

と定義します。「Sample2」というサブメニューをクリックすると `f_sample2` という関数が実行され ます。



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

次に、マウスで解を入力できるようにしましょう。

```
$canvas.bind('Button-1', proc {|x,y| button_press(x, y)}, "%x, %y")
```

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

```
def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    v = Dialog.getValue("123456789")
    $ans[k][i] = v.to_i
    showban()
  end
end
```

と定義します。更に、何か所か修正する必要があります。最初に

```
$ban = []
$ans = []
def f_sample1
```

```
  $ban = [
    [0,1,0,0,0,0,9,0,3],
    [0,9,0,0,0,7,0,0,0],
    [0,0,0,0,0,0,0,7,1],
    [6,0,9,0,0,0,0,0,0],
    [0,0,7,6,0,0,0,0,0],
    [2,0,0,8,0,5,0,0,0],
    [0,0,4,0,0,8,0,0,0],
    [0,3,0,0,2,0,0,0,0],
    [0,0,8,5,3,0,0,9,4]]
  $ans = Marshal.load(Marshal.dump($ban))
  showban()
end
def f_sample2
  $ban = [
    [0,0,0,0,0,6,2,0,1],
    [0,0,0,0,4,0,3,7,0],
    [0,0,3,0,9,2,0,0,6],
    [4,0,5,0,0,0,0,6,0],
    [0,0,0,0,7,0,0,0,0],
    [0,2,0,0,0,0,4,0,5],
    [9,0,0,4,5,0,1,0,0],
    [0,5,4,0,2,0,0,0,0],
    [7,0,1,9,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  showban()
end
```

のように、グローバル変数

```
$ans = []
```

を定義し、

```
  $ans = Marshal.load(Marshal.dump($ban))
```

のように、$ban を　$ans　にディープコピーします。更に

```
module Dialog
  def Dialog.getValue(init)
    var = TkVariable.new("")
    toplevel = TkToplevel.new do
      grab
    end
    TkLabel.new(toplevel) do
      text init
```

```
      font 'helvetica 18'
      pack
    end
    TkEntry.new(toplevel) do
      width 15
      font 'helvetica 18'
      textvariable var
      focus
      pack
    end
    TkButton.new(toplevel) do
      text 'OK'
      font 'helvetica 18'
      command do
        toplevel.grab("release")
        toplevel.destroy
      end
      pack
    end
    toplevel.wait_destroy
    var.value
  end
end
```

と自前のダイアログを定義します。これは

```
    v = Dialog.getValue("123456789")
```

と関数 button_press(x, y) の中で使っています。

```
def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
```

```
        fill 'blue'
      end
    end
  end
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
      end
    end
  end
  f1, f2 = "courier 24", "courier 12"
  c1, c2, c3 = "blue", "red", "black"
  for k in 0...9
    for i in 0...9
      if $ban[k][i] > 0
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ($ban[k][i]).to_s
          font f1
          fill c1
        end
      elsif $ans[k][i] > 0
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ($ans[k][i]).to_s
          font f1
          fill c2
        end
      end
    end
  end
end
```

のように、関数 showban() も修正します。結局、プログラムの全体は

```
require 'tk'

module Dialog
  def Dialog.getValue(init)
    var = TkVariable.new("")
```

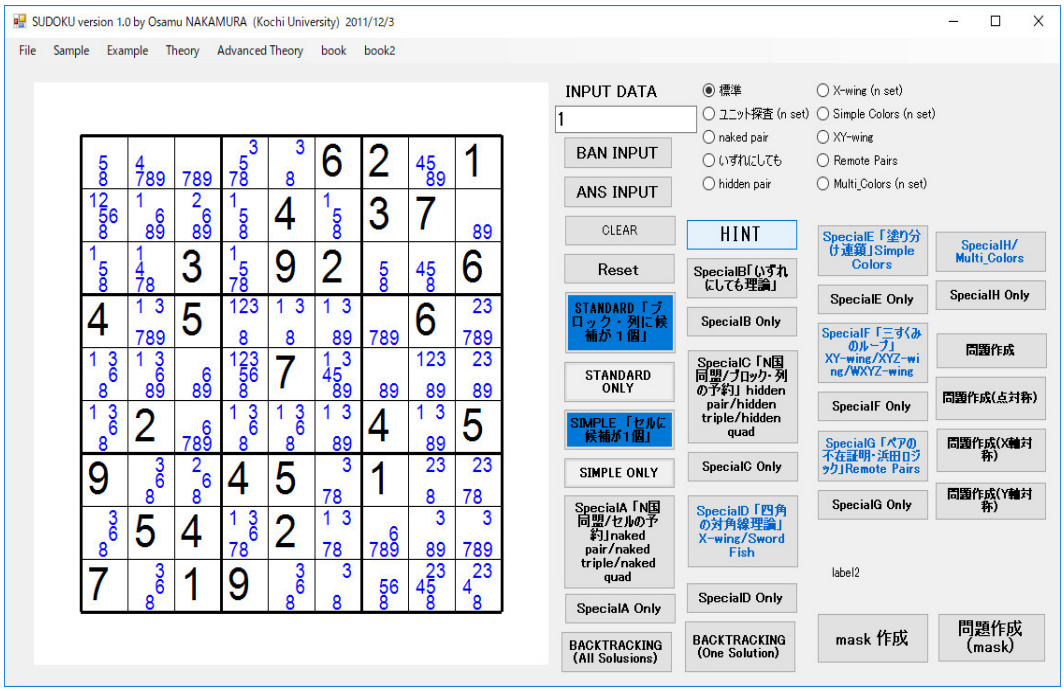
```
    toplevel = TkToplevel.new do
      grab
    end
    TkLabel.new(toplevel) do
      text init
      font 'helvetica 18'
      pack
    end
    TkEntry.new(toplevel) do
      width 15
      font 'helvetica 18'
      textvariable var
      focus
      pack
    end
    TkButton.new(toplevel) do
      text 'OK'
      font 'helvetica 18'
      command do
        toplevel.grab("release")
        toplevel.destroy
      end
      pack
    end
    toplevel.wait_destroy
    var.value
  end
end

BOARD_SIZE = 500
$canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack

def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
```

```
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
      end
    end
  end
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
      end
    end
  end
  f1, f2 = "courier 24", "courier 12"
  c1, c2, c3 = "blue", "red", "black"
  for k in 0...9
    for i in 0...9
      if $ban[k][i] > 0
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ($ban[k][i]).to_s
          font f1
          fill c1
        end
      elsif $ans[k][i] > 0
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ($ans[k][i]).to_s
          font f1
          fill c2
        end
      end
    end
  end
end

$ban = []
$ans = []
```

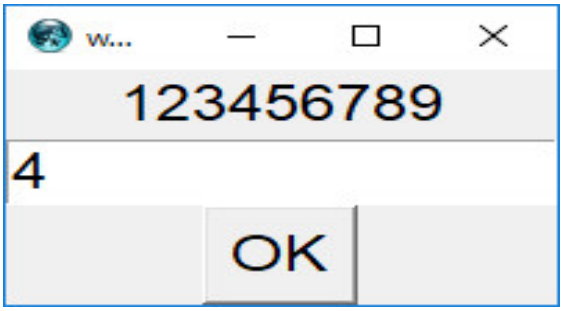
```
def f_sample1
  $ban = [
    [0,1,0,0,0,0,9,0,3],
    [0,9,0,0,0,7,0,0,0],
    [0,0,0,0,0,0,0,7,1],
    [6,0,9,0,0,0,0,0,0],
    [0,0,7,6,0,0,0,0,0],
    [2,0,0,8,0,5,0,0,0],
    [0,0,4,0,0,8,0,0,0],
    [0,3,0,0,2,0,0,0,0],
    [0,0,8,5,3,0,0,9,4]]
  $ans = Marshal.load(Marshal.dump($ban))
  showban()
end
def f_sample2
  $ban = [
    [0,0,0,0,0,6,2,0,1],
    [0,0,0,0,4,0,3,7,0],
    [0,0,3,0,9,2,0,0,6],
    [4,0,5,0,0,0,0,6,0],
    [0,0,0,0,7,0,0,0,0],
    [0,2,0,0,0,0,4,0,5],
    [9,0,0,4,5,0,1,0,0],
    [0,5,4,0,2,0,0,0,0],
    [7,0,1,9,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  showban()
end

def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    v = Dialog.getValue("123456789")
    $ans[k][i] = v.to_i
    showban()
  end
end
```

```
menubar = TkMenu.new
Tk.root.configure(menu: menubar)
samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})
samples.add_command(label: "Sample2", command: proc {f_sample2})

$canvas.bind('Button-1', proc {|x,y| button_press(x, y)}, "%x, %y")

Tk.mainloop
```

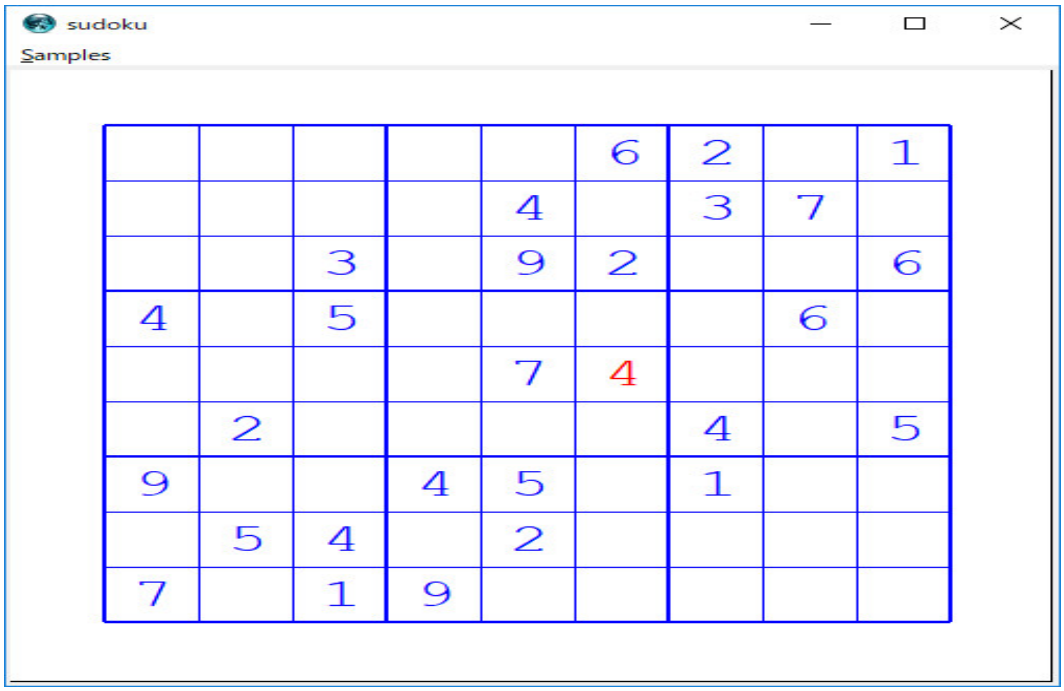
になります。私が C++ で作ったプログラムで調べると



となりますから、中央のブロックで４が入りうるのは７の右隣だけですから、中央のブロックの７の右隣は４です。ここに答えを入れてみましょう。中央のブロック の ７の右隣のセルをクリックします。とダイアログボックスが開きます。ラベルの 123456789　はこのセルに入りうる数字です。今後、ヒントの数字を表示するようになったら、この数字をセルの状況に応じて変化させるようにプログラムを修正します。4 を入力します。

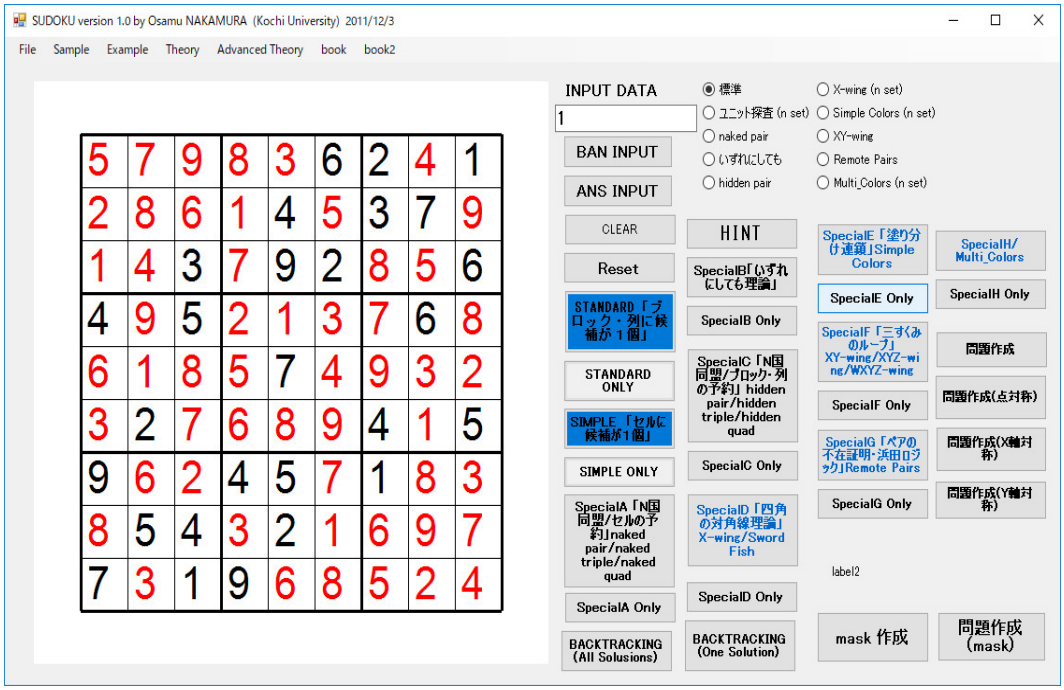


となります。「OK」のボタンをクリックします。

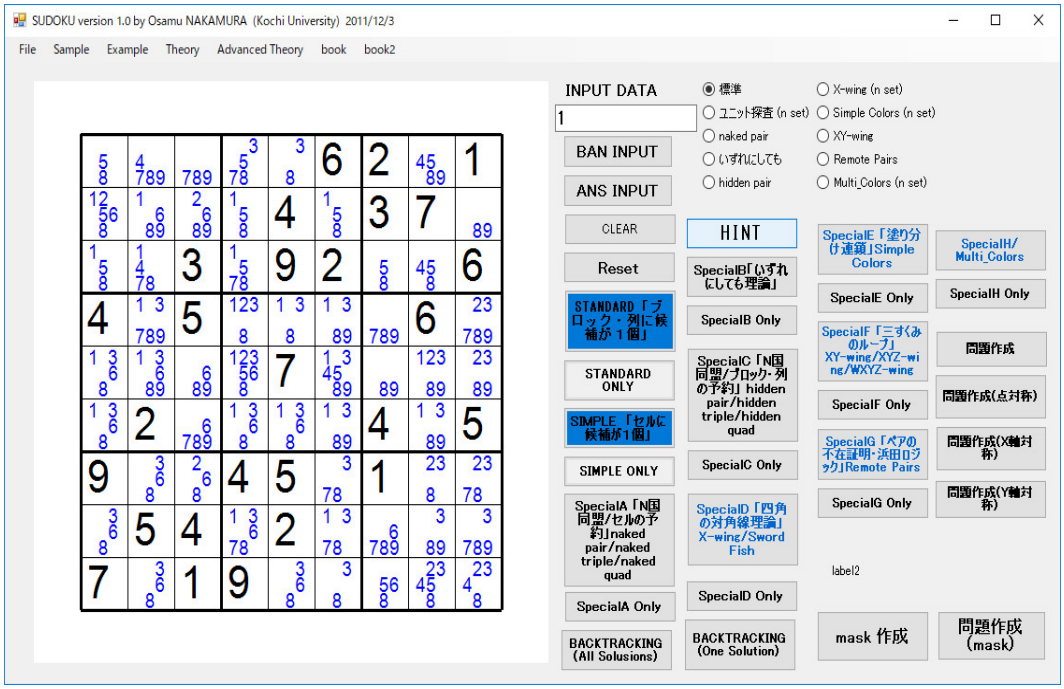


4 が表示されました。

この問題の解は



です。「SpecialE Only」で解けます。このボタンは私が「数独」の解法をインターネットで見つけた順にプログラミングしているだけで、易しい順にプログラミングしているわけではありません。



INPUT DATA 1

● 標準

のようなヒントを表示するようにプログラムを修正しましょう。ヒントの小さい数字は、1 から 9 までの数字のうち、縦の列、横の行、ブロックに現れる数字 を除いた、そのセルに入りうる可能性のある数字を表しています。Python 版のプログラミングで失敗したので、パソコンで問題を解かせるときにも通用するよう最初から慎重にプログラミングするようにします。

```
$box = [
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]]
]
$cand = [
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
```

```
  [[], [], [], [], [], [], [], [], []]
]

def set_cand
  for k in 0...9
    for j in 0...9
      $box[k/3*3+j/3][k%3*3+j%3][0] = k
      $box[k/3*3+j/3][k%3*3+j%3][1] = j
    end
  end
  for k in 0...9
    for j in 0...9
      numYoko = []
      for i in 0...9
        if $ans[k][i] > 0
          numYoko.push($ans[k][i])
        end
      end
      numTate = []
      for i in 0...9
        if $ans[i][j] > 0
          numTate.push($ans[i][j])
        end
      end
      numBlock = []
      n = k/3*3+j/3
      for i in 0...9
        if $ans[$box[n][i][0]][$box[n][i][1]] > 0
          numBlock.push($ans[$box[n][i][0]][$box[n][i][1]])
        end
      end
      $cand[k][j] = []
      for n in 1..9
        $cand[k][j].push(n)
      end
      $cand[k][j] = $cand[k][j] - (numYoko | numTate | numBlock)
    end
  end
end
```

と定義します。

ここで

```
$box = [
```

```
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]]
]
```

は

```
  for k in 0...9
    for j in 0...9
      $box[k/3*3+j/3][k%3*3+j%3][0] = k
      $box[k/3*3+j/3][k%3*3+j%3][1] = j
    end
  end
```

で、実際の値を定義しています。ブロック番号 $n(0 \leq n \leq 8)$ の $i(0 \leq i \leq 8)$ 番目のセルの座標 (k, j) を `k=$box[n][i][0]` と `j=$box[n][i][1]` で与えることが出来るようにするためです。座標 (k, j) のセルのブロック番号は `k/3*3+j/3` で与えられ、`k%3*3+j%3` 番目のセルになります。これを使えば、for 文が一つで済みます。

リスト `$cand` は

```
$cand = [
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []]
]
```

としていますが、Python には、集合を表すデータ構造が準備されていますが、Ruby ではリストで代用します。リストに集合演算が定義されています。`$cand[k][j]` に、

```
def set_cand
  for k in 0...9
    for j in 0...9
      $box[k/3*3+j/3][k%3*3+j%3][0] = k
      $box[k/3*3+j/3][k%3*3+j%3][1] = j
```

```
    end
  end
  for k in 0...9
    for j in 0...9
      numYoko = []
      for i in 0...9
        if $ans[k][i] > 0
          numYoko.push($ans[k][i])
        end
      end
      numTate = []
      for i in 0...9
        if $ans[i][j] > 0
          numTate.push($ans[i][j])
        end
      end
      numBlock = []
      n = k/3*3+j/3
      for i in 0...9
        if $ans[$box[n][i][0]][$box[n][i][1]] > 0
          numBlock.push($ans[$box[n][i][0]][$box[n][i][1]])
        end
      end
      $cand[k][j] = []
      for n in 1..9
        $cand[k][j].push(n)
      end
      $cand[k][j] = $cand[k][j] - (numYoko | numTate | numBlock)
    end
  end
end
```

で、座標 (k, j) のセルにセットしうる候補者をセットします。

そして、showban() を

```
      else
        ss = ''
        ss1 = ''
        ss2 = ''
        $cand[k][i].each_with_index do |elem, ind|
          if ind == 3
            ss1 = ss
            ss = ''
          elsif ind == 6
```

```
          ss2 = ss
          ss = ''
        end
        ss += elem.to_s
      end
      if ss1 != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.25)*s) do
          text ss1
          font f2
          fill c3
        end
      end
      if ss2 != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ss2
          font f2
          fill c3
        end
      end
      if ss1 != '' and ss2 != '' and ss != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
          text ss
          font f2
          fill c3
        end
      elsif ss1 != '' and ss != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
          text ss
          font f2
          fill c3
        end
      else
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ss
          font f2
          fill c3
        end
      end
```

を追加して、次のように修正します。

```
def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
```

```
end
k = 9
s = BOARD_SIZE/(k+2)
w = h = (BOARD_SIZE-s*9)/2
for i in 0..9
  if i % 3 == 0
    TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
      fill 'blue'
    end
  end
end
for i in 0..9
  if i % 3 == 0
    TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
    end
  end
end
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in 0...9
  for i in 0...9
    if $ban[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ban[k][i]).to_s
        font f1
        fill c1
      end
    elsif $ans[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ans[k][i]).to_s
        font f1
        fill c2
```

```
  end
else
  ss = ''
  ss1 = ''
  ss2 = ''
  $cand[k][i].each_with_index do |elem, ind|
    if ind == 3
      ss1 = ss
      ss = ''
    elsif ind == 6
      ss2 = ss
      ss = ''
    end
    ss += elem.to_s
  end
  if ss1 != ''
    TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.25)*s) do
      text ss1
      font f2
      fill c3
    end
  end
  if ss2 != ''
    TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
      text ss2
      font f2
      fill c3
    end
  end
  if ss1 != '' and ss2 != '' and ss != ''
    TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
      text ss
      font f2
      fill c3
    end
  elsif ss1 != '' and ss != ''
    TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
      text ss
      font f2
      fill c3
    end
  else
    TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
```

```
            text ss
            font f2
            fill c3
          end
        end
      end
    end
  end
end
```

そして、f_sample1 と f_sample2 に

```
  set_cand
```

を追加して

```
def f_sample1
  $ban = [
    [0,1,0,0,0,0,9,0,3],
    [0,9,0,0,0,7,0,0,0],
    [0,0,0,0,0,0,0,7,1],
    [6,0,9,0,0,0,0,0,0],
    [0,0,7,6,0,0,0,0,0],
    [2,0,0,8,0,5,0,0,0],
    [0,0,4,0,0,8,0,0,0],
    [0,3,0,0,2,0,0,0,0],
    [0,0,8,5,3,0,0,9,4]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
```

のように修正します。結局、プログラムの全体は

```
require 'tk'

module Dialog
  def Dialog.getValue(init)
    var = TkVariable.new("")
    toplevel = TkToplevel.new do
      grab
    end
    TkLabel.new(toplevel) do
      text init
      font 'helvetica 18'
      pack
```

```
    end
    TkEntry.new(toplevel) do
      width 15
      font 'helvetica 18'
      textvariable var
      focus
      pack
    end
    TkButton.new(toplevel) do
      text 'OK'
      font 'helvetica 18'
      command do
        toplevel.grab("release")
        toplevel.destroy
      end
      pack
    end
    toplevel.wait_destroy
    var.value
  end
end

BOARD_SIZE = 500
$canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack

def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
      end
    end
```

```
end
for i in 0..9
  if i % 3 == 0
    TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
      width 2.0
    end
  else
    TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
      fill 'blue'
    end
  end
end
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in 0...9
  for i in 0...9
    if $ban[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ban[k][i]).to_s
        font f1
        fill c1
      end
    elsif $ans[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ans[k][i]).to_s
        font f1
        fill c2
      end
    else
      ss = ''
      ss1 = ''
      ss2 = ''
      $cand[k][i].each_with_index do |elem, ind|
        if ind == 3
          ss1 = ss
          ss = ''
        elsif ind == 6
          ss2 = ss
          ss = ''
        end
        ss += elem.to_s
      end
```

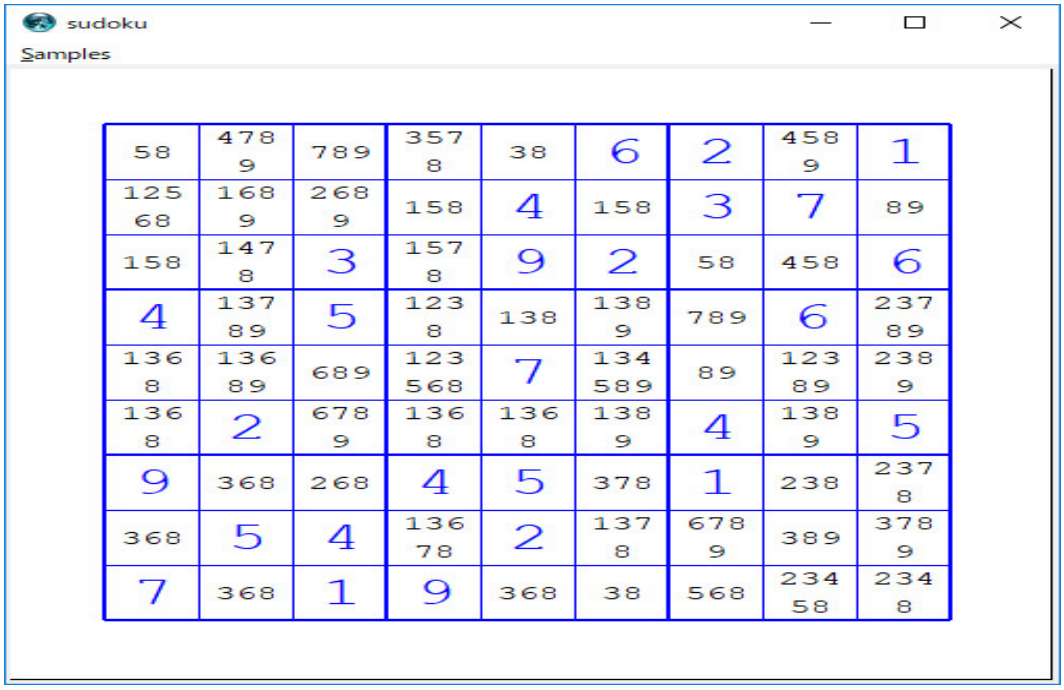
```
        if ss1 != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.25)*s) do
            text ss1
            font f2
            fill c3
          end
        end
        if ss2 != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
            text ss2
            font f2
            fill c3
          end
        end
        if ss1 != '' and ss2 != '' and ss != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
            text ss
            font f2
            fill c3
          end
        elsif ss1 != '' and ss != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
            text ss
            font f2
            fill c3
          end
        else
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
            text ss
            font f2
            fill c3
          end
        end
      end
    end
  end
end

$box = [
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
```

```
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]]
]
$cand = [
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []]
]

def set_cand
  for k in 0...9
    for j in 0...9
      $box[k/3*3+j/3][k%3*3+j%3][0] = k
      $box[k/3*3+j/3][k%3*3+j%3][1] = j
    end
  end
  for k in 0...9
    for j in 0...9
      numYoko = []
      for i in 0...9
        if $ans[k][i] > 0
          numYoko.push($ans[k][i])
        end
      end
      numTate = []
      for i in 0...9
        if $ans[i][j] > 0
          numTate.push($ans[i][j])
        end
      end
      numBlock = []
      n = k/3*3+j/3
      for i in 0...9
        if $ans[$box[n][i][0]][$box[n][i][1]] > 0
```

```
          numBlock.push($ans[$box[n][i][0]][$box[n][i][1]])
        end
      end
      $cand[k][j] = []
      for n in 1..9
        $cand[k][j].push(n)
      end
      $cand[k][j] = $cand[k][j] - (numYoko | numTate | numBlock)
    end
  end
end

$ban = []
$ans = []
def f_sample1
  $ban = [
    [0,1,0,0,0,0,9,0,3],
    [0,9,0,0,0,7,0,0,0],
    [0,0,0,0,0,0,0,7,1],
    [6,0,9,0,0,0,0,0,0],
    [0,0,7,6,0,0,0,0,0],
    [2,0,0,8,0,5,0,0,0],
    [0,0,4,0,0,8,0,0,0],
    [0,3,0,0,2,0,0,0,0],
    [0,0,8,5,3,0,0,9,4]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
def f_sample2
  $ban = [
    [0,0,0,0,0,6,2,0,1],
    [0,0,0,0,4,0,3,7,0],
    [0,0,3,0,9,2,0,0,6],
    [4,0,5,0,0,0,0,6,0],
    [0,0,0,0,7,0,0,0,0],
    [0,2,0,0,0,0,4,0,5],
    [9,0,0,4,5,0,1,0,0],
    [0,5,4,0,2,0,0,0,0],
    [7,0,1,9,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
```

```
end

def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    v = Dialog.getValue("123456789")
    $ans[k][i] = v.to_i
    showban()
  end
end


menubar = TkMenu.new
Tk.root.configure(menu: menubar)
samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})
samples.add_command(label: "Sample2", command: proc {f_sample2})

$canvas.bind('Button-1', proc {|x,y| button_press(x, y)}, "%x, %y")

Tk.mainloop
```

となります。実行すると



のように表示されます。

更に、

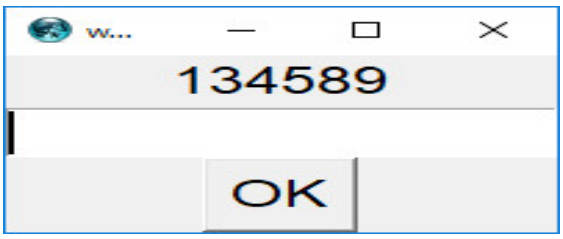
```
def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    v = Dialog.getValue("123456789")
    $ans[k][i] = v.to_i
    showban()
  end
end
```

を

```
def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    ss = ''
    for n in $cand[k][i]
```

```
      ss += n.to_s
    end
    v = Dialog.getValue(ss)
    $ans[k][i] = v.to_i
    set_cand
    showban()
  end
end
```

と修正します。上の局面で、中央の 7 の右隣のセルをクリックすると



のように,、入力すべき数字の列がラベルに表示されます。

数字を打ち込むことによるヒント（候補者）の修正は最小限度にとどめることも出来ますが、打ち間違えて、再度打ち直すこともあるので、

```
    set_cand
```

と、全体を書き直すようにしています。

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

関数 fun_sample3() を

```
def f_sample3
  $ban = [
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
```

と定義し、

```
samples.add_command(label: "Sample3", command: proc {f_sample3})
```

を追加します。更に、ラジオボタンを２つ追加します。

```
$action = TkVariable.new(1)
```

と定義し、

```
TkRadiobutton.new(text: 'ban[][]', variable: $action, value: 0).pack
TkRadiobutton.new(text: 'ans[][]', variable: $action, value: 1).pack
```

を追加します。関数 button_press(x, y)

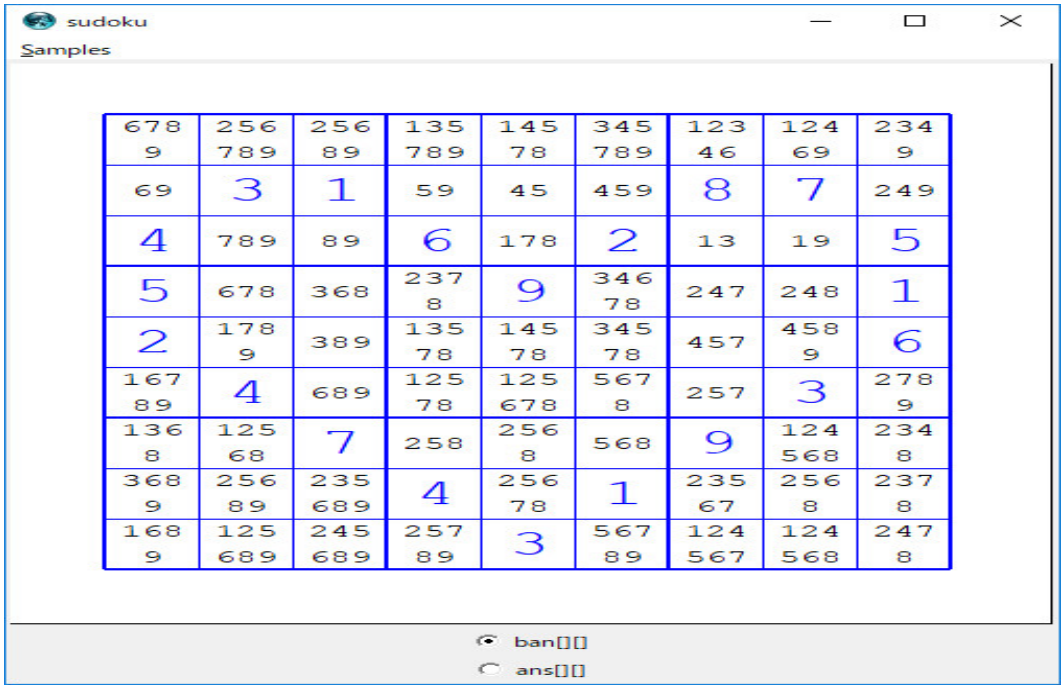
```
def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    ss = ''
    for n in $cand[k][i]
      ss += n.to_s
    end
    v = Dialog.getValue(ss)
    $ans[k][i] = v.to_i
    set_cand
    showban()
  end
end
```

を

```
def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    ss = ''
    for n in $cand[k][i]
      ss += n.to_s
    end
    v = Dialog.getValue(ss)
    if $action.value.to_i == 0
      $ban[k][i] = v.to_i
```

```
      $ans[k][i] = v.to_i
    else
      $ans[k][i] = v.to_i
    end
    set_cand
    showban()
  end
end
```

と修正します。実行すると



6789	256789	25689	135789	14578	345789	12346	12469	2349
69	3	1	59	45	459	8	7	249
4	789	89	6	178	2	13	19	5
5	678	368	2378	9	34678	247	248	1
2	1789	389	13578	14578	34578	457	4589	6
16789	4	689	12578	125678	5678	257	3	2789
1368	12568	7	258	2568	568	9	124568	2348
3689	25689	235689	4	25678	1	23567	2568	2378
1689	125689	245689	25789	3	56789	124567	124568	2478

のように、データを入力することが出来ます。これは私の C++ のプログラムが作った問題です。13 分ぐらいかかりました。難しい問題です。ラジオボタンの ans[][] をクリックすれば、解を入力するように切り替えることが出来ます。

次に、局面を保存できるようにしましょう。

```
$pathname = ""
$filename = ""
```

を追加する。メニューの定義に

```
files =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Files', under: 0, menu: files)
files.add_command(label: "Save", command: proc {save_ban})
```

を追加し、メニューの定義の全体は

```
menubar = TkMenu.new
Tk.root.configure(menu: menubar)
```

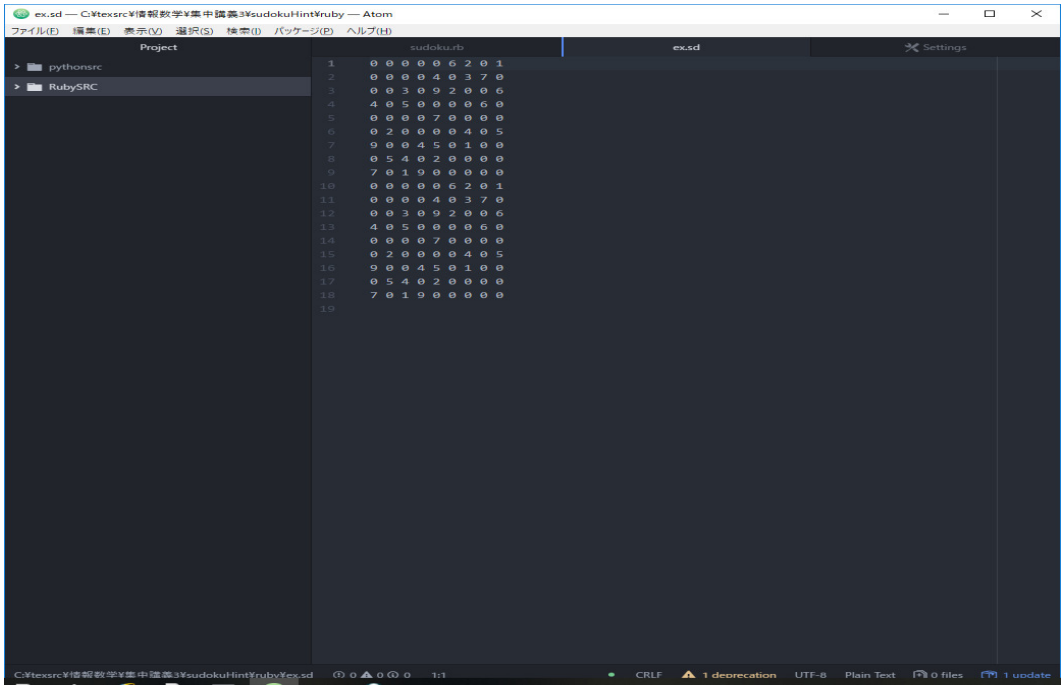
```
files =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Files', under: 0, menu: files)
files.add_command(label: "Save", command: proc {save_ban})

samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})
samples.add_command(label: "Sample2", command: proc {f_sample2})
samples.add_command(label: "Sample3", command: proc {f_sample3})
```

になるようにします。

関数 save_ban() は

```
def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    ss = ''
    for n in $cand[k][i]
      ss += n.to_s
    end
    v = Dialog.getValue(ss)
    if $action.value.to_i == 0
      $ban[k][i] = v.to_i
      $ans[k][i] = v.to_i
    else
      $ans[k][i] = v.to_i
    end
    set_cand
    showban()
  end
end
```

です。実行すると



のように、作られたファイルは、単に $ban[][] と $ans[][] の値を並べているだけです。

保存したデータを読み込むことが出来るようにしましょう。メニューの定義に

```
files.add_command(label: "Open", command: proc {open_ban})
```

を追加し、メニューの定義の全体は

```
menubar = TkMenu.new
Tk.root.configure(menu: menubar)
files =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Files', under: 0, menu: files)
files.add_command(label: "Save", command: proc {save_ban})
files.add_command(label: "Open", command: proc {open_ban})

samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})
samples.add_command(label: "Sample2", command: proc {f_sample2})
samples.add_command(label: "Sample3", command: proc {f_sample3})
```

となります。関数 open_ban()

```
def open_ban
  filetype = "{All {*}}"
  $filename = Tk.getOpenFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
```

```
    f = open( $filename , "r" )
    for cnt in 0...18
      line = f.readline
      s = line.split()
      nlist = s.map{|item| item.to_i}
      if cnt < 9
        $ban[cnt] = nlist
      else
        $ans[cnt-9] = nlist
      end
    end
    f.close
    set_cand
    showban()
  end
end
```

を追加します。全体のプログラムは

```
require 'tk'

module Dialog
  def Dialog.getValue(init)
    var = TkVariable.new("")
    toplevel = TkToplevel.new do
      grab
    end
    TkLabel.new(toplevel) do
      text init
      font 'helvetica 18'
      pack
    end
    TkEntry.new(toplevel) do
      width 15
      font 'helvetica 18'
      textvariable var
      focus
      pack
    end
    TkButton.new(toplevel) do
      text 'OK'
      font 'helvetica 18'
      command do
        toplevel.grab("release")
```

```
        toplevel.destroy
      end
      pack
    end
    toplevel.wait_destroy
    var.value
  end
end

BOARD_SIZE = 500
$canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack

def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
      end
    end
  end
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
      end
    end
  end
```

```
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in 0...9
  for i in 0...9
    if $ban[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ban[k][i]).to_s
        font f1
        fill c1
      end
    elsif $ans[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ans[k][i]).to_s
        font f1
        fill c2
      end
    else
      ss = ''
      ss1 = ''
      ss2 = ''
      $cand[k][i].each_with_index do |elem, ind|
        if ind == 3
          ss1 = ss
          ss = ''
        elsif ind == 6
          ss2 = ss
          ss = ''
        end
        ss += elem.to_s
      end
      if ss1 != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.25)*s) do
          text ss1
          font f2
          fill c3
        end
      end
      if ss2 != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
          text ss2
          font f2
          fill c3
        end
```

```
        end
        if ss1 != '' and ss2 != '' and ss != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
            text ss
            font f2
            fill c3
          end
        elsif ss1 != '' and ss != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
            text ss
            font f2
            fill c3
          end
        else
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
            text ss
            font f2
            fill c3
          end
        end
      end
    end
  end
end

$box = [
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]]
]
$cand = [
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
```

```
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []]
]

def set_cand
  for k in 0...9
    for j in 0...9
      $box[k/3*3+j/3][k%3*3+j%3][0] = k
      $box[k/3*3+j/3][k%3*3+j%3][1] = j
    end
  end
  for k in 0...9
    for j in 0...9
      numYoko = []
      for i in 0...9
        if $ans[k][i] > 0
          numYoko.push($ans[k][i])
        end
      end
      numTate = []
      for i in 0...9
        if $ans[i][j] > 0
          numTate.push($ans[i][j])
        end
      end
      numBlock = []
      n = k/3*3+j/3
      for i in 0...9
        if $ans[$box[n][i][0]][$box[n][i][1]] > 0
          numBlock.push($ans[$box[n][i][0]][$box[n][i][1]])
        end
      end
      $cand[k][j] = []
      for n in 1..9
        $cand[k][j].push(n)
      end
      $cand[k][j] = $cand[k][j] - (numYoko | numTate | numBlock)
    end
  end
end

$ban = []
```

```
$ans = []
def f_sample1
  $ban = [
    [0,1,0,0,0,0,9,0,3],
    [0,9,0,0,0,7,0,0,0],
    [0,0,0,0,0,0,0,7,1],
    [6,0,9,0,0,0,0,0,0],
    [0,0,7,6,0,0,0,0,0],
    [2,0,0,8,0,5,0,0,0],
    [0,0,4,0,0,8,0,0,0],
    [0,3,0,0,2,0,0,0,0],
    [0,0,8,5,3,0,0,9,4]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
def f_sample2
  $ban = [
    [0,0,0,0,0,6,2,0,1],
    [0,0,0,0,4,0,3,7,0],
    [0,0,3,0,9,2,0,0,6],
    [4,0,5,0,0,0,0,6,0],
    [0,0,0,0,7,0,0,0,0],
    [0,2,0,0,0,0,4,0,5],
    [9,0,0,4,5,0,1,0,0],
    [0,5,4,0,2,0,0,0,0],
    [7,0,1,9,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
def f_sample3
  $ban = [
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
```

```
  set_cand
  showban()
end

$action = TkVariable.new(0)

def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    ss = ''
    for n in $cand[k][i]
      ss += n.to_s
    end
    v = Dialog.getValue(ss)
    if $action.value.to_i == 0
      $ban[k][i] = v.to_i
      $ans[k][i] = v.to_i
    else
      $ans[k][i] = v.to_i
    end
    set_cand
    showban()
  end
end

$pathname = ""
$filename = ""

def save_ban
  filetype = "{All {*}}"
  $filename = Tk.getSaveFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
    f = open( $filename , "w" )
    for x in $ban
      ss = ""
      for n in x
        ss += n.to_s + " "
      end
```

```
      ss += "\n"
      f.write(ss)
    end
    for x in $ans
      ss = ""
      for n in x
        ss += n.to_s + " "
      end
      ss += "\n"
      f.write(ss)
    end
    f.close
  end
end
def open_ban
  filetype = "{All {*}}"
  $filename = Tk.getOpenFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
    f = open( $filename , "r" )
    for cnt in 0...18
      line = f.readline
      s = line.split()
      nlist = s.map{|item| item.to_i}
      if cnt < 9
        $ban[cnt] = nlist
      else
        $ans[cnt-9] = nlist
      end
    end
    f.close
    set_cand
    showban()
  end
end

menubar = TkMenu.new
Tk.root.configure(menu: menubar)
files =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Files', under: 0, menu: files)
files.add_command(label: "Save", command: proc {save_ban})
files.add_command(label: "Open", command: proc {open_ban})
```

```
samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})
samples.add_command(label: "Sample2", command: proc {f_sample2})
samples.add_command(label: "Sample3", command: proc {f_sample3})

TkRadiobutton.new(text: 'ban[][]', variable: $action, value: 0).pack
TkRadiobutton.new(text: 'ans[][]', variable: $action, value: 1).pack

$canvas.bind('Button-1', proc {|x,y| button_press(x, y)}, "%x, %y")

Tk.mainloop
```

となりました。

次は「局面を印刷できるようにしてみましょう」ですが、やり方が分かりません！

代わりに **table** 形式で局面を表現した、 **html** ファイルを作り、保存しましょう。

最初に

```
require 'date'
```

を追加し、メニューに

```
files.add_separator()
files.add_command(label: "HTML", command: proc {save_html})
```

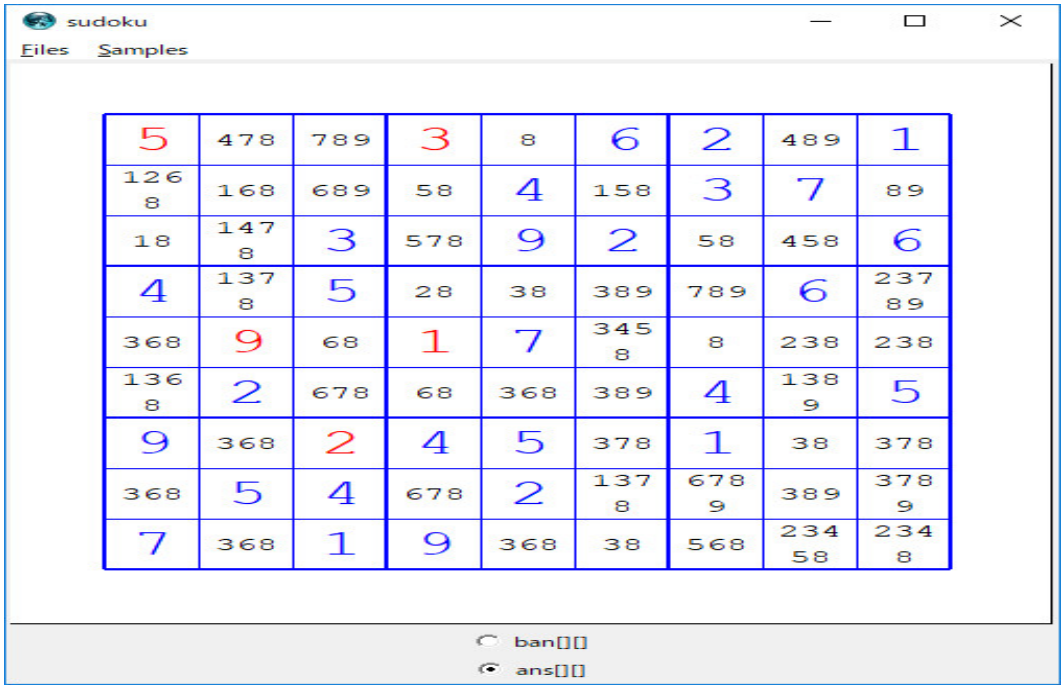
を追加し、関数 save_html を

```
def save_html
  td_array = [
    ['<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">'],
    ['<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">'],
    ['<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">'],
    ['<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">'],
    ['<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">'],
    ['<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">',
```

```
      '<td class="c6">', '<td class="c7">', '<td class="c8">'],
    ['<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">'],
    ['<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">'],
    ['<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">']
  ]
  filetype = "{All {*}}"
  $filename = Tk.getSaveFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
    f = open( $filename , "w" )
    ss = <<"EOS"
<html>
<head>
<style>
table {empty-cells:show; border-collapse:collapse;}
td {
  width:50px; height:50px;
  border-style:solid; border-width:1px; border-color:#000000;
  text-align:center; vertical-align:middle;
  font-family:Arial, Verdana, Sans-serif; font-size:2em;
}
.c0 {border-top-width:3px; border-left-width:3px;}
.c1 {border-top-width:3px;}
.c2 {border-top-width:3px; border-right-width:3px;}
.c3 {border-left-width:3px;}
.c4 {}
.c5 {border-right-width:3px;}
.c6 {border-bottom-width:3px; border-left-width:3px;}
.c7 {border-bottom-width:3px;}
.c8 {border-bottom-width:3px; border-right-width:3px;}
</style>
</head>
<body>
<table align="center">
<caption><font size="8"><p> #{Date.today} <p> </caption>
EOS
    f.write(ss)
```

```
    for k in 0..8
      f.write("<tr>")
      for j in 0..8
        if $ban[k][j] > 0
          ss = td_array[k][j] + $ban[k][j].to_s + "</td>"
          f.write(ss)
        elsif $ans[k][j] > 0
          ss = td_array[k][j].sub(/>/,' style="color:red">') +
          $ans[k][j].to_s + "</td>"
          f.write(ss)
        else
          ss = td_array[k][j] + " </td>"
          f.write(ss)
        end
        if j % 3 == 0
          f.write("\n")
        end
      end
      f.write("</tr>\n")
    end
    ss = <<"EOS"
</table>
</body>
</html>
EOS
    f.write(ss)
    f.close
  end
end
```

と定義する。実行し、



の局面を保存すると

```
<html>
<head>
<style>
table {empty-cells:show; border-collapse:collapse;}
td {
  width:50px; height:50px;
  border-style:solid; border-width:1px; border-color:#000000;
  text-align:center; vertical-align:middle;
  font-family:Arial, Verdana, Sans-serif; font-size:2em;
}
.c0 {border-top-width:3px; border-left-width:3px;}
.c1 {border-top-width:3px;}
.c2 {border-top-width:3px; border-right-width:3px;}
.c3 {border-left-width:3px;}
.c4 {}
.c5 {border-right-width:3px;}
.c6 {border-bottom-width:3px; border-left-width:3px;}
.c7 {border-bottom-width:3px;}
.c8 {border-bottom-width:3px; border-right-width:3px;}
</style>
</head>
<body
<table align="center">
```

```
<caption><font size="8"><p> 2018-06-12 <p> </caption>
<tr><td class="c0" style="color:red">5</td>
<td class="c1"> </td><td class="c2" style="color:red">9</td><td class="c0"> </td>
<td class="c1"> </td><td class="c2">6</td><td class="c0">2</td>
<td class="c1"> </td><td class="c2">1</td></tr>
<tr><td class="c3"> </td>
<td class="c4"> </td><td class="c5"> </td><td class="c3"> </td>
<td class="c4">4</td><td class="c5"> </td><td class="c3">3</td>
<td class="c4">7</td><td class="c5"> </td></tr>
<tr><td class="c6"> </td>
<td class="c7"> </td><td class="c8">3</td><td class="c6"> </td>
<td class="c7">9</td><td class="c8">2</td><td class="c6"> </td>
<td class="c7"> </td><td class="c8">6</td></tr>
<tr><td class="c0">4</td>
<td class="c1"> </td><td class="c2">5</td><td class="c0"> </td>
<td class="c1"> </td><td class="c2"> </td><td class="c0"> </td>
<td class="c1">6</td><td class="c2"> </td></tr>
<tr><td class="c3"> </td>
<td class="c4"> </td><td class="c5"> </td><td class="c3"> </td>
<td class="c4">7</td><td class="c5"> </td><td class="c3"> </td>
<td class="c4"> </td><td class="c5"> </td></tr>
<tr><td class="c6"> </td>
<td class="c7">2</td><td class="c8"> </td><td class="c6"> </td>
<td class="c7"> </td><td class="c8"> </td><td class="c6">4</td>
<td class="c7"> </td><td class="c8">5</td></tr>
<tr><td class="c0">9</td>
<td class="c1"> </td><td class="c2"> </td><td class="c0">4</td>
<td class="c1">5</td><td class="c2"> </td><td class="c0">1</td>
<td class="c1"> </td><td class="c2"> </td></tr>
<tr><td class="c3"> </td>
<td class="c4">5</td><td class="c5">4</td><td class="c3"> </td>
<td class="c4">2</td><td class="c5"> </td><td class="c3"> </td>
<td class="c4"> </td><td class="c5"> </td></tr>
<tr><td class="c6">7</td>
<td class="c7"> </td><td class="c8">1</td><td class="c6">9</td>
<td class="c7"> </td><td class="c8"> </td><td class="c6"> </td>
<td class="c7"> </td><td class="c8"> </td></tr>
</table>
</body>
</html>
```

のようなファイルを保存します。これを開くと

2018-06-12

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

のようなページが開かれます。局面と html ファイルを作った日時が表示されています。米朝首脳会談が行われた日にここまでプログラミング出来ました。これを印刷すれば、局面の印刷ができます。これで良いことにします。ここまでで、プログラムは 435 行です。一般に、50 行のプログラムは誰でも作れ、更に、自分の専門分野のプログラムだあれば、500 行のプログラムは誰でも作れると言われています。

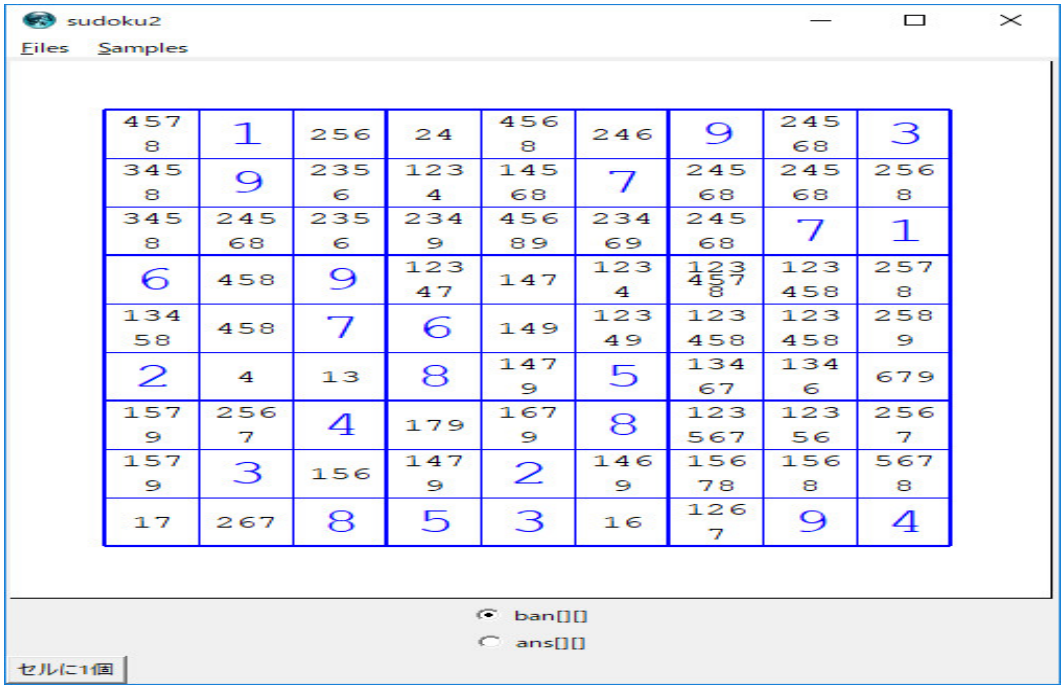
次に、コンピュータに問題を解かせてみましょう。候補が 1 個しかないセルは確定します。

```
TkButton.new(text:"セルに 1 個", command: proc{uniqucell}).pack(side:'left')
```

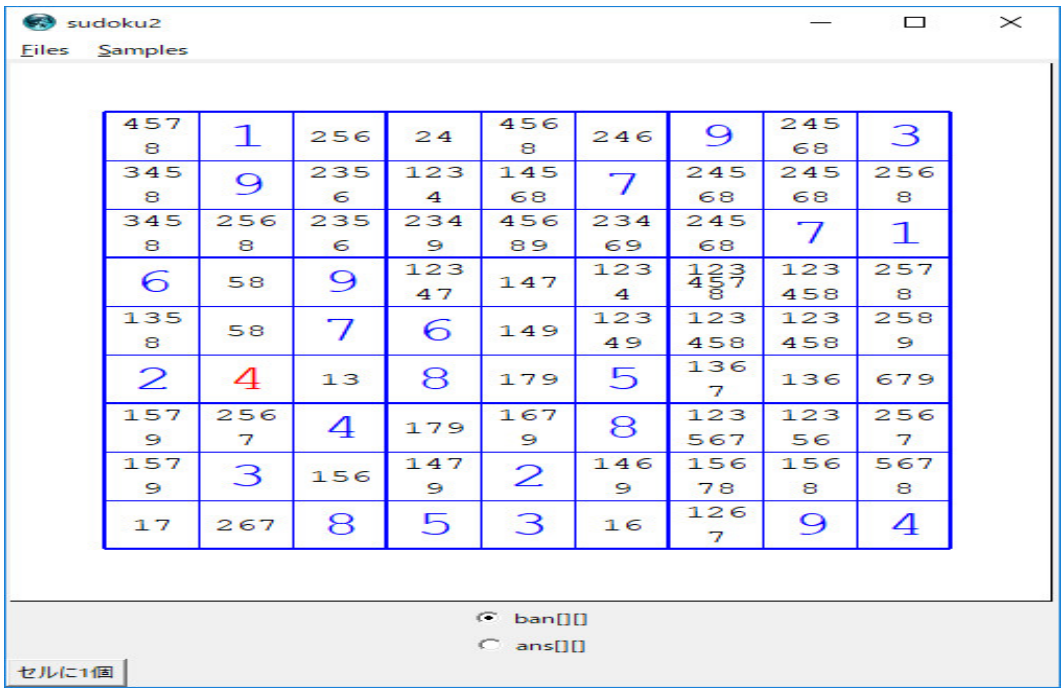
を追加し、

```
def uniqucell
  for j in 0...9
    for i in 0...9
      if $cand[j][i].length == 1 then
        $ans[j][i] = $cand[j][i].shift
      end
    end
  end
  set_cand
  showban()
end
```

と定義します。実行し、メニューの Sample1 をクリックすると



です。左端中央のブロックに候補が 4 だけのセルがあります。「セルに 1 個」のボタンをクリックすると



となります。

　各ブロックや各列や各行で候補になっている場所が一か所しかない数字は確定します。これをコンピュータに見つけさせ、確定するようにします。

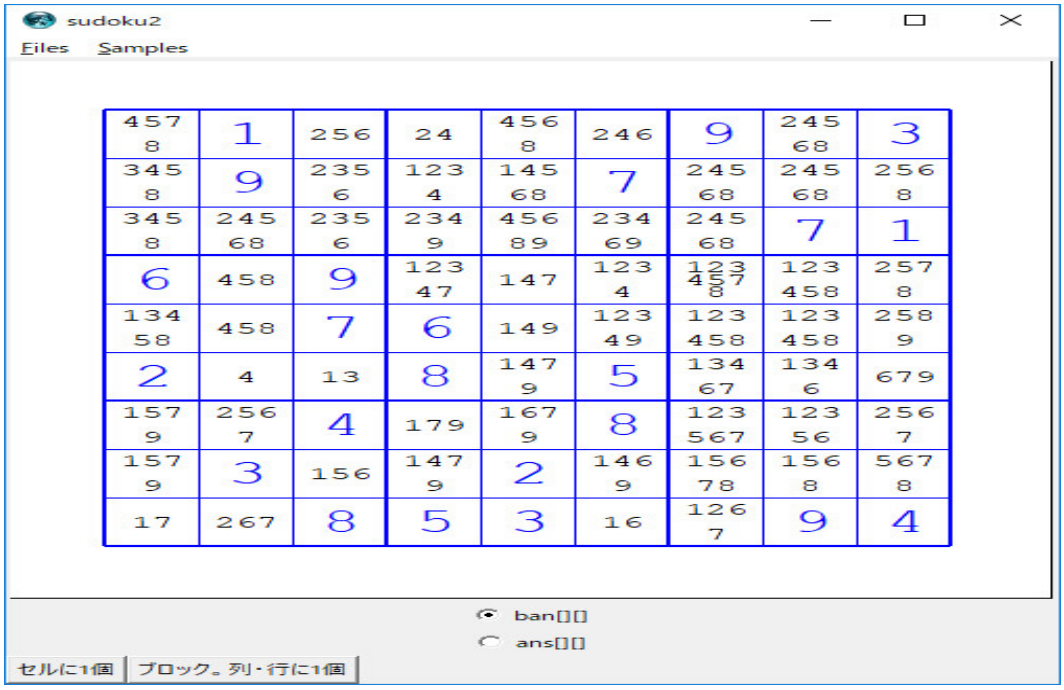
```
TkButton.new(text:"ブロック。列・行に 1 個",
  command: proc{uniquenum}).pack(side:'left')
```

を追加し、

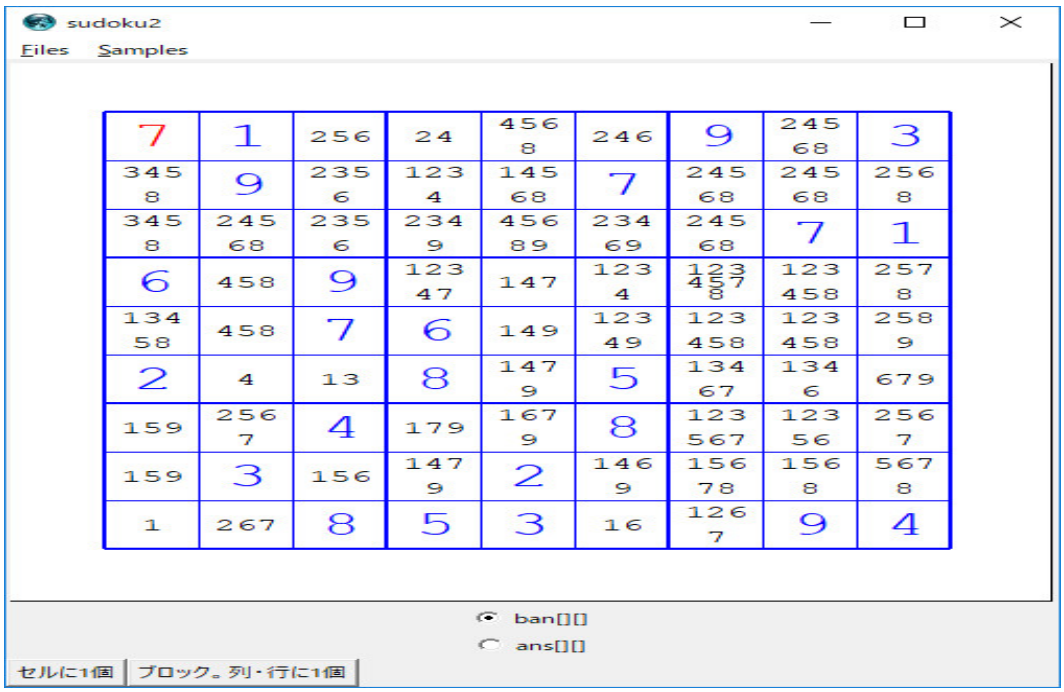
```
def uniquenum
  for j in 0...9
    for i in 0...9
      if $ans[j][i] > 0 then
        next
      end
      for n in $cand[j][i]
        flag = false
        for k in 0...9
          if k == i then
            next
          end
          if $ans[j][k] > 0 then
            next
          end
          if $cand[j][k].include?(n) then
            flag = true
            break
          end
        end
        if !flag then
          $ans[j][i] = n
          set_cand
          showban()
          return
        end
        flag = false
        for k in 0...9
          if k == j then
            next
          end
          if $ans[k][i] > 0 then
            next
          end
          if $cand[k][i].include?(n) then
            flag = true
            break
          end
        end
        if !flag then
          $ans[j][i] = n
```

```
          set_cand
          showban()
          return
        end
        flag = false
        m = j/3*3+i/3
        for k in 0...9
          if k == j%3*3+i%3 then
            next
          end
          if $ans[$box[m][k][0]][$box[m][k][1]] > 0 then
            next
          end
          if $cand[$box[m][k][0]]][$box[m][k][1]]].include?(n) then
            flag = true
            break
          end
        end
        if !flag then
          $ans[j][i] = n
          set_cand
          showban()
          return
        end

      end
    end
  end
end
```

と定義します。実行し、メニューの Sample1 をクリックすると



です。左上隅のセルの候補 7 はそのブロックで唯一の候補です。従って、このセルは 7 に確定します。更に、このセルの 7 はその行の唯一の候補でもあります。列を見ると 7 は 4 か所にあります。ブロックか列か行かでユニックであれば確定します。「ブロック・列・行に 1 個」のボタンをクリックすると



となります。「ブロック・列・行に 1 個」のボタンを何回か押すと解けるやさしい問題もあります。

最後にバックトラッキング（虱潰し探索）によって、どんな問題でも解けるようにしてみましょう。

```
TkButton.new(text:"虱潰し探索",
```

```
  command: proc{backtrack}).pack(side: 'left')
```

を追加し、関数 completeP() を

```
def completeP
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        return false
      end
    end
  end
  for j in 0...9
    for n in 1..9
      flag = false
      for i in 0...9
        if $ans[j][i] == n then
          flag = true
          break
        end
      end
      if !flag then
        return false
      end
    end
  end
  for i in 0...9
    for n in 1..9
      flag = false
      for j in 0...9
        if $ans[j][i] == n then
          flag = true
          break
        end
      end
      if !flag then
        return false
      end
    end
  end
  for k in 0...9
    for n in 1..9
      flag = false
      for i in 0...9
```

```
        if $ans[$box[k][i][0]][$box[k][i][1]] == n then
          flag = true
          break
        end
      end
      if !flag then
        return false
      end
    end
  end
  return true
end
```

と定義し、関数 losingP() を

```
ef losingP
  set_cand
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        if $cand[j][i].length == 0 then
          return true
        end
      end
    end
  end
  for j in 0...9
    p = [0,0,0,0,0,0,0,0,0,0]
    for i in 0...9
      p[$ans[j][i]] = p[$ans[j][i]] + 1
    end
    for n in 1..9
      if p[n] > 1 then
        return true
      end
    end
  end
  for i in 0...9
    p = [0,0,0,0,0,0,0,0,0,0]
    for j in 0...9
      p[$ans[j][i]] = p[$ans[j][i]] + 1
    end
    for n in 1..9
      if p[n] > 1 then
```

```
        return true
      end
    end
  end
  for k in 0...9
    p = [0,0,0,0,0,0,0,0,0,0]
    for i in 0...9
      p[$ans[$box[k][i][0]][$box[k][i][1]]] =
        p[$ans[$box[k][i][0]][$box[k][i][1]]] + 1
    end
    for n in 1..9
      if p[n] > 1 then
        return true
      end
    end
  end
  return false
end
```

と定義し、関数 solver() を

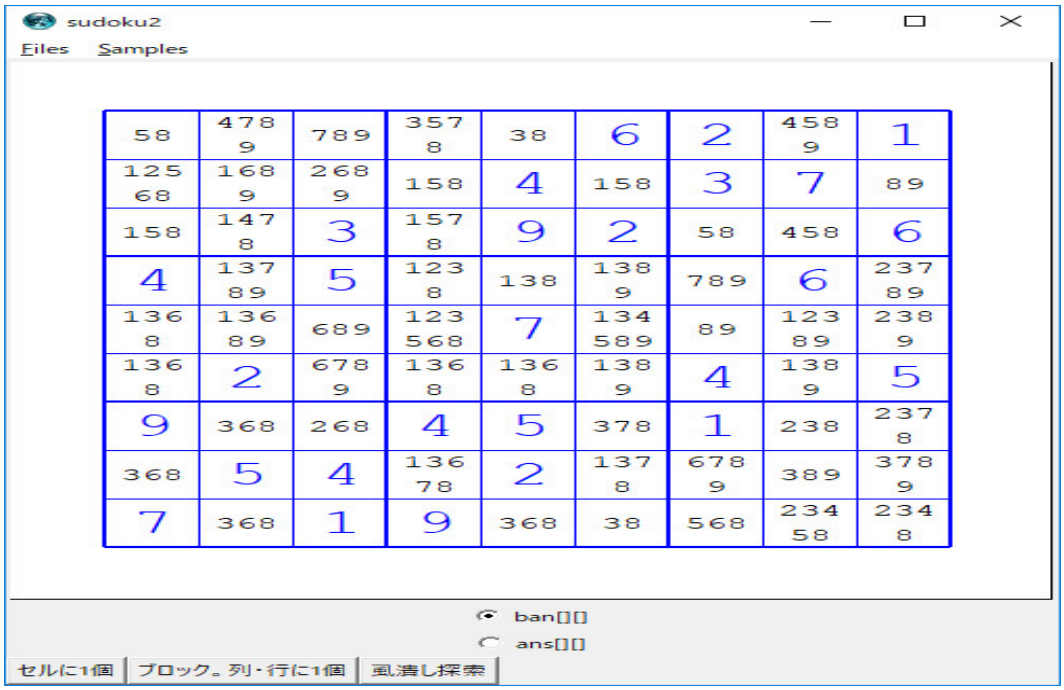
```
def solver
  if losingP then
    return false
  end
  if completeP then
    return true
  end
  flag = false
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        jj = j
        ii = i
        flag = true
        break
      end
      if flag then
       break
      end
    end
  end
  s = $cand[jj][ii].dup
  s.each do |n|
```

```
    $ans[jj][ii] = n
    if solver then
      return true
    end
    $ans[jj][ii] = 0
  end
  return false
end
```

と定義し付け加え、

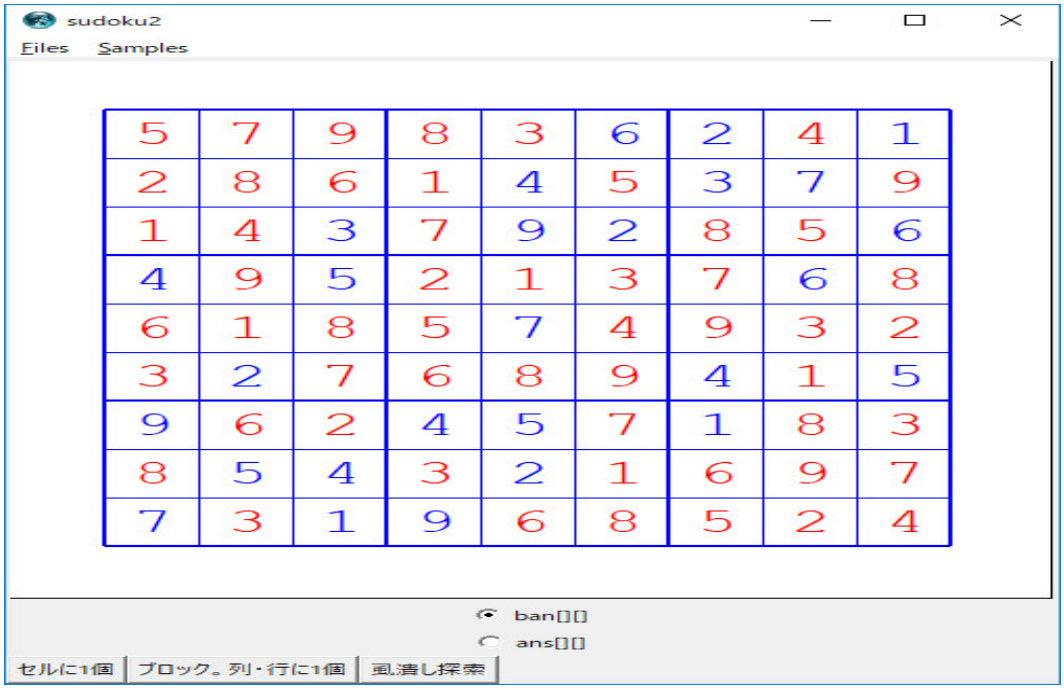
```
def backtrack
  solver
  showban()
end
```

と定義します。実行し、メニューの Sample2 をクリックすると



58	4789	789	3578	38	6	2	4589	1
12568	1689	2689	158	4	158	3	7	89
158	1478	3	1578	9	2	58	458	6
4	13789	5	1238	138	1389	789	6	23789
1368	13689	689	123568	7	134589	89	12389	2389
1368	2	6789	1368	1368	1389	4	1389	5
9	368	268	4	5	378	1	238	2378
368	5	4	13678	2	1378	6789	389	3789
7	368	1	9	368	38	568	23458	2348

です。「虱潰し探索」のボタンをクリックすると途中「応答なし」の表示が出たりしますが、そのまま待っていると



となります。関数 uniqucell() と uniquenum() を作り直して、もっと高速にしてみましょう。

```
def uniqucell
  flag = false
  for j in 0...9
    for i in 0...9
      if $ans[j][i] > 0 then
        next
      end
      if $cand[j][i].length == 1 then
        $ans[j][i] = $cand[j][i].shift
        flag = true
      end
    end
  end
  set_cand
  showban()
  return flag
end
```

と

```
def uniquenum
  for j in 0...9
    for i in 0...9
      if $ans[j][i] > 0 then
```

```
  next
end
for n in $cand[j][i]
  flag = false
  for k in 0...9
    if k == i then
      next
    end
    if $ans[j][k] > 0 then
      next
    end
    if $cand[j][k].include?(n) then
      flag = true
      break
    end
  end
  if !flag then
    $ans[j][i] = n
    set_cand
    showban()
    return true
  end
  flag = false
  for k in 0...9
    if k == j then
      next
    end
    if $ans[k][i] > 0 then
      next
    end
    if $cand[k][i].include?(n) then
      flag = true
      break
    end
  end
  if !flag then
    $ans[j][i] = n
    set_cand
    showban()
    return true
  end
  flag = false
  m = j/3*3+i/3
```

```
        for k in 0...9
          if k == j%3*3+i%3 then
            next
          end
          if $ans[$box[m][k][0]][$box[m][k][1]] > 0 then
            next
          end
          if $cand[$box[m][k][0]][$box[m][k][1]].include?(n) then
            flag = true
            break
          end
        end
        if !flag then
          $ans[j][i] = n
          set_cand
          showban()
          return true
        end
      end
    end
  end
  return false
end
```

と修正します。このように修正しても、「セルに 1 個」と「ブロック。列・行に 1 個」のボタンは正常に動きます。このように修正して、関数 regularPlay() を

```
def regularPlay
  flag = false
  loop do
    loop do
      flag = uniqucell
      if !flag then
        break
      end
    end
    flag = uniquenum
    if !flag then
      break
    end
  end
end
```

と定義し、関数 solver() を

```
def solver
  regularPlay
  if losingP then
    return false
  end
  if completeP then
    return true
  end
  flag = false
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        jj = j
        ii = i
        flag = true
        break
      end
      if flag then
       break
      end
    end
  end
  if flag then
    tmpans = Marshal.load(Marshal.dump($ans))
    set_cand
    s = $cand[jj][ii].dup
    s.each do |n|
      $ans[jj][ii] = n
      if solver then
        return true
      end
      $ans = Marshal.load(Marshal.dump(tmpans))
      $ans[jj][ii] = 0
      set_cand
    end
  end
  return false
end
```

と修正します。

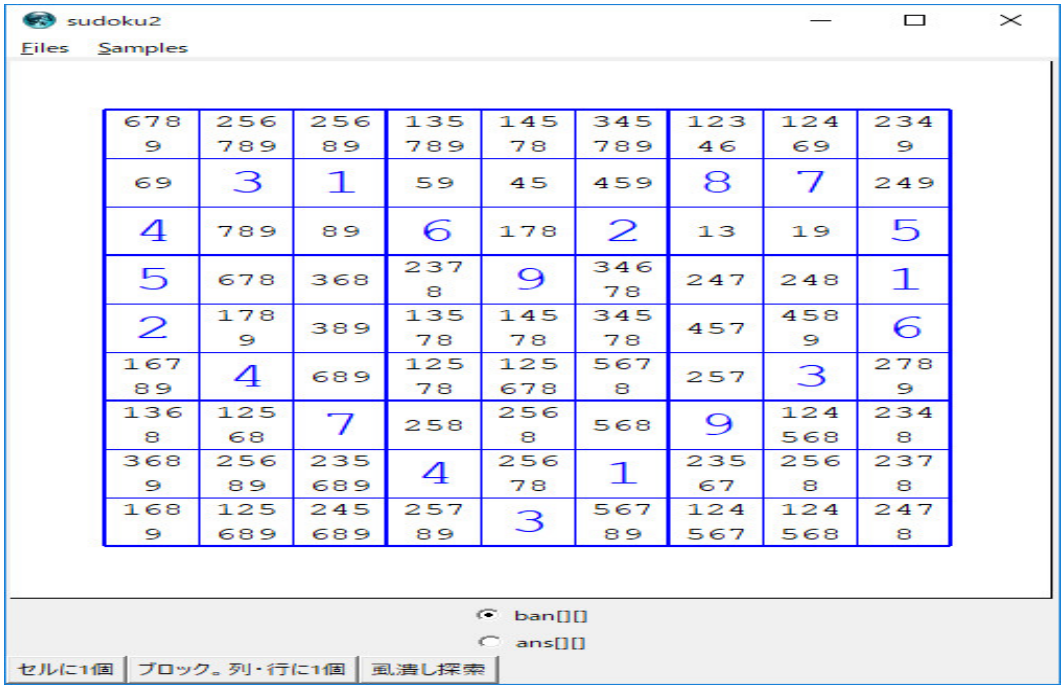
```
samples.add_command(label: "Sample4", command: proc {f_sample4})
```

を追加し、

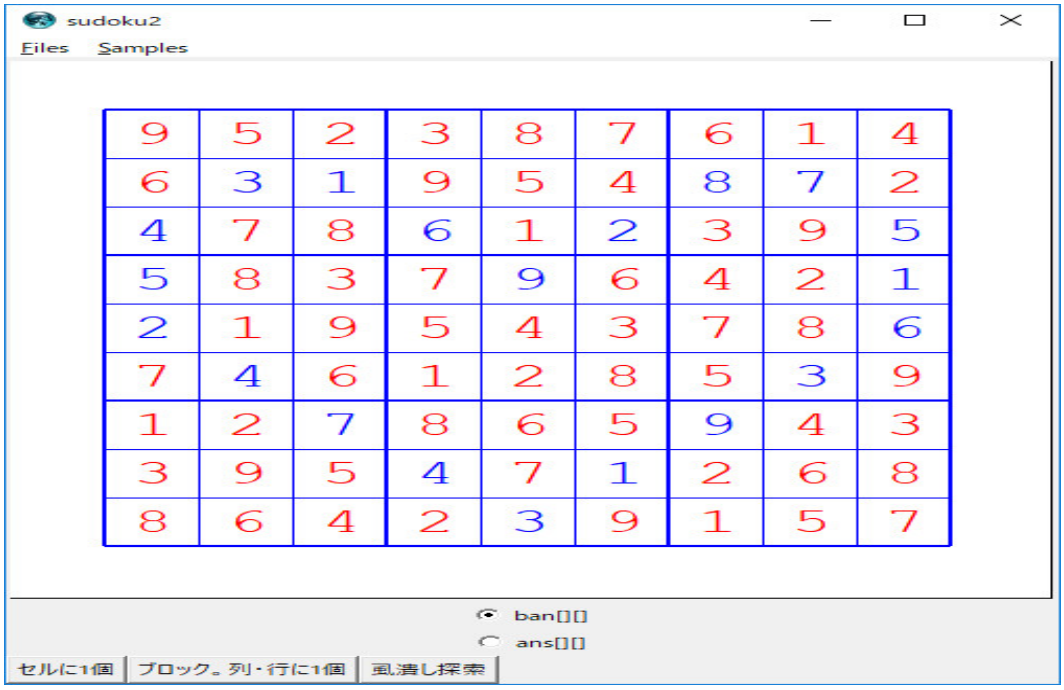
```
def f_sample4
  $ban = [
    [0,0,0,0,0,0,0,0,0],
    [0,3,1,0,0,0,8,7,0],
    [4,0,0,6,0,2,0,0,5],
    [5,0,0,0,9,0,0,0,1],
    [2,0,0,0,0,0,0,0,6],
    [0,4,0,0,0,0,0,3,0],
    [0,0,7,0,0,0,9,0,0],
    [0,0,0,4,0,1,0,0,0],
    [0,0,0,0,3,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
```

と定義します。

実行し、メニューの Sample4 をクリックすると



です。「虱潰し探索」のボタンをクリックすると



となります。だいぶ早くなりました。「数独」を人間が解いている法則をプログラミングして、regularPlay() に放り込むともっと早くなります。

最終的なプログラムの全体は

```
require 'tk'
require 'date'

module Dialog
  def Dialog.getValue(init)
    var = TkVariable.new("")
    toplevel = TkToplevel.new do
      grab
    end
    TkLabel.new(toplevel) do
      text init
      font 'helvetica 18'
      pack
    end
    TkEntry.new(toplevel) do
      width 15
      font 'helvetica 18'
      textvariable var
      focus
      pack
    end
    TkButton.new(toplevel) do
```

```
      text 'OK'
      font 'helvetica 18'
      command do
        toplevel.grab("release")
        toplevel.destroy
      end
      pack
    end
    toplevel.wait_destroy
    var.value
  end
end

BOARD_SIZE = 500
$canvas = TkCanvas.new(width: BOARD_SIZE, height: BOARD_SIZE).pack

def showban
  TkcRectangle.new($canvas, 0, 0, BOARD_SIZE, BOARD_SIZE) do
    fill 'white'
  end
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w, h+i*s, w+9*s, h+i*s) do
        fill 'blue'
      end
    end
  end
  for i in 0..9
    if i % 3 == 0
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
        fill 'blue'
        width 2.0
      end
    else
      TkcLine.new($canvas, w+i*s, h, w+i*s, h+9*s) do
```

```
      fill 'blue'
    end
  end
end
f1, f2 = "courier 24", "courier 12"
c1, c2, c3 = "blue", "red", "black"
for k in 0...9
  for i in 0...9
    if $ban[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ban[k][i]).to_s
        font f1
        fill c1
      end
    elsif $ans[k][i] > 0
      TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
        text ($ans[k][i]).to_s
        font f1
        fill c2
      end
    else
      ss = ''
      ss1 = ''
      ss2 = ''
      $cand[k][i].each_with_index do |elem, ind|
        if ind == 3
          ss1 = ss
          ss = ''
        elsif ind == 6
          ss2 = ss
          ss = ''
        end
        ss += elem.to_s
      end
      if ss1 != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.25)*s) do
          text ss1
          font f2
          fill c3
        end
      end
      if ss2 != ''
        TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
```

```
            text ss2
            font f2
            fill c3
          end
        end
        if ss1 != '' and ss2 != '' and ss != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
            text ss
            font f2
            fill c3
          end
        elsif ss1 != '' and ss != ''
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.75)*s) do
            text ss
            font f2
            fill c3
          end
        else
          TkcText.new($canvas, w+(i+0.5)*s, h+(k+0.5)*s) do
            text ss
            font f2
            fill c3
          end
        end
      end
    end
  end
end

$box = [
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]],
  [[0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0], [0,0]]
]
$cand = [
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
```

```
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []],
  [[], [], [], [], [], [], [], [], []]
]

def set_cand
  for k in 0...9
    for j in 0...9
      $box[k/3*3+j/3][k%3*3+j%3][0] = k
      $box[k/3*3+j/3][k%3*3+j%3][1] = j
    end
  end
  for k in 0...9
    for j in 0...9
      numYoko = []
      for i in 0...9
        if $ans[k][i] > 0
          numYoko.push($ans[k][i])
        end
      end
      numTate = []
      for i in 0...9
        if $ans[i][j] > 0
          numTate.push($ans[i][j])
        end
      end
      numBlock = []
      n = k/3*3+j/3
      for i in 0...9
        if $ans[$box[n][i][0]][$box[n][i][1]] > 0
          numBlock.push($ans[$box[n][i][0]][$box[n][i][1]])
        end
      end
      $cand[k][j] = []
      for n in 1..9
        $cand[k][j].push(n)
      end
      $cand[k][j] = $cand[k][j] - (numYoko | numTate | numBlock)
    end
```

```
  end
end

$ban = []
$ans = []
def f_sample1
  $ban = [
    [0,1,0,0,0,0,9,0,3],
    [0,9,0,0,0,7,0,0,0],
    [0,0,0,0,0,0,0,7,1],
    [6,0,9,0,0,0,0,0,0],
    [0,0,7,6,0,0,0,0,0],
    [2,0,0,8,0,5,0,0,0],
    [0,0,4,0,0,8,0,0,0],
    [0,3,0,0,2,0,0,0,0],
    [0,0,8,5,3,0,0,9,4]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
def f_sample2
  $ban = [
    [0,0,0,0,0,6,2,0,1],
    [0,0,0,0,4,0,3,7,0],
    [0,0,3,0,9,2,0,0,6],
    [4,0,5,0,0,0,0,6,0],
    [0,0,0,0,7,0,0,0,0],
    [0,2,0,0,0,0,4,0,5],
    [9,0,0,4,5,0,1,0,0],
    [0,5,4,0,2,0,0,0,0],
    [7,0,1,9,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
def f_sample3
  $ban = [
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
```

```
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0],
    [0,0,0,0,0,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end
def f_sample4
  $ban = [
    [0,0,0,0,0,0,0,0,0],
    [0,3,1,0,0,0,8,7,0],
    [4,0,0,6,0,2,0,0,5],
    [5,0,0,0,9,0,0,0,1],
    [2,0,0,0,0,0,0,0,6],
    [0,4,0,0,0,0,0,3,0],
    [0,0,7,0,0,0,9,0,0],
    [0,0,0,4,0,1,0,0,0],
    [0,0,0,0,3,0,0,0,0]]
  $ans = Marshal.load(Marshal.dump($ban))
  set_cand
  showban()
end

$action = TkVariable.new(0)

def button_press(x, y)
  k = 9
  s = BOARD_SIZE/(k+2)
  w = h = (BOARD_SIZE-s*9)/2
  i = (x.to_i - w) / s
  k = (y.to_i - h) / s
  if i >= 0 and i < 9 and k >= 0 and k < 9
    ss = ''
    for n in $cand[k][i]
      ss += n.to_s
    end
    v = Dialog.getValue(ss)
    if $action.value.to_i == 0
      $ban[k][i] = v.to_i
      $ans[k][i] = v.to_i
    else
      $ans[k][i] = v.to_i
    end
```

```
    set_cand
    showban()
  end
end

$pathname = ""
$filename = ""

def save_ban
  filetype = "{All {*}}"
  $filename = Tk.getSaveFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
    f = open( $filename , "w" )
    for x in $ban
      ss = ""
      for n in x
        ss += n.to_s + " "
      end
      ss += "\n"
      f.write(ss)
    end
    for x in $ans
      ss = ""
      for n in x
        ss += n.to_s + " "
      end
      ss += "\n"
      f.write(ss)
    end
    f.close
  end
end
def open_ban
  filetype = "{All {*}}"
  $filename = Tk.getOpenFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
    f = open( $filename , "r" )
    for cnt in 0...18
      line = f.readline
      s = line.split()
      nlist = s.map{|item| item.to_i}
```

```
      if cnt < 9
        $ban[cnt] = nlist
      else
        $ans[cnt-9] = nlist
      end
    end
    f.close
    set_cand
    showban()
  end
end


def save_html
  td_array = [
    ['<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">'],
    ['<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">'],
    ['<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">'],
    ['<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">'],
    ['<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">'],
    ['<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">'],
    ['<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">',
      '<td class="c0">', '<td class="c1">', '<td class="c2">'],
    ['<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">',
      '<td class="c3">', '<td class="c4">', '<td class="c5">'],
    ['<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">',
      '<td class="c6">', '<td class="c7">', '<td class="c8">']
  ]
```

```
  filetype = "{All {*}}"
  $filename = Tk.getSaveFile(filetypes: filetype, initialdir: $pathname)
  if $filename != ""
    $pathname = File::dirname($filename)
    f = open( $filename , "w" )
    ss = <<"EOS"
<html>
<head>
<style>
table {empty-cells:show; border-collapse:collapse;}
td {
  width:50px; height:50px;
  border-style:solid; border-width:1px; border-color:#000000;
  text-align:center; vertical-align:middle;
  font-family:Arial, Verdana, Sans-serif; font-size:2em;
}
.c0 {border-top-width:3px; border-left-width:3px;}
.c1 {border-top-width:3px;}
.c2 {border-top-width:3px; border-right-width:3px;}
.c3 {border-left-width:3px;}
.c4 {}
.c5 {border-right-width:3px;}
.c6 {border-bottom-width:3px; border-left-width:3px;}
.c7 {border-bottom-width:3px;}
.c8 {border-bottom-width:3px; border-right-width:3px;}
</style>
</head>
<body>
<table align="center">
<caption><font size="8"><p> #{Date.today} <p> </caption>
EOS
    f.write(ss)
    for k in 0..8
      f.write("<tr>")
      for j in 0..8
        if $ban[k][j] > 0
          ss = td_array[k][j] + $ban[k][j].to_s + "</td>"
          f.write(ss)
        elsif $ans[k][j] > 0
          ss = td_array[k][j].sub(/>/,' style="color:red">') +
          $ans[k][j].to_s + "</td>"
          f.write(ss)
        else
```

```
          ss = td_array[k][j] + " </td>"
          f.write(ss)
        end
        if j % 3 == 0
          f.write("\n")
        end
      end
      f.write("</tr>\n")
    end
    ss = <<"EOS"
</table>
</body>
</html>
EOS
    f.write(ss)
    f.close
  end
end

def uniqucell
  flag = false
  for j in 0...9
    for i in 0...9
      if $ans[j][i] > 0 then
        next
      end
      if $cand[j][i].length == 1 then
        $ans[j][i] = $cand[j][i].shift
        flag = true
      end
    end
  end
  set_cand
  showban()
  return flag
end

def uniquenum
  for j in 0...9
    for i in 0...9
      if $ans[j][i] > 0 then
        next
      end
```

```
for n in $cand[j][i]
  flag = false
  for k in 0...9
    if k == i then
      next
    end
    if $ans[j][k] > 0 then
      next
    end
    if $cand[j][k].include?(n) then
      flag = true
      break
    end
  end
  if !flag then
    $ans[j][i] = n
    set_cand
    showban()
    return true
  end
  flag = false
  for k in 0...9
    if k == j then
      next
    end
    if $ans[k][i] > 0 then
      next
    end
    if $cand[k][i].include?(n) then
      flag = true
      break
    end
  end
  if !flag then
    $ans[j][i] = n
    set_cand
    showban()
    return true
  end
  flag = false
  m = j/3*3+i/3
  for k in 0...9
    if k == j%3*3+i%3 then
```

```
            next
          end
          if $ans[$box[m][k][0]][$box[m][k][1]] > 0 then
            next
          end
          if $cand[$box[m][k][0]][$box[m][k][1]].include?(n) then
            flag = true
            break
          end
        end
        if !flag then
          $ans[j][i] = n
          set_cand
          showban()
          return true
        end

      end
    end
  end
  return false
end

def completeP
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        return false
      end
    end
  end
  for j in 0...9
    for n in 1..9
      flag = false
      for i in 0...9
        if $ans[j][i] == n then
          flag = true
          break
        end
      end
      if !flag then
        return false
      end
```

```
    end
  end
  for i in 0...9
    for n in 1..9
      flag = false
      for j in 0...9
        if $ans[j][i] == n then
          flag = true
          break
        end
      end
      if !flag then
        return false
      end
    end
  end
  for k in 0...9
    for n in 1..9
      flag = false
      for i in 0...9
        if $ans[$box[k][i][0]][$box[k][i][1]] == n then
          flag = true
          break
        end
      end
      if !flag then
        return false
      end
    end
  end
  return true
end

def losingP
  set_cand
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        if $cand[j][i].length == 0 then
          return true
        end
      end
    end
```

```
  end
  for j in 0...9
    p = [0,0,0,0,0,0,0,0,0,0]
    for i in 0...9
      p[$ans[j][i]] = p[$ans[j][i]] + 1
    end
    for n in 1..9
      if p[n] > 1 then
        return true
      end
    end
  end
  for i in 0...9
    p = [0,0,0,0,0,0,0,0,0,0]
    for j in 0...9
      p[$ans[j][i]] = p[$ans[j][i]] + 1
    end
    for n in 1..9
      if p[n] > 1 then
        return true
      end
    end
  end
  for k in 0...9
    p = [0,0,0,0,0,0,0,0,0,0]
    for i in 0...9
      p[$ans[$box[k][i][0]][$box[k][i][1]]] =
        p[$ans[$box[k][i][0]][$box[k][i][1]]] + 1
    end
    for n in 1..9
      if p[n] > 1 then
        return true
      end
    end
  end
  return false
end

def regularPlay
  flag = false
  loop do
    loop do
      flag = uniqucell
```

```
      if !flag then
        break
      end
    end
    flag = uniquenum
    if !flag then
      break
    end
  end
end

def solver
  regularPlay
  if losingP then
    return false
  end
  if completeP then
    return true
  end
  flag = false
  for j in 0...9
    for i in 0...9
      if $ans[j][i] == 0 then
        jj = j
        ii = i
        flag = true
        break
      end
      if flag then
       break
      end
    end
  end
  if flag then
    tmpans = Marshal.load(Marshal.dump($ans))
    set_cand
    s = $cand[jj][ii].dup
    s.each do |n|
      $ans[jj][ii] = n
      if solver then
        return true
      end
      $ans = Marshal.load(Marshal.dump(tmpans))
```

```
      $ans[jj][ii] = 0
      set_cand
    end
  end
  return false
end

def backtrack
  solver
  showban()
end

menubar = TkMenu.new
Tk.root.configure(menu: menubar)
files =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Files', under: 0, menu: files)
files.add_command(label: "Save", command: proc {save_ban})
files.add_command(label: "Open", command: proc {open_ban})
files.add_separator()
files.add_command(label: "HTML", command: proc {save_html})

samples =TkMenu.new(menubar, tearoff: false)
menubar.add_cascade(label: 'Samples', under: 0, menu: samples)
samples.add_command(label: "Sample1", command: proc {f_sample1})
samples.add_command(label: "Sample2", command: proc {f_sample2})
samples.add_command(label: "Sample3", command: proc {f_sample3})
samples.add_command(label: "Sample4", command: proc {f_sample4})

TkRadiobutton.new(text: 'ban[][]', variable: $action, value: 0).pack
TkRadiobutton.new(text: 'ans[][]', variable: $action, value: 1).pack
TkButton.new(text:"セルに 1 個", command: proc{uniqucell}).pack(side: 'left')
TkButton.new(text:"ブロック。列・行に 1 個",
  command: proc{uniquenum}).pack(side: 'left')
#TkButton.new(text:"ブロック。列・行に 1 個",
#  command: proc{regularPlay}).pack(side: 'left')
TkButton.new(text:"虱潰し探索",
  command: proc{backtrack}).pack(side: 'left')

$canvas.bind('Button-1', proc {|x,y| button_press(x, y)}, "%x, %y")

Tk.mainloop
```

です。

何とか VC++で作ったプログラムと同等の以上ものを作りましたが、やはりまとまった書籍の情報がある VC++ の方が楽です。Ruby/Tk の情報はインターネットが頼りで、気長に情報を探せば、それを使って、 Ruby でもこの様にプログラミングすることは可能です。後は、これを雛形に他のパズルのプログラムも作れます。興味があれば、それぞれの命令の意味はインターネットで調べて下さい。親切な人たちが必要な情報を書き込んでくれています。

数独の解法のプログラミングの本には、Java の本ですが、棚床弘樹著「鉛筆パズルゲームプログラミング　ナンバープレス、お絵かきパズル、ナンバークロスワードのアルゴリズム」ソフトバンククリエイティブ　２００７年がありました。図書館で探すか古本屋で探して読んで下さい。昔、面白い本だと思いましたが、今となっては、利用価値のない（この本のプログラムを最新の Java でコンパイルしようとしましたが私はコンパイルできませんでした）古い本かもわかりませんが？また、David Flanagan まつもと　ゆきひろ著「プログラミング言語 Ruby」 O'REILLY にも、Ruby による数独を解くためのプログラムが載っています。Ruby は初心者にとっては Python より難しいと思いますが、Python も Ruby も、プログラミング言語を勉強するのが始めてでも、1 日 10 時間 × 30 日間、合計 300 時間も勉強すれば、このようなプログラムが作れるようになります。多分。計算機に「数独」を人間が使っている法則を使って Ruby で解かせたければ、この続きを自分でプログラミングしてみるか（少しは Python 版の数独に書きましたが)、佐藤理史著「AI プログラミング入門　Ruby で数独」近代科学社（読もうとするといきなり最初のプログラムで、ARGF て何？、「Ruby の参考書にはこれを使った例が載ってなかったけど」となりますが、

```
ARGF.each do |line|
  puts line
end
```

というプログラム（ファイル名 argftest.rb）で、

```
>ruby argftest.rb argftest.rb
```

や

```
>ruby argftest.rb argftest.rb argftest.rb
```

を実行してみれば、初心者には不可思議な ARGF の説明（ARGF : ARGV で指定されたファイルの仮想的な結合体、または ARGV が空なら標準入力へのアクセスを提供する IO オブジェクト。`$<` の同義語、ARGV : コマンド行で指定された引数を格納する配列。`$*`の同義語）が何を言っていたか分かります。プログラミング言語を理解するには、このように実験してみれば、コンピュータが答えを教えてくれます。119 頁の短い本だし、Ruby の基本を学んだあとに、自分が興味ある独力でもなんとかなりそうな分野で、プロの人の作ったプログラムを読んでみることは勉強になります。Ruby の基礎を学んだばかりのプログラミングの素人が趣味で作った上のプログラムは、正しいとは思いますが、無駄が多いです。プロの作ったプログラムは無駄を極端に省いています。）を読んでみてください。この本には問題の作り方は載っていません。また、「数独」の解法と問題作成を C でしたければ、英語の本ですが、Giulio Zambon 著「Sudoku Programming with C」 Apress があります。ダウンロードしたプログラムは、VC++ では、何か所か修正しなければいけませんが、修正は簡単です。配列 `\box[][]` の使いかたと最後の html ファイルの構造はこの本を参考にさせて貰いました。問題の作り方も書いていますが、この部分は棚床弘樹さんの本同様、期待したものではなかったです。常識的な問題の作り方が説明されているだけです。

「ひとりにしてくれ」の解法解析の文書も株式会社ニコリさまの許可が下りましたのでアップしています。こちらも参照してください。