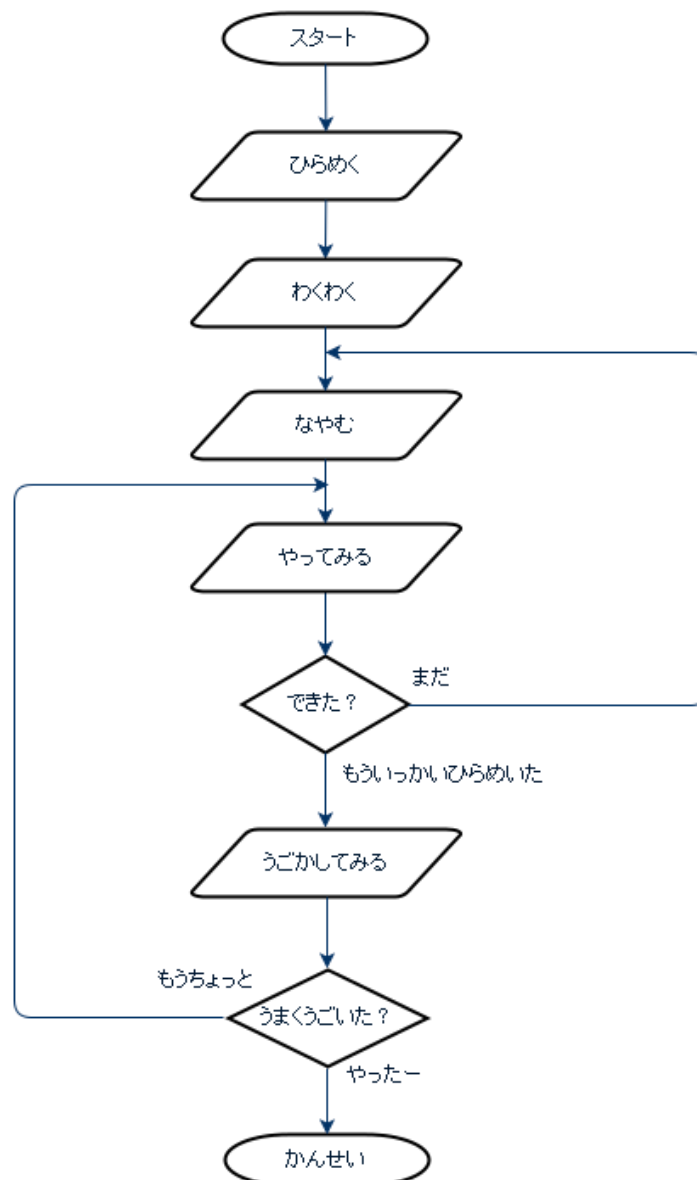


# はじめてのプログラミング

## Python バージョン



## ■はじめに

パソコンもスマホもゲーム機も、テレビも冷蔵庫も車の中にもプログラムが入っています。信号も電気や水道もすべてプログラムがコントロールしています。ロボットはひとりでに動いたり、人工知能は人間の代わりに考えて働いてくれるかもしれませんが、これもすべて人間が作ったプログラムで動いています。ドラえもんもロボットですからプログラムで動いています。

実は、たくさんのモノがプログラムでできています。そして、プログラムが書ければ、あなたはなんでも作れます。あなたが書いたプログラムで、ぜひ世界を動かしましょう。

## ■Python(パイソン)とは

本書は、パイソンを使用したはじめてのプログラミング教室の教材です。Python は、1991 年にオランダ人のグイド・ヴァンロッサム氏によって開発されたプログラミング言語です。名前の由来は、イギリスのテレビ局 BBC が製作したコメディ番組『空飛ぶモンティ・パイソン』です。Python という英単語が意味する爬虫類のニシキヘビが Python 言語のマスコットやアイコンとして使われています。

Python はコードの読みやすさや書きやすさを重視して作られました。また、文法はシンプルですが、機能は非常に強力で効率よく開発ができるので、ビジネスでも頻繁に使われる本格的なプログラミング言語です。Google や Youtube, Facebook でも使われています。

ここではプログラムの書き方を習うことより、課題を通してプログラミングに慣れることが目的です。プログラミングを楽しみましょう。そして、算数と違いプログラミングの答えは 1 つではありません。国語と同じで色んな書き方があります。そして国語と同じで他の人が読めるようにきれいに書くこと、だらだらと書かず分かりやすく書くことが大事です。

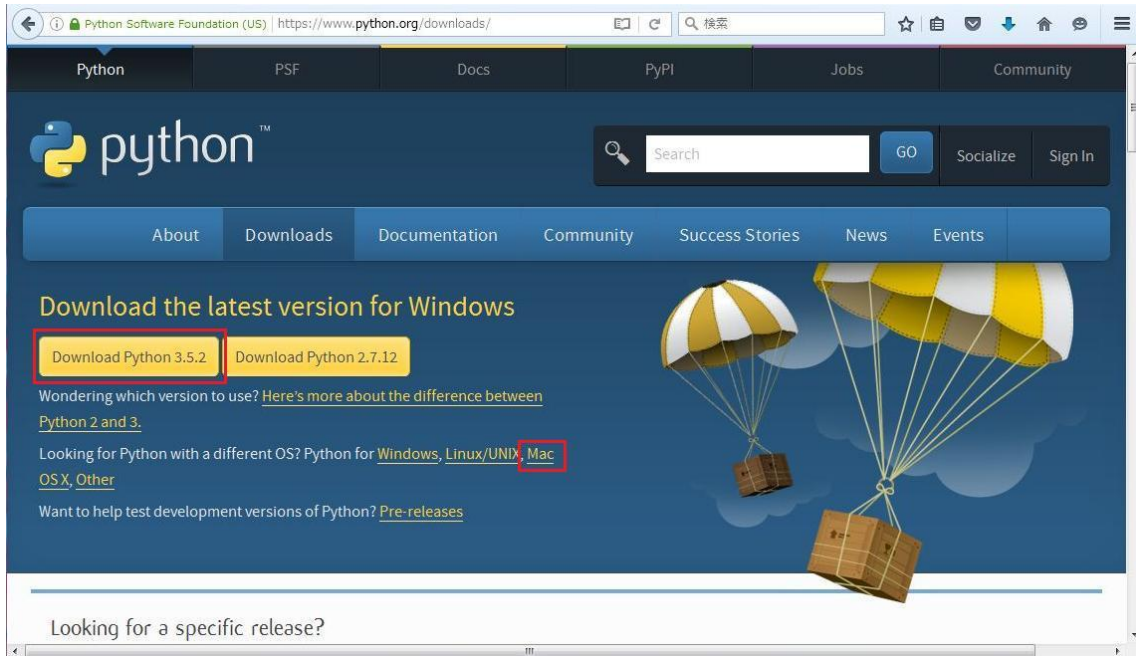
あなたのヒラメキをプログラムにしたら、うまく動くか自分の目で確かめましょう。

## ■準備 1 Python(パイソン)をインストールしよう

## ①パイソンのダウンロード

下記の URL からパイソンをダウンロードします。パソコンの OS が Windows であれば、下図の赤枠で囲んだ「Download Python 3.5.2」をクリックしよう。Mac OS の場合は赤枠で囲んだ「Mac OS X」をクリックしよう。

<https://www.python.org/downloads/>



## ②パイソンのインストール

ダウンロードしたインストーラーをダブルクリックして実行します。バージョンによって、ファイル名が変わります。



①Python(パイソン)のアイコンをダブルクリックして、Python(パイソン)を起動します。

## ■練習1 インタプリタを電卓代わりに使ってみよう

①Python(パイソン)のアイコンをダブルクリックして、Python(パイソン)を起動します。

デスクトップにPython(パイソン)のショートカットを配置した場合、下記アイコンをダブルクリックします。



図 1-1

②Python(パイソン)が起動すると、下記の画面が表示されます。Python(パイソン)をシェルウィンドウで使用する場合、下記の画面に直接プログラムを入力します。

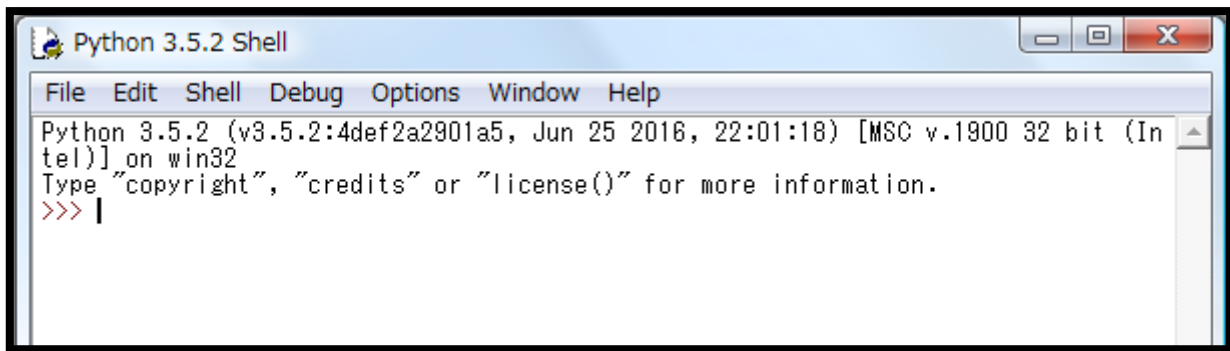


図 1-2

③下記のプログラムを入力した場合、どのような結果が返ってきますか。やってみましょう。

```
>>> 3+2
```

```
>>> 3*2
```

```
>>> 3-2
```

```
>>> 3/2
```

演算子の説明を読んで、もう一度入力してみよう。結果を書き込んで、口に出して説明してみよう。

表 1-1 算術演算子

演算子	意味	例	結果
+	加算(足し算)	17+5+4	
-	減算(引き算)	100-3	
*	乗算(かけ算)	17*5	
/	浮動小数点数の除算(わり算)	10/4	
//	整数の除算(わり算の答えを切り捨て)	10//4	
%	剰余(わり算の答え余り)	10%4	

## ■練習2 簡単な電卓作ろう パート1

①簡単な足し算をする電卓を作ります。下記のプログラムをコードウィンドウで入力しよう。

コードウィンドウは、シェルウィンドウのメニューから、「File」を開き「New File」を選択します。

# 簡単な足し算

```
from time import sleep
while True:
    print('足される数を入力してください')
    x = int(input(''))
    print('足す数を入力してください')
    y = int(input(''))
    print('x と y の合計は、', x+y, 'です')
    sleep(3)
```

②プログラムを保存します。コードウィンドウのメニューから、「File」を開き「Save As...」を選択します。

③プログラムを実行します。コードウィンドウのメニューから、「Run」を開き「Run Module」を選択します。

④実行結果は、シェルウィンドウに出力されます。プログラムにエラーがあった場合、シェルウィンドウに赤い文字で表示されます。エラーなく、実行できたかな？

⑤次は、引き算やかけ算、わり算を試してみよう。

⑥教科書の P102～105 を見ながら、次の命令の働きを調べてください。

- print
- input
- while True:
- sleep

⑦このプログラムのくり返しは、何行目から何行目まででしょうか？

## ■練習3 簡単な電卓作ろう パート2

①電卓を作ります。下記のプログラムをコードウィンドウで入力しよう。

```
# 簡単な電卓
from time import sleep
while True:
    x = int(input('x='))
    y = int(input('y='))
    op = input('足し算:+ , 引き算:-, 掛け算:*, 割り算:/ のいずれかを入力 =')
    if op == '+':
        z = x + y
    elif op == '-':
        z = x - y
    elif op == '*':
        z = x * y
    elif op == '/':
        z = x / y
    else:
        z = 'エラー'
    print('答えは、', z, 'です')
    sleep(3)
```

②プログラムを保存して実行します。

③エラーなく、実行できたかな？

④上記のプログラムでは変数を使っています。どこかな？

また、教科書 P108～109 を見ながら、変数が何かを調べてみよう。

⑤==はどういう意味かな？ 教科書の P118～119 を見ながら、比べるための記号を調べてみよう。

⑤教科書の P111 に、次の命令の働きが説明されています。なぜ int 命令を使ったか考えてみよう。

・ int

## ■練習4 クリックカウンター

①ボタンをクリックしてカウントするプログラムを作ります。シェルウィンドウに回数が出力されます。

```
# クリックカウンター
from tkinter import *
window = Tk()
count = 0
def click():
    global count
    count += 1
    print(count, '回')
click1 = Button(window, text='クリック', command=click)
click1.pack()
```

②プログラムを保存して実行します。

③エラーなく、実行できたかな？ プログラムの動きを説明してください。

④シェルウィンドウに出力される回数を10回ずつに変わります。下記の一行を追加して、判断文を作ろう。

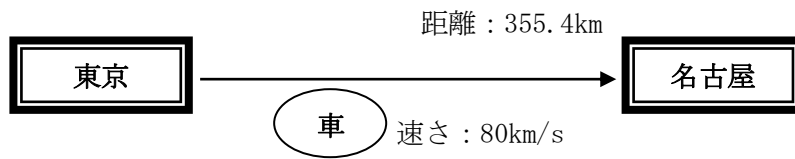
```
if count % 10 == 0:
```

③エラーなく、実行できたかな？ プログラムの動きがどんな風変わったか説明してください。

## ■練習 5 距離・速度から時間を計算しよう

東京～名古屋間は 355.4km です。高速道路を使って時速 80km のスピードで走りました。

東京から名古屋まで行くのに何時間かかるでしょう？



①距離とスピードと時間の 3 つの変数を作ろう。(distance, speed, time)

②距離とスピードから時間を求める計算式は何だろう？

③教科書の P116～117 を見ながら、計算結果を表示させよう。

④エラーなく、実行できたかな？

⑤変数を使ったプログラムの流れを説明して下さい。



## □練習 5 解答例

```
# 東京一名古屋間まで何時間か？
```

```
distance = 355.4
```

```
speed = 80
```

```
time = distance / speed
```

```
print(time, "h")
```

次は下記の print 命令に書き換えて下さい。

```
# 東京一名古屋間まで何時間か？
```

```
distance = 355.4
```

```
speed = 80
```

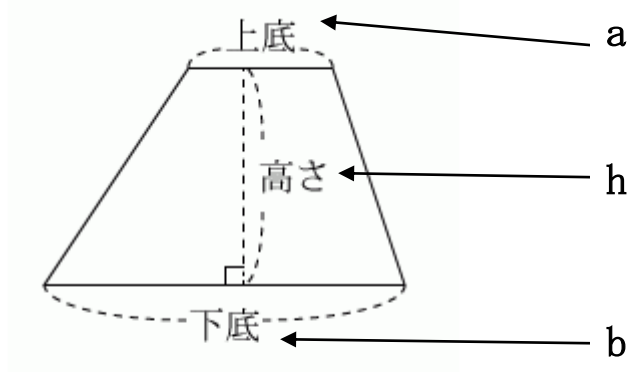
```
hour = distance / speed
```

```
min = (hour * 60) % 60
```

```
print(round(hour), "h", round(min), "min")
```

## ■練習 6 台形の面積を求めよう

台形の面積はどうやって求めるか知っていますか？



$$( \text{上底} + \text{下底} ) \times \text{高さ} \div 2$$

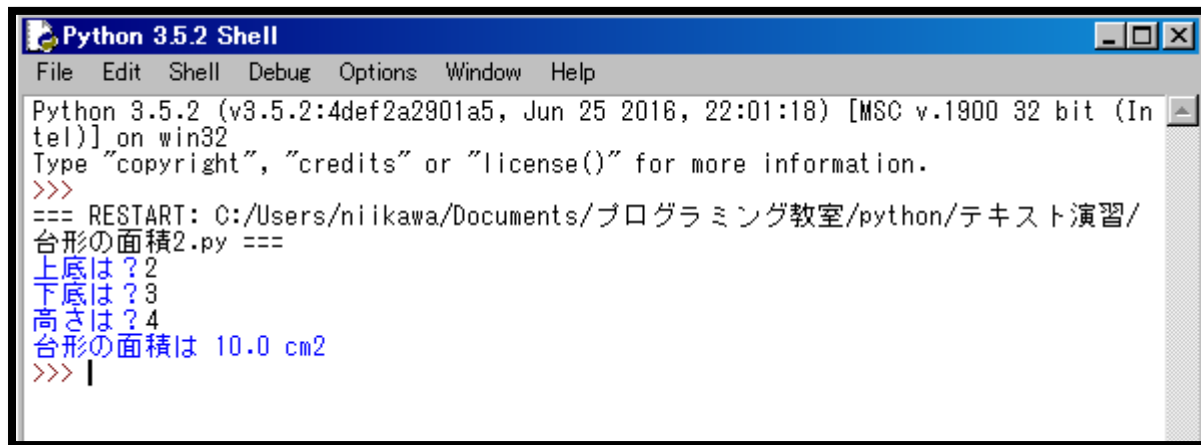
①変数 a, b, h に数字をセットしよう。

②変数 daikai に面積の計算結果をセットしよう。

③print 関数を使って、結果を表示させよう。

```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\
台形の面積1.py ===
台形の面積は 10.0 cm2
>>> |
```

④input 関数を使おう。input 関数の使い方は、P110～111 を見てね。



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/niikawa/Documents/プログラミング教室/python/テキスト演習/
台形の面積2.py ===
上底は? 2
下底は? 3
高さは? 4
台形の面積は 10.0 cm2
>>> |
```

## □練習 6 解答例

# 台形の面積を求める計算 1

a = 2

b = 3

h = 4

daikei = (a + b) \* h / 2

print("台形の面積は", daikei, "cm2")

# 台形の面積を求める計算 2

a = int(input("上底は?"))

b = int(input("下底は?"))

h = int(input("高さは?"))

daikei = (a + b) \* h / 2

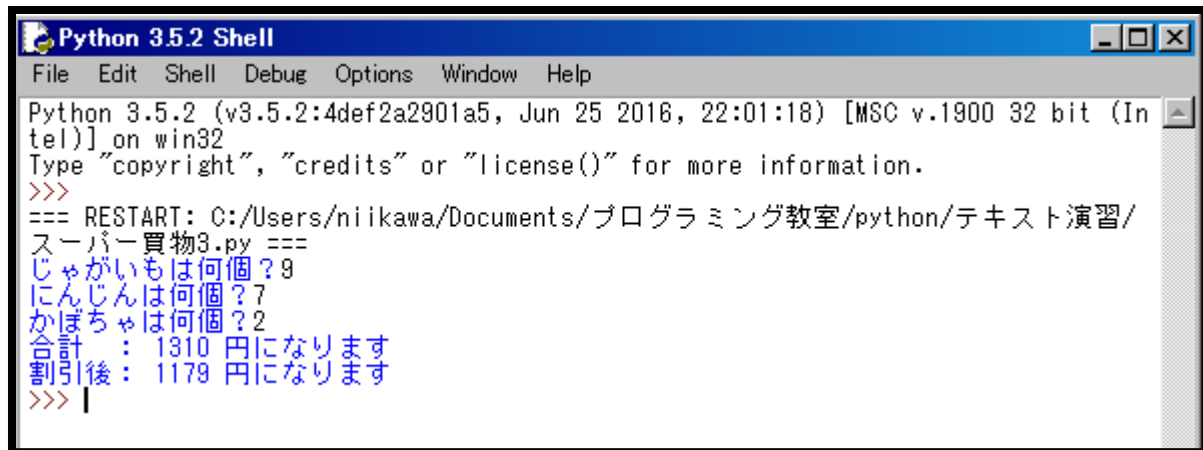
print("台形の面積は", daikei, "cm2")

## □考えよう 1

スーパーの計算をするプログラムを作ってください。じゃがいもは 50 円、にんじんは 80 円、かぼちゃは 150 円です。合計を表示するプログラムを作ってください。

合計の表示が上手く行ったら、下記の改良をしてください。

- ・じゃがいも、にんじん、かぼちゃの個数を聞いて計算する
- ・今日は土曜朝一なんで、10%オフにする。



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/niiikawa/Documents/プログラミング教室/python/テキスト演習/スーパー買物3.py ===
じゃがいもは何個? 9
にんじんは何個? 7
かぼちゃは何個? 2
合計 : 1310 円になります
割引後 : 1179 円になります
>>> |
```

## □考えよう 1 解答例

## # スーパーの計算 1

```
# Value
potato = 50
carrot = 80
pumpkin = 150
# Rate
rate = 0.9

sum=(potato * 9 + carrot * 7 + pumpkin * 2)
print("合計    :", sum, "円になります")
print("割引後:", round(sum*0.9), "円になります")
```

---

## # スーパーの計算 2

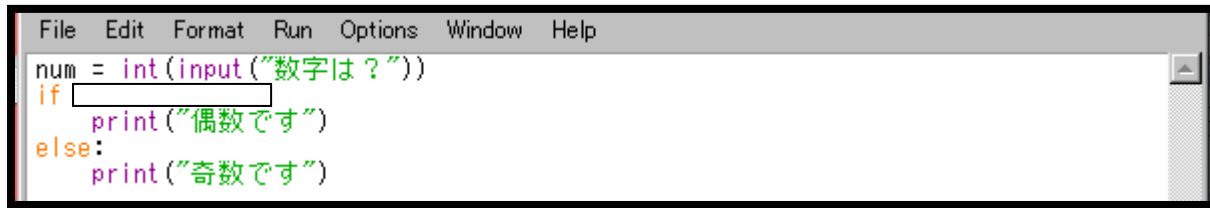
```
# Rate
rate = 0.9

# Sub-total
potato = int(input("じゃがいもは何個?")) * 50
carrot = int(input("にんじんは何個?")) * 80
pumpkin = int(input("かぼちゃは何個?")) * 150

# Total
sum=potato + carrot + pumpkin
print("合計    :", sum, "円になります")
print("割引後:", round(sum*0.9), "円になります")
```

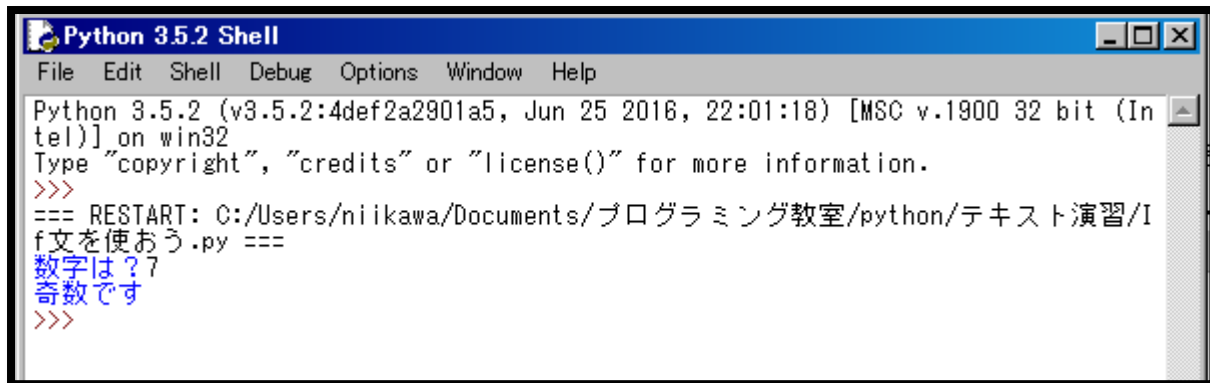
## ■練習7 if (もし…) のプログラムを作ろう

偶数と奇数を判定するプログラムを作ろう。くらべるための記号は、教科書 P118 を見て下さい。



```
File Edit Format Run Options Window Help
num = int(input("数字は？"))
if :
    print("偶数です")
else:
    print("奇数です")
```

実行結果：

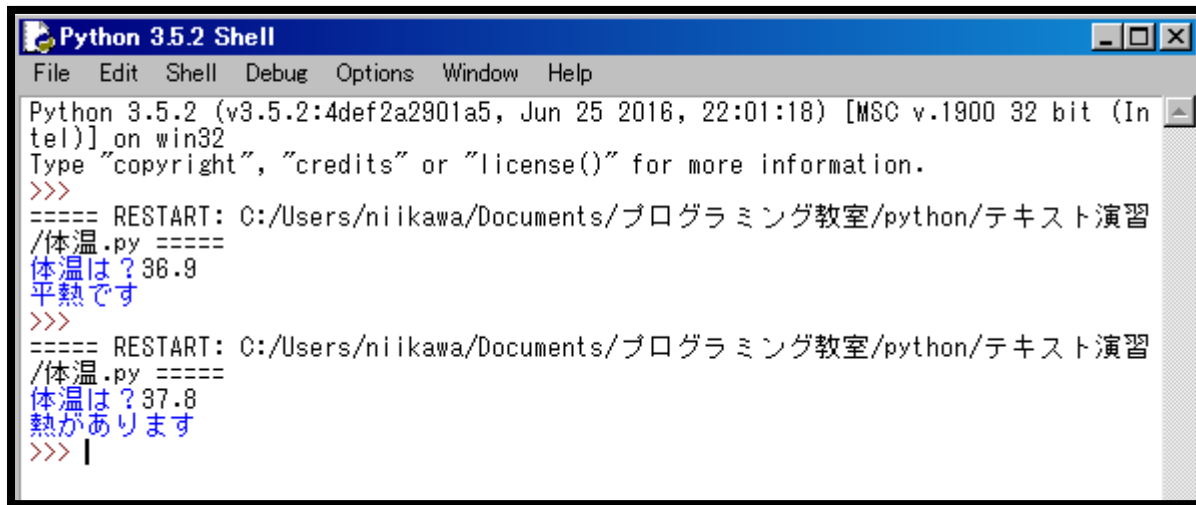


```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:/Users/niiikawa/Documents/プログラミング教室/python/テキスト演習/If文を使おう.py ===
数字は？7
奇数です
>>>
```

## □考えよう 2

体温を入力したら熱があるか平熱かを答えてくれるプログラムを作ろう。

実行結果：



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: C:/Users/niiikawa/Documents/プログラミング教室/python/テキスト演習
/体温.py =====
体温は? 36.9
平熱です
>>>
===== RESTART: C:/Users/niiikawa/Documents/プログラミング教室/python/テキスト演習
/体温.py =====
体温は? 37.8
熱があります
>>> |
```

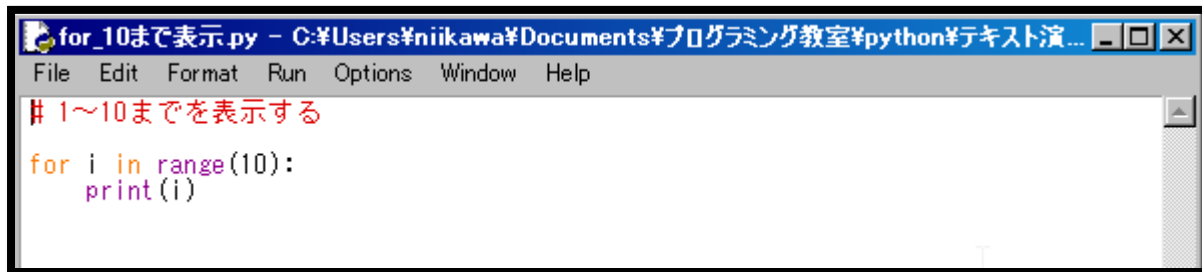


□考えよう 2 解答例

```
temp = float(input("体温は？"))
if temp >= 37:
    print("熱があります")
else:
    print("平熱です")
```

## ■練習 8 for (くり返し) のプログラムを作ろう

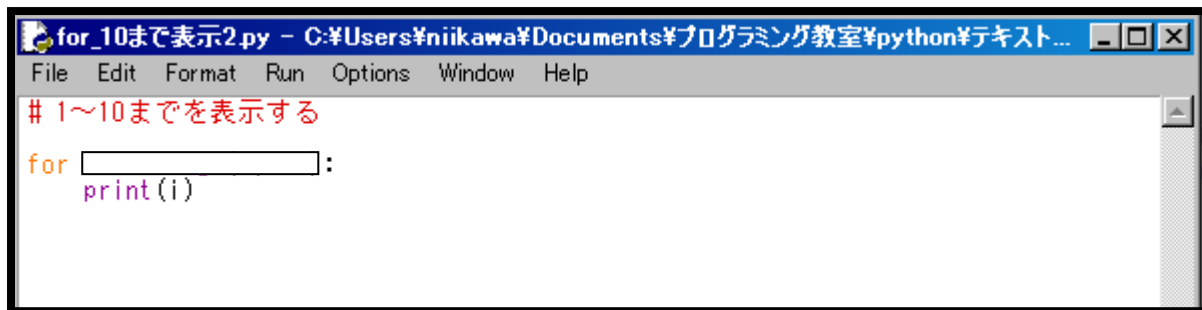
for (くり返し) を使って、1~10 までを表示させよう。for の使い方は、教科書 P122 を見て下さい。



```
for_10まで表示.py - C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演...
File Edit Format Run Options Window Help
# 1~10までを表示する
for i in range(10):
    print(i)
```

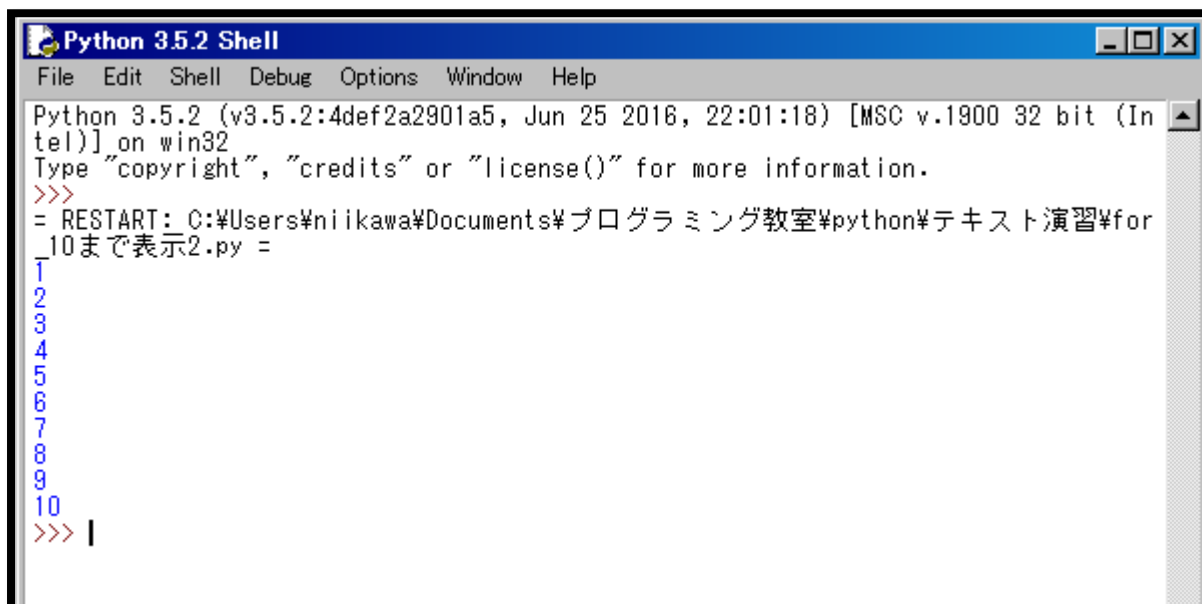
①数字はいくつからいくつまで表示されましたか？ 上手く、1~10 まで表示できたかな？

②教科書を見て、回数ではなく、開始と終了の数字を指定しよう。



```
for_10まで表示2.py - C:\Users\niikawa\Documents\プログラミング教室\python\テキスト...
File Edit Format Run Options Window Help
# 1~10までを表示する
for [input box]:
    print(i)
```

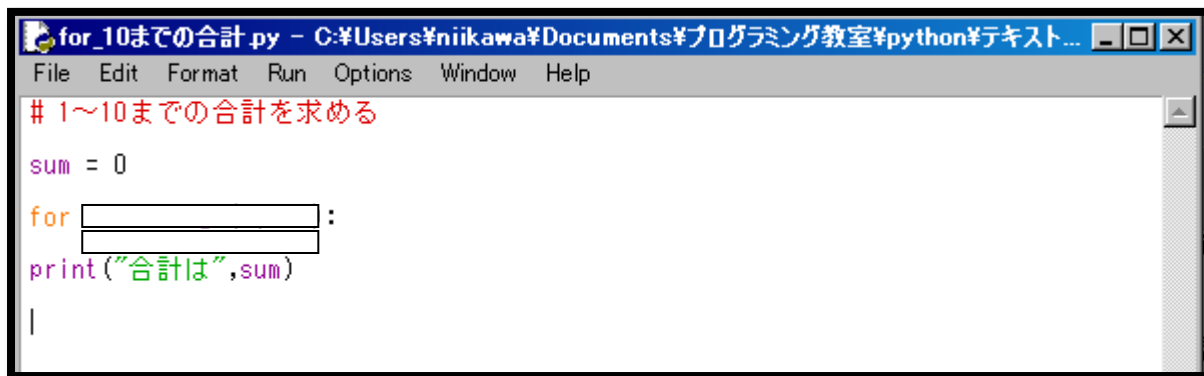
実行結果：



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\for_10まで表示2.py =
1
2
3
4
5
6
7
8
9
10
>>> |
```

## ■練習9 1～10 までの合計

for を使って、1～10 までの合計を求めよう。



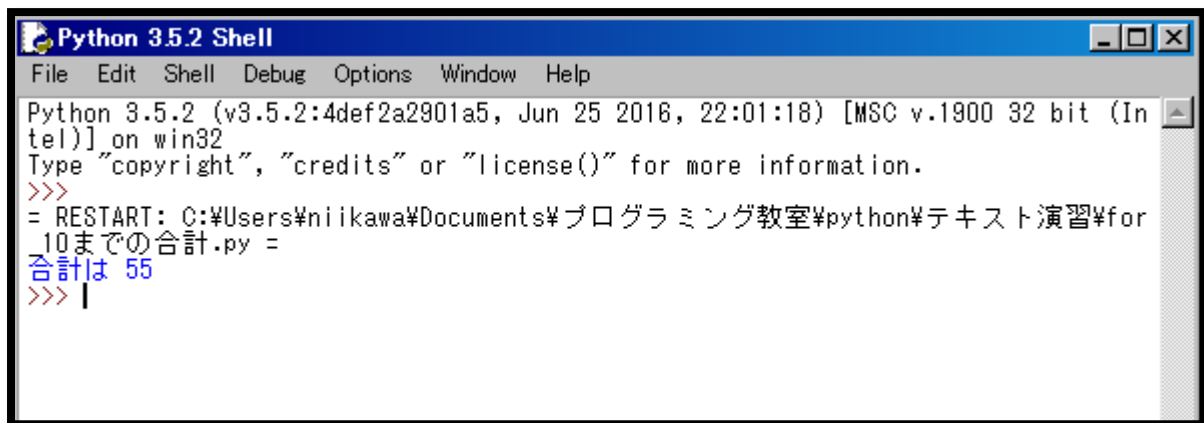
```
for_10までの合計.py - C:\Users\niikawa\Documents\プログラミング教室\python\テキスト...
File Edit Format Run Options Window Help
# 1～10までの合計を求める

sum = 0

for :
    
print("合計は",sum)

|
```

実行結果：

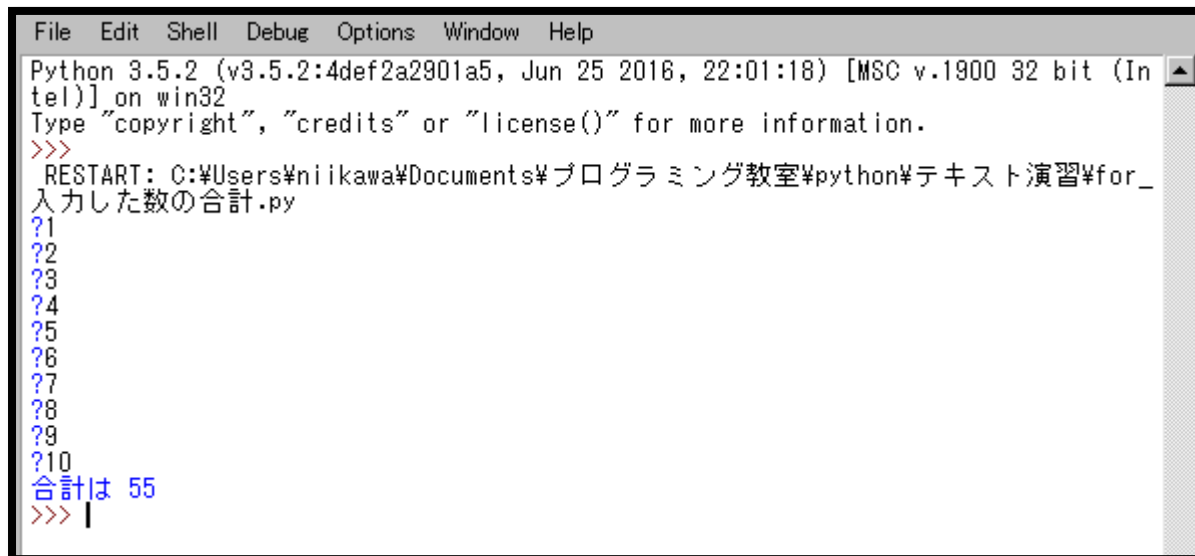


```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\for_10までの合計.py =
合計は 55
>>> |
```

□考えよう 3 入力した数の合計を求めよう

10 回数を入力して、合計を表示させるプログラムを作ろう。

実行結果：



```
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\for_
入力した数の合計.py
?1
?2
?3
?4
?5
?6
?7
?8
?9
?10
合計は 55
>>> |
```

□考えよう 3 解答例

# 入力した数の合計を求める

```
sum = 0
```

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

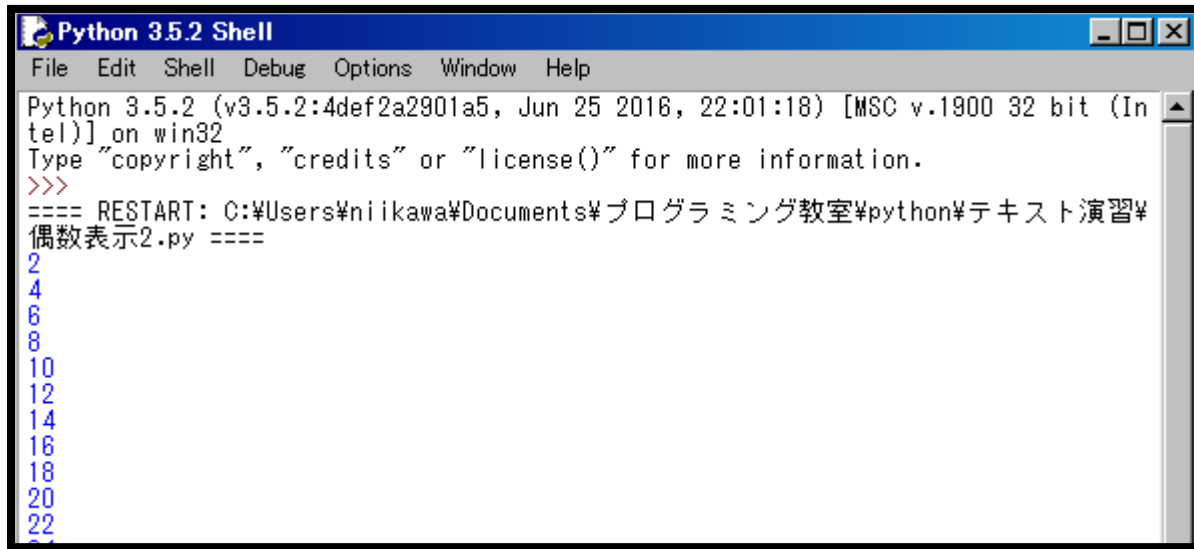
```
    sum = sum + int(input("?"))
```

```
print("合計は", sum)
```

□考えよう 4 1~100 までの偶数を表示させよう

1~100 までの数字から偶数を表示させるプログラムを作ろう。

実行結果：



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
==== RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\偶数表示2.py ====
2
4
6
8
10
12
14
16
18
20
22
```

## □考えよう 4 解答例

# 1～100 までの偶数を表示する

```
for i in range(0, 101, 2):  
    if i == 0:  
        continue  
    else:  
        print (i)
```

---

# 1～100 までの偶数を表示する

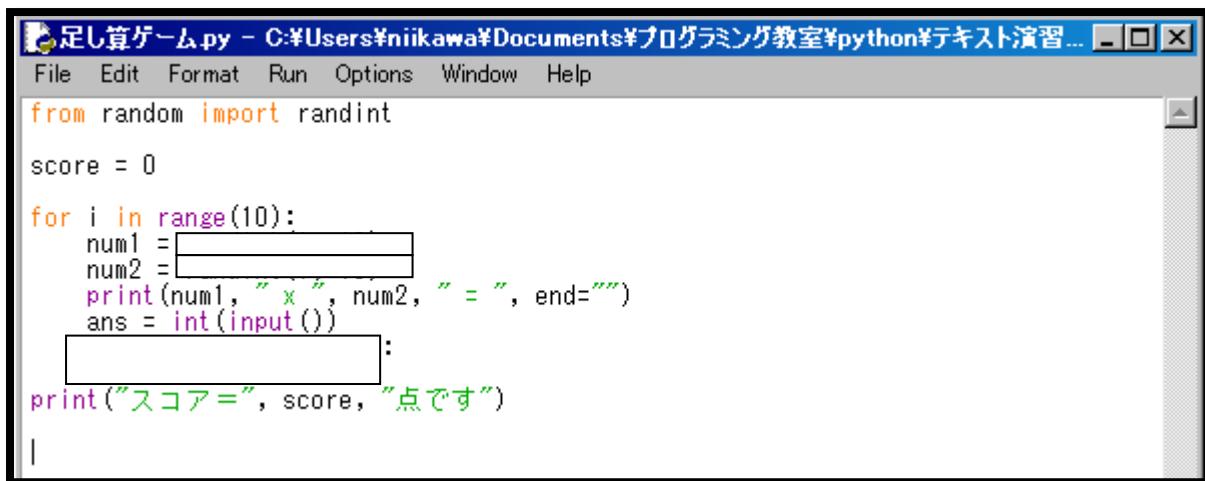
```
for i in range(1, 101):  
    if i % 2 == 0:  
        print (i)
```

## ■練習 10 ランダムを使おう

ランダムを使ったプログラムを作ろう。

プログラムの流れ：

- ①ランダムを使うため、randint 関数を読み込みます。  
「from (モジュール名) import (関数名)」
- ②変数 num1, num2 に 1~10 までのランダムな数字を入れます。  
randint の使い方は、教科書 P104 を見て下さい。
- ③変数 ans に答えを入力します。
- ④もし変数 ans と num1,num2 のかけ算が等しい場合は、変数 score に 1 を足します。



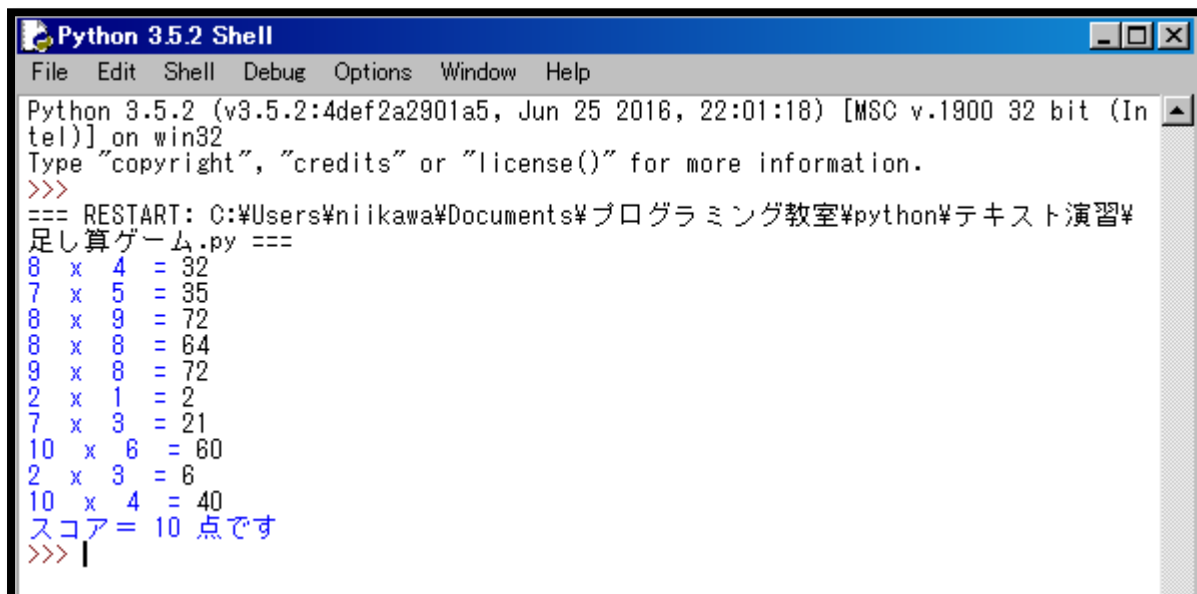
```

from random import randint

score = 0

for i in range(10):
    num1 = 
    num2 = 
    print(num1, " x ", num2, " = ", end="")
    ans = int(input())
    :
print("スコア=", score, "点です")
|
  
```

実行結果：



```

Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
=== RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\足し算ゲーム.py ===
8 x 4 = 32
7 x 5 = 35
8 x 9 = 72
8 x 8 = 64
9 x 8 = 72
2 x 1 = 2
7 x 3 = 21
10 x 6 = 60
2 x 3 = 6
10 x 4 = 40
スコア = 10 点です
>>> |
  
```



■練習 11 ゆうれいゲームに挑戦

教科書 P96 を見ながら、ゆうれいゲームを作ってみよう。

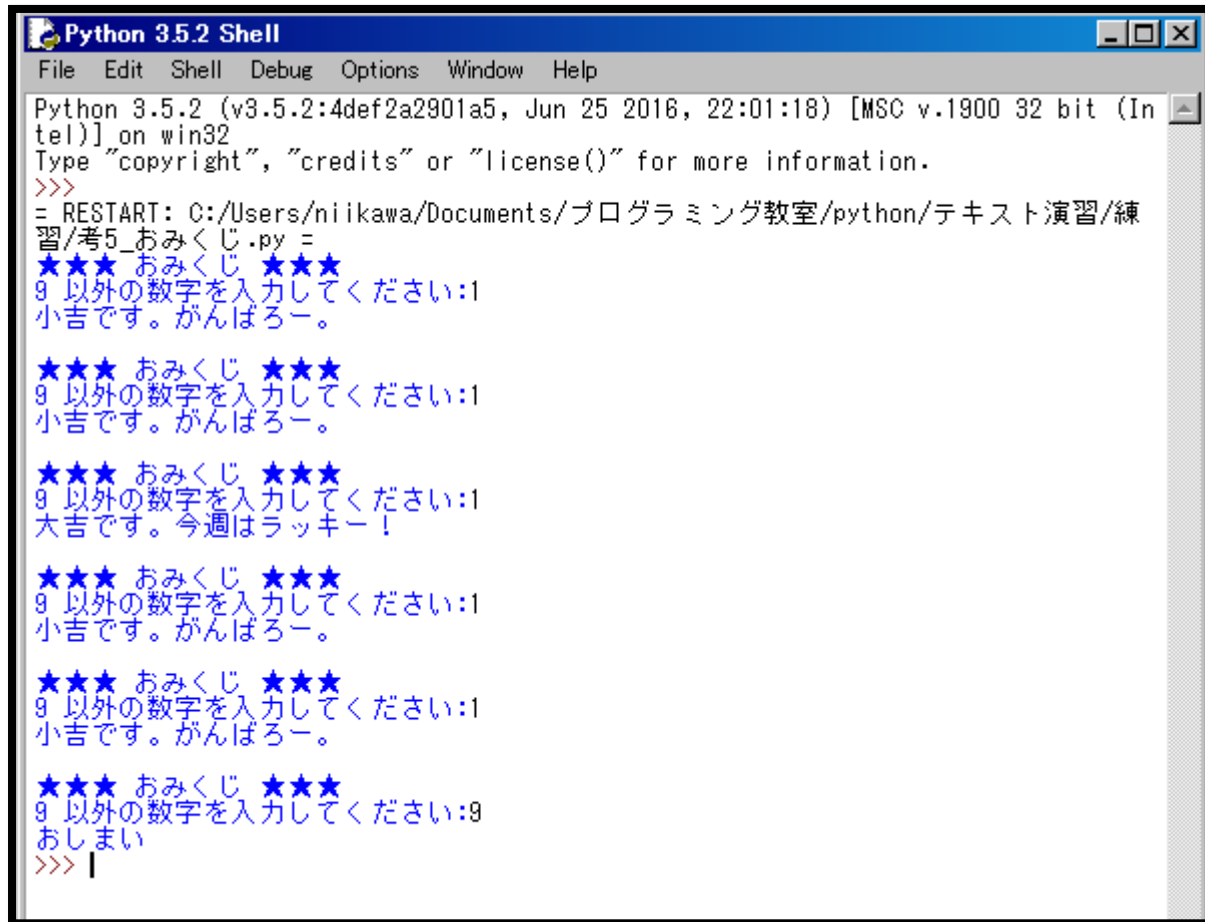
- ①ゲームは、上手く動いたかな？
- ②どこからどこまでがくり返しかな？ 説明してください。
- ③くり返しが終わる条件は何か？
- ④プログラムのどこを通ると、くり返しから抜けますか？

## □考えよう 5 おみくじを作ろう

ランダムを使って、おみくじを作ろう。

randint のランダムの他に、while のくり返しと if の条件分岐を使います。

実行結果：



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:/Users/niikawa/Documents/プログラミング教室/python/テキスト演習/練習/考5_おみくじ.py =
★★★★ おみくじ ★★★★★
9 以外の数字を入力してください:1
小吉です。がんばろー。

★★★★ おみくじ ★★★★★
9 以外の数字を入力してください:1
小吉です。がんばろー。

★★★★ おみくじ ★★★★★
9 以外の数字を入力してください:1
大吉です。今週はラッキー！

★★★★ おみくじ ★★★★★
9 以外の数字を入力してください:1
小吉です。がんばろー。

★★★★ おみくじ ★★★★★
9 以外の数字を入力してください:1
小吉です。がんばろー。

★★★★ おみくじ ★★★★★
9 以外の数字を入力してください:9
おしまい
>>> |
```

## □考えよう 5 解答例

# おみくじ

```
from random import randint

while True:
    print("★★★★ おみくじ ★★★★★")
    if int(input("9 以外の数字を入力してください:")) == 9:
        break

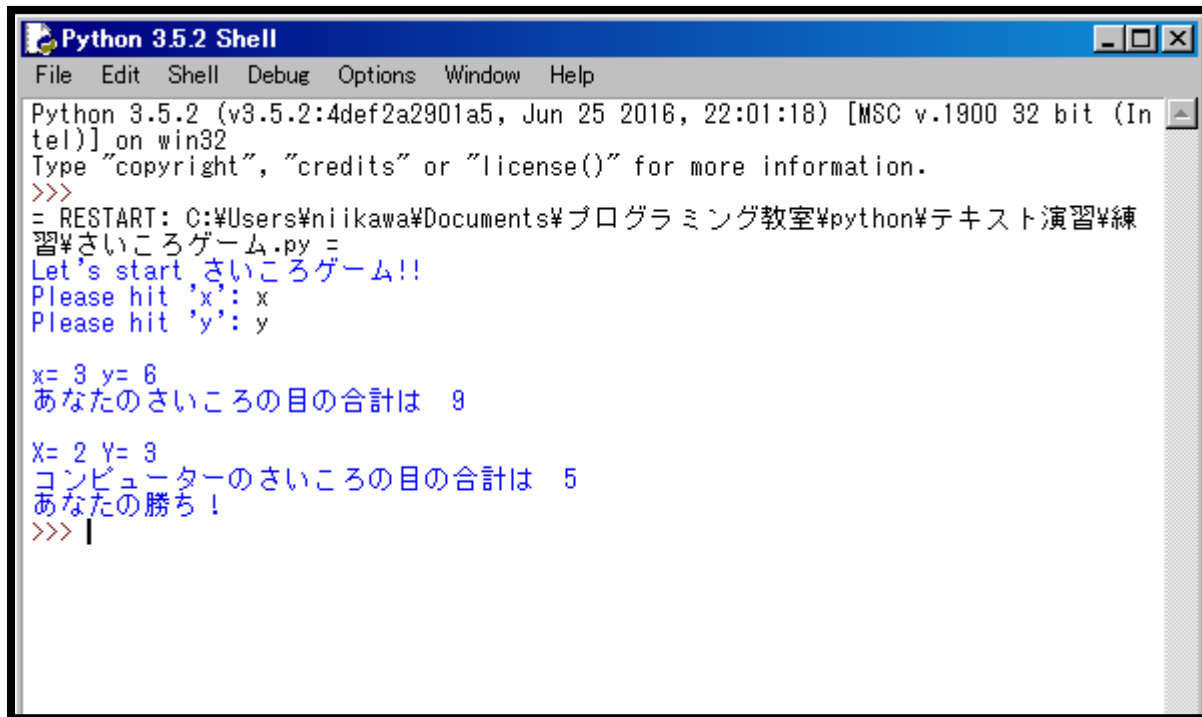
    kuji = randint(1, 3)
    if kuji == 1:
        print("大吉です。今週はラッキー!")
    elif kuji == 2:
        print("小吉です。がんばろー。")
    else:
        print("がびーん。凶です。今日は早めに寝よう。")
    print("")

print("おしまい")
```

## □考えよう 6 サイコロゲーム

2回サイコロをふって出た数の合計を人間とコンピューターで競うゲームを作ろう。

実行結果：



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\練習\さいころゲーム.py =
Let's start,さいころゲーム!!
Please hit 'x': x
Please hit 'y': y

x= 3 y= 6
あなたのさいころの目の合計は 9

X= 2 Y= 3
コンピューターのさいころの目の合計は 5
あなたの勝ち！
>>> |
```

## □考えよう 6 解答例

```
# さいころゲーム

from random import randint

print("Let's start さいころゲーム!!")

a = 0

while a != 'x':
    a=input("Please hit 'x': ")
    x = randint(1, 6)

while a != 'y':
    a=input("Please hit 'y': ")
    y = randint(1, 6)

print('')
print('x=', x, 'y=', y, sep=' ')
print('あなたのさいころの目の合計は ', x+y)

X = randint(1, 6)
Y = randint(1, 6)

print('')
print('X=', X, 'Y=', Y, sep=' ')
print('コンピューターのさいころの目の合計は ', X+Y)

if x+y > X+Y:
    print('あなたの勝ち!')
elif x+y == X+Y:
    print('引きわけね...')
else:
    print('コンピューターの勝ちよ!!')
```

## ■練習 12 グラフィカルな線を引こう

縦線を引いてくれる GUI アプリを作ろう。GUI アプリを作るため、グラフィックライブラリ tkinter を使います。

プログラムの流れ：

①tkinter のすべての関数を読み込みます。

「from (モジュール名) import (関数名)」

②tkinter が描くためのキャンバス (Canvas) をセットします。

幅 (width) = 900 ピクセル、高さ (height) = 400 ピクセルにします。

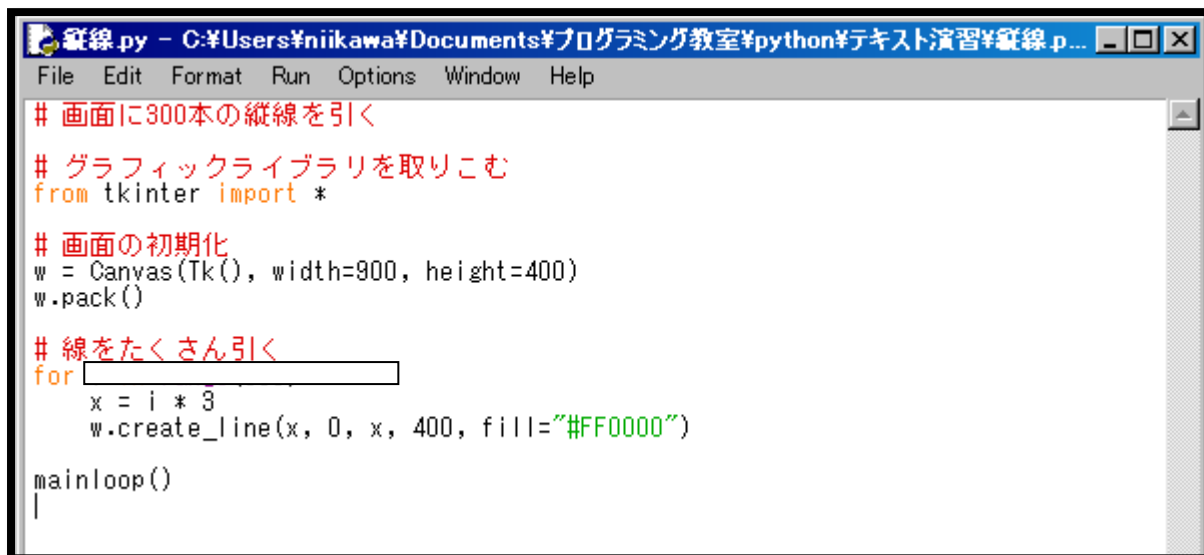
③for のループを使って、線をたくさん引くことにします。

ループ変数 i を使って 300 回繰り返します。ループ変数 i は、X 座標を作るのにも使います。

④w.create\_line で縦線を引きます。

( ) の中身は、(開始の X 座標, 開始の Y 座標, 終了の X 座標, 終了の Y 座標, 色)を表します。

FF0000 は赤色になります。



```
# 縦線.py - C:\Users\niikawa\Documents\プログラミング教室\python\テキスト演習\縦線.p...
File Edit Format Run Options Window Help

# 画面に300本の縦線を引く

# グラフィックライブラリを取りこむ
from tkinter import *

# 画面の初期化
w = Canvas(Tk(), width=900, height=400)
w.pack()

# 線をたくさん引く
for i in range(300):
    x = i * 3
    w.create_line(x, 0, x, 400, fill="#FF0000")

mainloop()
|
```

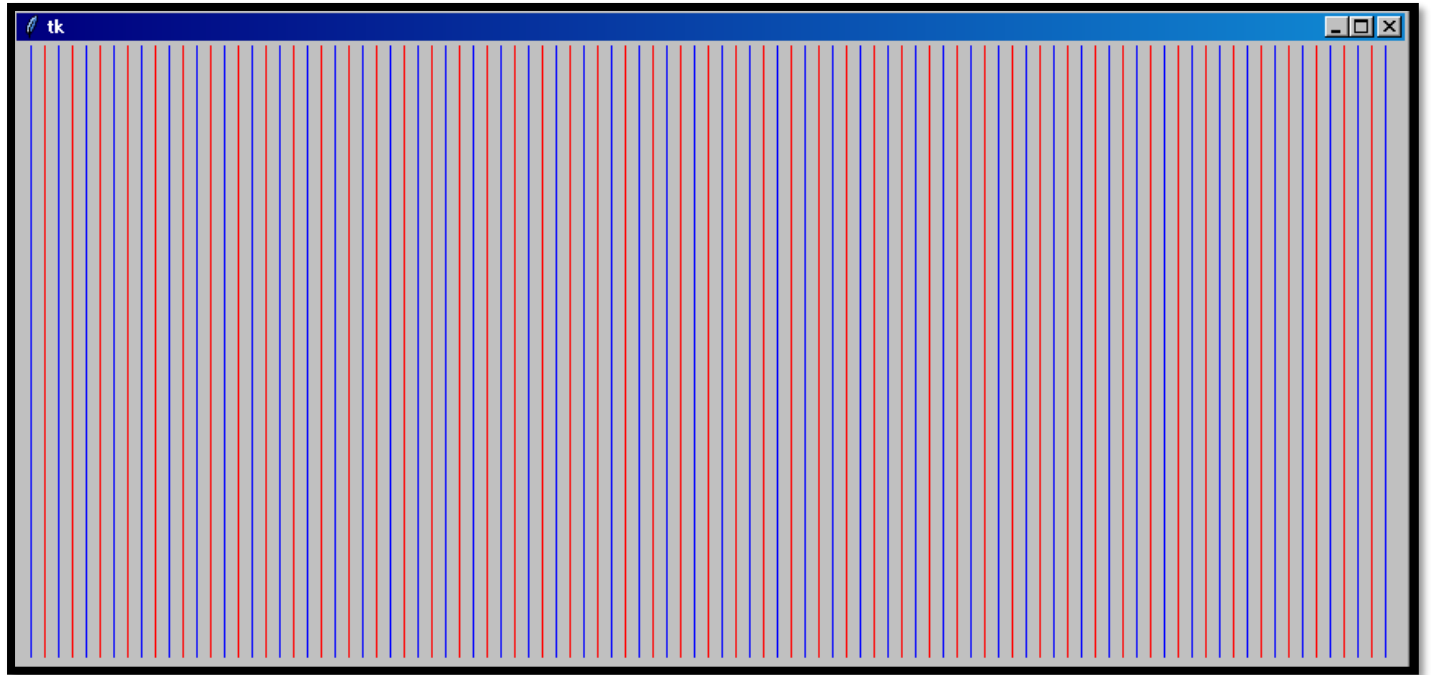
□考えよう 7 グラフィカルな線を引こう パート 2

赤色と青色で交互に縦線を引く GUI アプリを作ろう。

縦線は合計で、100 本引いて下さい。

なお、FF0000 は赤色、0000FF は青色になります。

実行結果：



## □考えよう 7 解答例

```
# 画面に 300 本の縦線を引く
```

```
# グラフィックライブラリを取りこむ
```

```
from tkinter import *
```

```
# 画面の初期化
```

```
w = Canvas(Tk(), width=900, height=400)
```

```
w.pack()
```

```
# 線をたくさん引く
```

```
for i in range(100):
```

```
    x = i * 9
```

```
    if i % 2 == 0:
```

```
        c = "#FF0000"
```

```
    else:
```

```
        c = "#0000FF"
```

```
    w.create_line(x, 0, x, 400, fill=c)
```

```
mainloop()
```

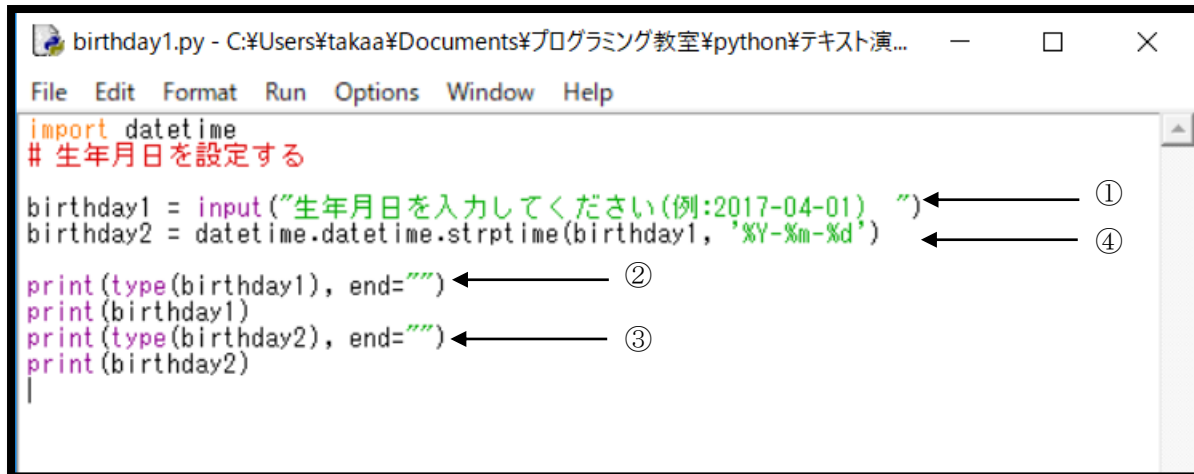


## ■練習 13 日付を使ったプログラムを作ろう

datetime モジュールを読み込み、日付を使ったプログラムを作ろう。

プログラムの流れ：

- ① input 関数を使って、生年月日を入力して birthday1 変数に入れる。
- ② datetime 関数を使って、birthday1 変数のフォーマットを変換して、birthday2 に入れる。
- ③ type と print 関数を使って、birthday1 変数が何型かを調べる。birthday1 変数の中身を表示する。
- ④ type と print 関数を使って、birthday2 変数が何型かを調べる。birthday2 変数の中身を表示する。



```
import datetime
# 生年月日を設定する

birthday1 = input("生年月日を入力してください(例:2017-04-01) ")
birthday2 = datetime.datetime.strptime(birthday1, '%Y-%m-%d')

print(type(birthday1), end=" ")
print(birthday1)
print(type(birthday2), end=" ")
print(birthday2)
```

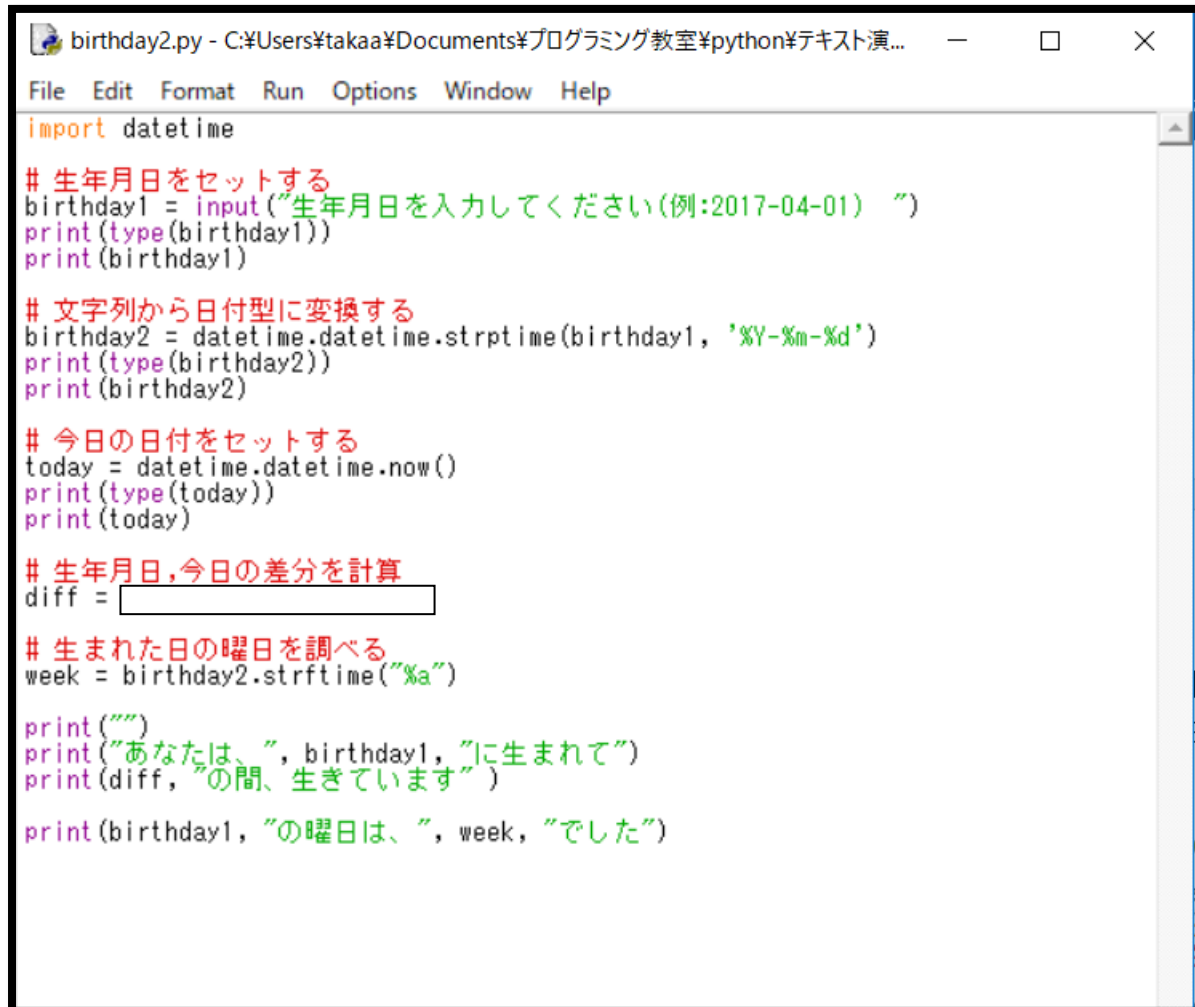
質問：

birthday1 変数、birthday2 変数は、それぞれどんな型でしょうか？

なぜ、②のフォーマット変換が必要だと思う？

## ■練習 14 あなたは、生まれて何日目？

今日の日付とあなたの生年月日の差分を計算して、生まれて何日目かを計算しよう。



```
birthday2.py - C:\Users\takaa\Documents\プログラミング教室\python\テキスト演...
File Edit Format Run Options Window Help

import datetime

# 生年月日をセットする
birthday1 = input("生年月日を入力してください(例:2017-04-01) ")
print(type(birthday1))
print(birthday1)

# 文字列から日付型に変換する
birthday2 = datetime.datetime.strptime(birthday1, '%Y-%m-%d')
print(type(birthday2))
print(birthday2)

# 今日の日付をセットする
today = datetime.datetime.now()
print(type(today))
print(today)

# 生年月日,今日の差分を計算
diff = 

# 生まれた日の曜日を調べる
week = birthday2.strftime("%a")

print("")
print("あなたは、", birthday1, "に生まれて")
print(diff, "の間、生きています")

print(birthday1, "の曜日は、", week, "でした")
```

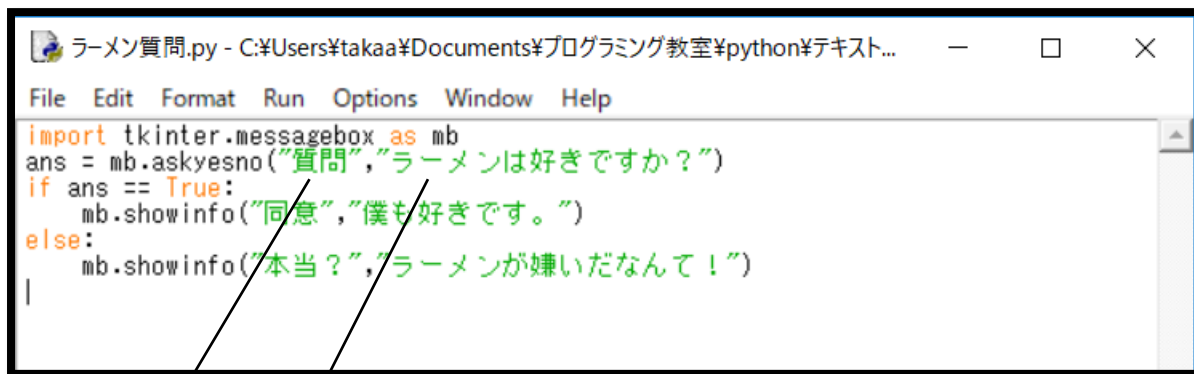
## ■練習 15 メッセージボックスを使ってみよう

メッセージボックスを使ったプログラムを作ろう。

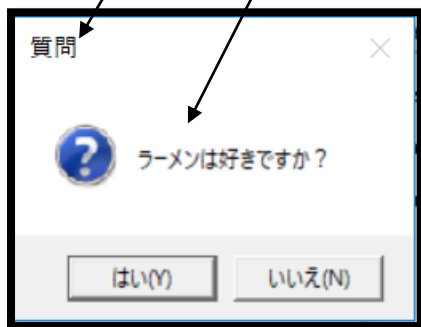
tkinter の messagebox モジュールを使うと、下記ダイアログの関数を使うことができる。

下記の関数を実行すると、メッセージを表示してユーザーがボタンを押すまで待ちます。

askokcancel()	ok, cancel ボタン (ok => True, cancel => False)
askquestion()	yes, no ボタン (ok => 'yes', no => 'no')
askretrycancel()	retry, cancel ボタン (retry => True, cancel => False)
askyesno()	yes, no ボタン (yes => True, no => False)
showerror()	ok ボタン (ok => 'ok')
showinfo()	ok ボタン (ok => 'ok')
showwarning()	ok ボタン (ok => 'ok')

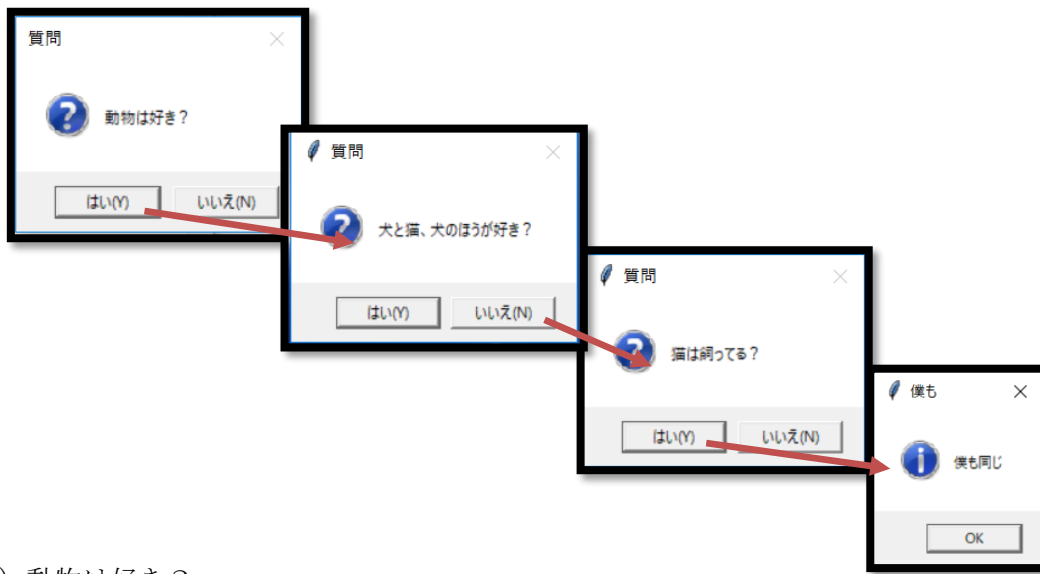


```
import tkinter.messagebox as mb
ans = mb.askyesno("質問", "ラーメンは好きですか？")
if ans == True:
    mb.showinfo("同意", "僕も好きです。")
else:
    mb.showinfo("本当？", "ラーメンが嫌いだなって！")
```



□考えよう 8 どんどんでてくる質問プログラムを作ろう

メッセージボックスを使って、どんどんでてくる質問するプログラムを作ろう。



例) 動物は好き?

はい → 「犬と猫、犬の方が好き?」

はい → 「犬は飼ってる?」

はい → 「いいなー」と表示

いいえ → 「猫飼ってる?」

はい → 「私と同じ」と表示

いいえ → 「私は猫を飼ってるよ」と表示

…など

## □考えよう 8 解答例

```
import tkinter.messagebox as mb

ans = mb.askyesno('質問', '動物は好き?')

if ans == True:
    ans = mb.askyesno('質問', '犬と猫、犬のほうが好き?')
    if ans == True:
        ans = mb.askyesno('質問', '犬は飼ってる?')
        if ans == True:
            mb.showinfo('ほんと!?', 'いいなー')
        else:
            mb.showinfo('ほんと!?', '飼ってないんだ')
    else:
        ans = mb.askyesno('質問', '猫は飼ってる?')
        if ans == True:
            mb.showinfo('僕も', '僕も同じ')
        else:
            mb.showinfo('ほんと!?', '飼ってないんだ')
else:
    mb.showinfo('僕は', '僕は猫を飼っている')
```

## ■練習 16 and / or を使った条件を作ろう

- ①1～20 までの数字のなかで 2 または 3 で割り切れる数を表示させよう。条件に or を使います。  
 ②実行結果は正しく表示されたかな。

```

2または3で割り切れる数.py - C:\Users\takaa\Documents\プログラミング教室\python...
File Edit Format Run Options Window Help
# 1～100までの数字で2または3で割り切れる数を表示する
for i in range (1,21):
    if  or :
        print(i)
  
```

実行結果：

```

Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\2または3で割り切れる数.py =
2
3
4
6
8
9
10
12
14
15
16
18
20
>>>
  
```

- ③次に、1～20 までの数字のなかで 2 と 3 両方で割り切れる数を表示させよう。条件に and を使います。  
 ④実行結果は正しく表示されたかな。

```

== RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\2と3で割り切れる数.py ==
6
12
18
>>> |
  
```

□考えよう 9 うるう年判定プログラムを作ろう

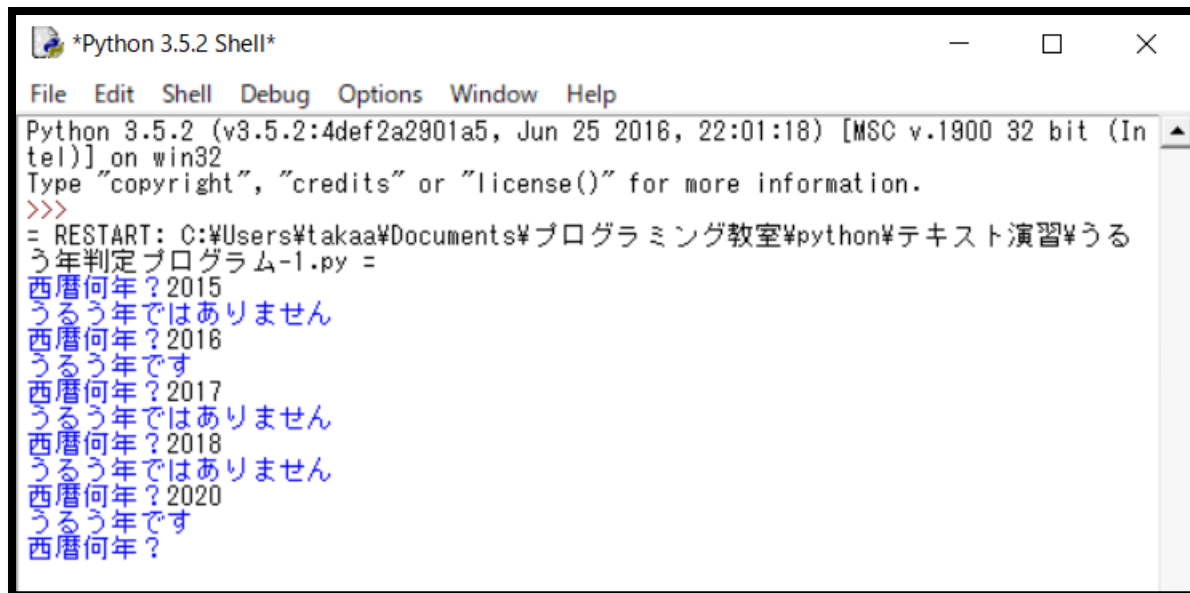
うるう年を知っているかな? 2月29日は4年に1度しか来ません。

この2月29日がある年を、うるう年といいます。うるう年の条件は、下記の3つです。

- ・年が4で割り切れれば「うるう年」
- ・ただし、100で割り切れる年は「うるう年ではない」
- ・ただし、400で割り切れれば「うるう年」

※上記条件は、if と else を繰り返して作ることもできますが、ここでは and や or を使って作ろう。

実行結果：



```
*Python 3.5.2 Shell*
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\うるう年判定プログラム-1.py =
西暦何年? 2015
うるう年ではありません
西暦何年? 2016
うるう年です
西暦何年? 2017
うるう年ではありません
西暦何年? 2018
うるう年ではありません
西暦何年? 2020
うるう年です
西暦何年?
```

## □考えよう 9 解答例

```
while True:
    year = int(input("西暦何年？"))
    if (year%400 == 0) or ((year%4 == 0) and (year%100 != 0)):
        print("うるう年です")
    else:
        print("うるう年ではありません")
```



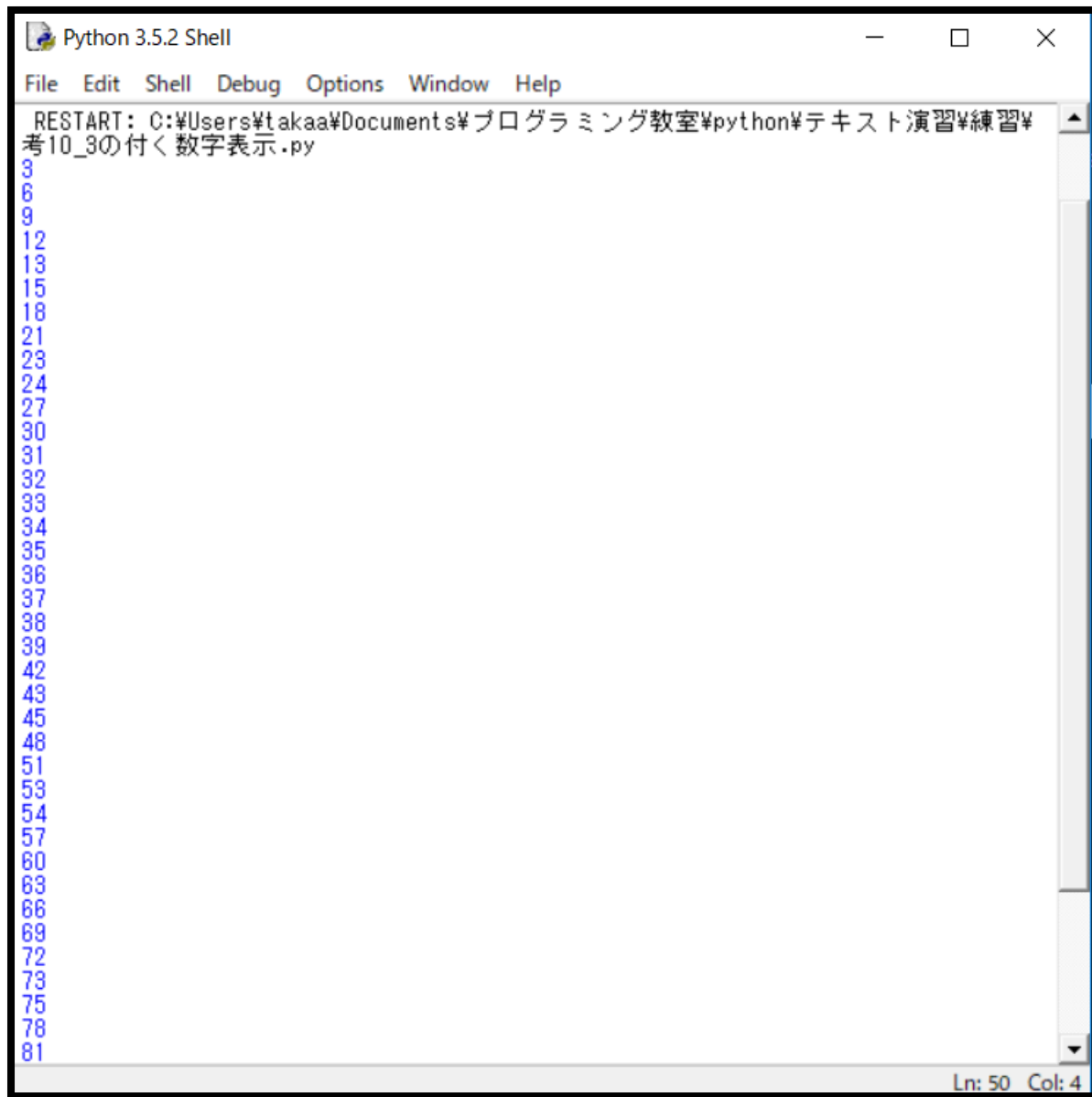
□ 考えよう 10 3 の付く数字を表示させよう

1~100 までの 3 の倍数か 3 の付く数字を表示させよう (13、23 や 31、32 も含まれるよ)。

せっかくだから、声を出してチェックしよう。(さん, ろく, きゅう, じゅうに…)

実行結果：

(実際の結果は、99 まで続きます)



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習\
考10_3の付く数字表示.py
3
6
9
12
13
15
18
21
23
24
27
30
31
32
33
34
35
36
37
38
39
42
43
45
48
51
53
54
57
60
63
66
69
72
73
75
78
81
Ln: 50 Col: 4
```

## □考えよう 10 解答例

# 1～100 までの 3 の倍数か 3 の付く数字を表示する

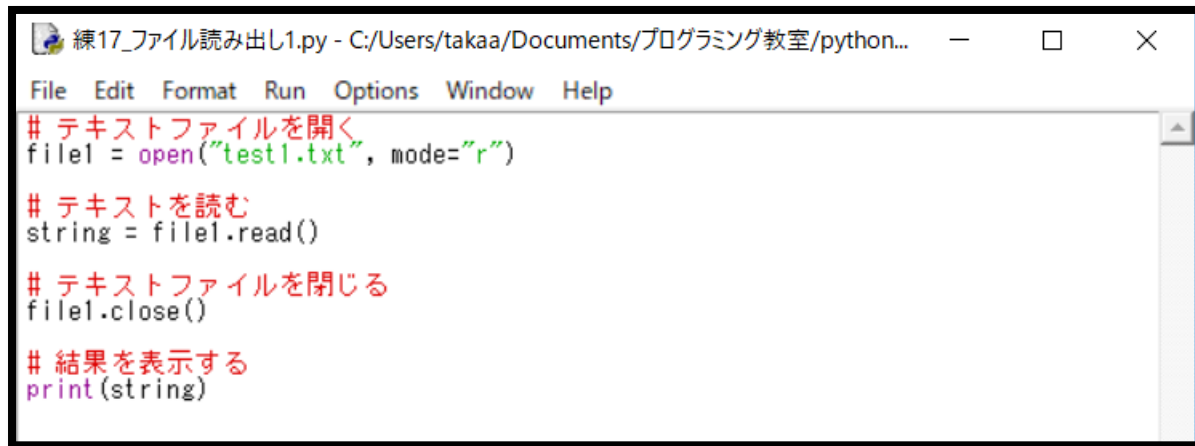
```
for i in range(1, 101):  
    if i % 3 == 0 or i % 10 == 3 or (i >= 30 and i <= 39):  
        print (i)
```

## ■練習 17 ファイルからテキストを読み出そう

プログラムと同じフォルダにテキストファイル (test1.txt) を作成し、好きな文字列を書きましょう。

open 関数は、mode 引数に r を指定するとファイルを読み出しモード開きます。w を指定すると書き込みモードで開きます (すでにファイルが存在すると内容を消して上書きします)。

a を指定すると書き込みモードで開き、すでにファイルが存在すると追加で書き込みます。

A screenshot of a Python IDE window titled '練17\_ファイル読み出し1.py - C:/Users/takaa/Documents/プログラミング教室/python...'. The window contains a Python script with the following code:

```
# テキストファイルを開く
file1 = open("test1.txt", mode="r")

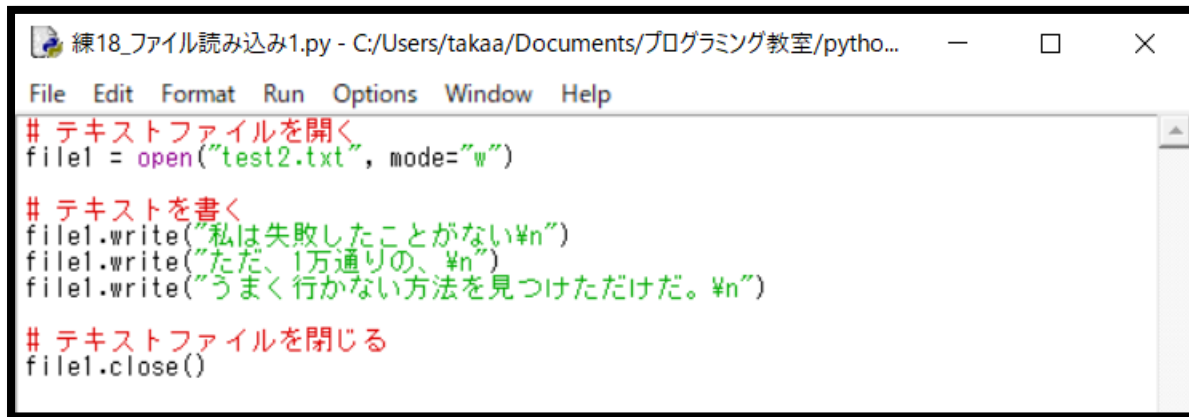
# テキストを読む
string = file1.read()

# テキストファイルを閉じる
file1.close()

# 結果を表示する
print(string)
```

## ■練習 18 ファイルにテキストを書き込もう

プログラムを実行したら、test2.txt を開き内容を確認しよう。



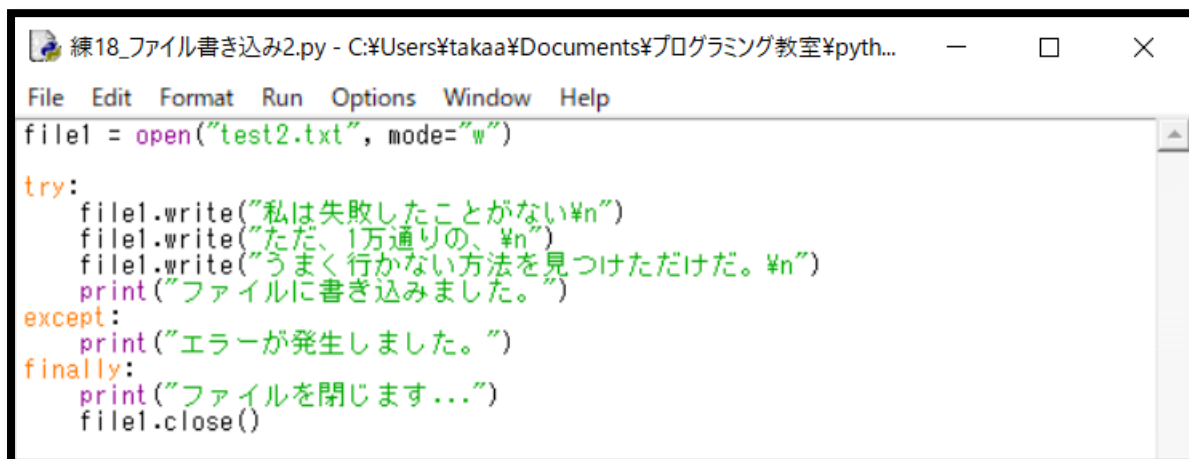
```
練18_ファイル読み込み1.py - C:/Users/takaa/Documents/プログラミング教室/pytho...
File Edit Format Run Options Window Help
# テキストファイルを開く
file1 = open("test2.txt", mode="w")

# テキストを書く
file1.write("私は失敗したことがない\n")
file1.write("ただ、1万通りの、\n")
file1.write("うまく行かない方法を見つけたただけだ。 \n")

# テキストファイルを閉じる
file1.close()
```

次にエラー処理を付け加えよう。下記のように try..finally.. 構文を追加するだけで、ファイルを開いた後に別の誰かがファイルを削除しても、確実にファイルを閉じることができるよ。

また、except 構文はエラーの時だけ実行されます。



```
練18_ファイル書き込み2.py - C:\Users\takaa\Documents\プログラミング教室\pyth...
File Edit Format Run Options Window Help
file1 = open("test2.txt", mode="w")

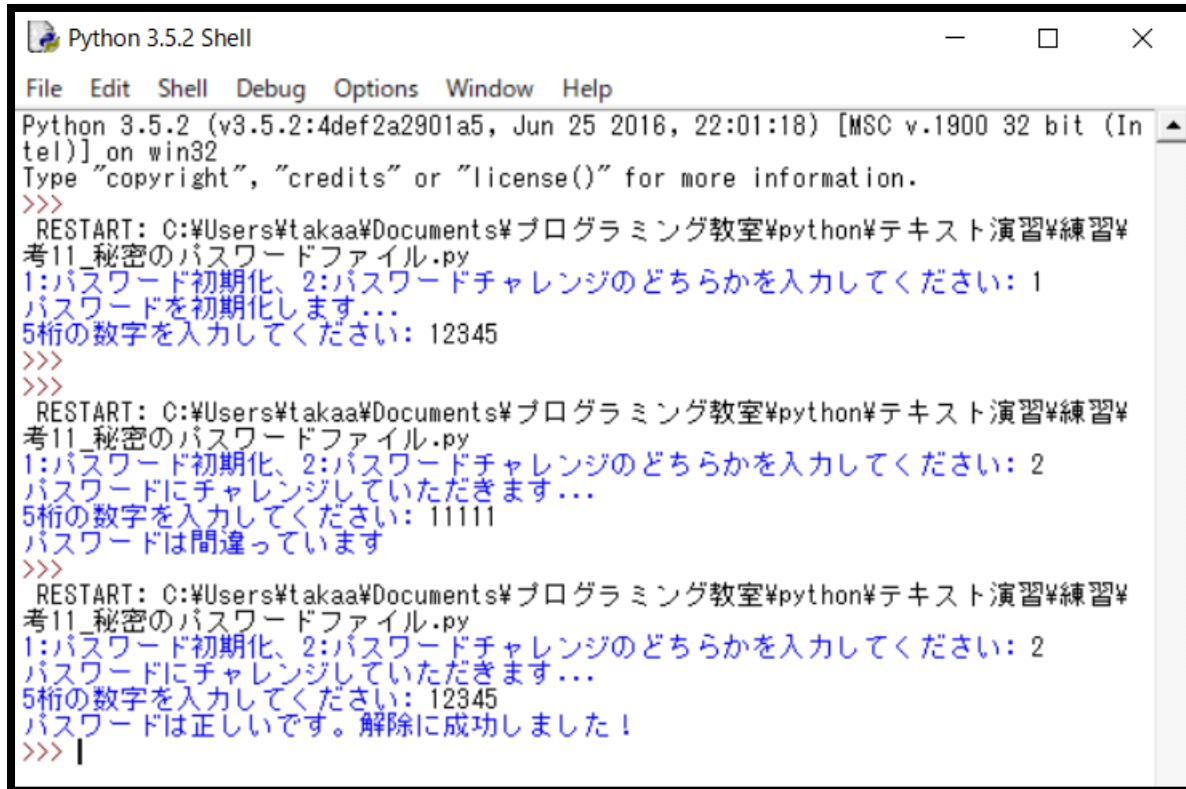
try:
    file1.write("私は失敗したことがない\n")
    file1.write("ただ、1万通りの、\n")
    file1.write("うまく行かない方法を見つけたただけだ。 \n")
    print("ファイルに書き込みました。")
except:
    print("エラーが発生しました。")
finally:
    print("ファイルを閉じます...")
    file1.close()
```

## □考えよう 11 秘密のパスワードファイルを作るアプリを作ろう

パスワードを作成する処理（ファイルに書き出す）、パスワードが正しいかをチェックする処理（ファイルから読み出した数字と入力した数字が等しいかを判定する）を作成します。

それぞれの処理をメニューから選ぶようにします。

実行結果：



```
Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習\考11_秘密のパスワードファイル.py
1:パスワード初期化、2:パスワードチャレンジのどちらかを入力してください: 1
パスワードを初期化します...
5桁の数字を入力してください: 12345
>>>
RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習\考11_秘密のパスワードファイル.py
1:パスワード初期化、2:パスワードチャレンジのどちらかを入力してください: 2
パスワードにチャレンジしていただきます...
5桁の数字を入力してください: 11111
パスワードは間違っています
>>>
RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習\考11_秘密のパスワードファイル.py
1:パスワード初期化、2:パスワードチャレンジのどちらかを入力してください: 2
パスワードにチャレンジしていただきます...
5桁の数字を入力してください: 12345
パスワードは正しいです。解除に成功しました！
>>> |
```

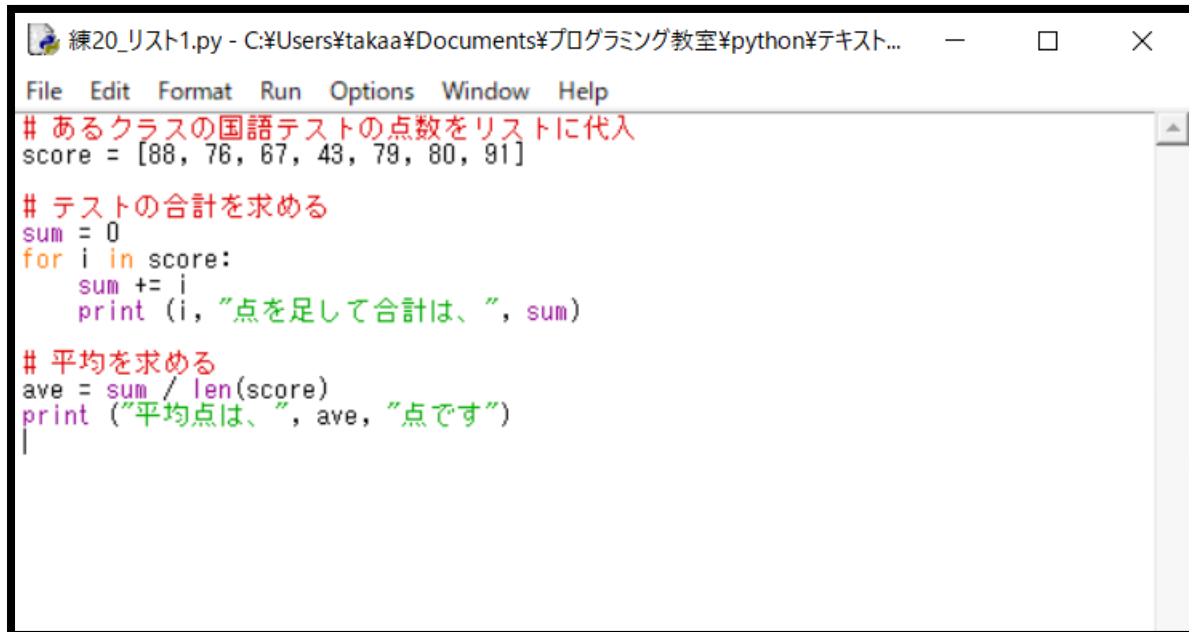
## □考えよう 11 解答例

```
if int(input("1:パスワード初期化、2:パスワードチャレンジのどちらかを入力してください: ")) == 1:
    print("パスワードを初期化します...")
    with open("pw.txt", "w") as f:
        string1 = input("5桁の数字を入力してください: ")
        f.write(string1)
else:
    print("パスワードにチャレンジしていただきます...")
    with open("pw.txt", "r") as f:
        string1 = f.read()
        if int(input("5桁の数字を入力してください: ")) == int(string1):
            print("パスワードは正しいです。解除に成功しました!")
        else:
            print("パスワードは間違っています")
```

## ■練習 20 リストを使おう

リストを使って、1つの変数で複数の値を管理できます。複数の値を1つの変数で管理できるので、便利です。角括弧 ([ ]) の中に、コンマで区切って複数の値を記述します。

また、len()関数を使うことでリスト内の値の個数を調べることができます。



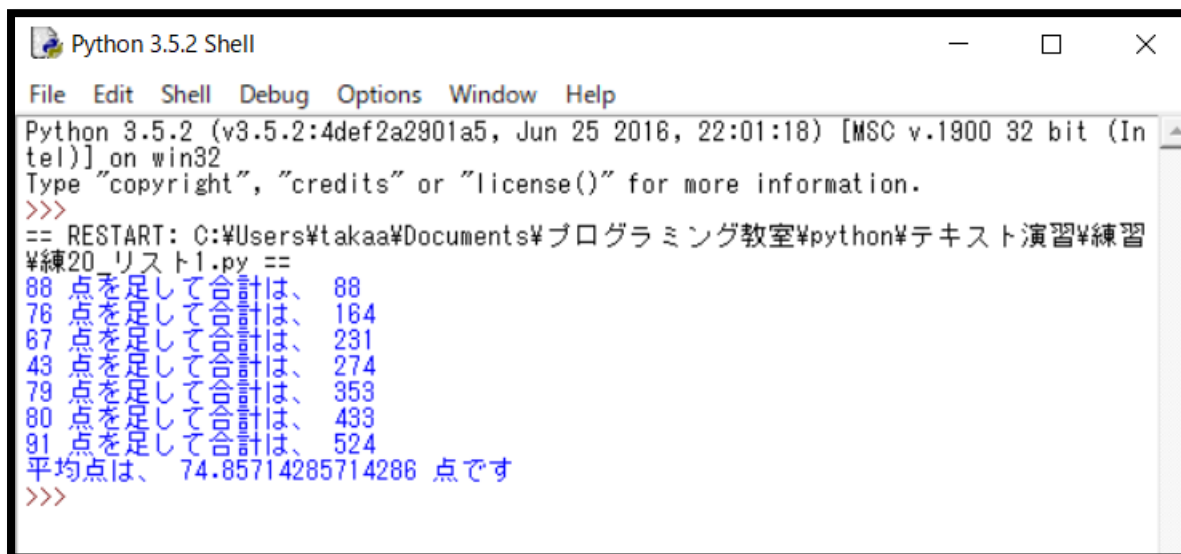
```

File Edit Format Run Options Window Help
# あるクラスの国語テストの点数をリストに代入
score = [88, 76, 67, 43, 79, 80, 91]

# テストの合計を求める
sum = 0
for i in score:
    sum += i
    print(i, "点を足して合計は、", sum)

# 平均を求める
ave = sum / len(score)
print("平均点は、", ave, "点です")

```



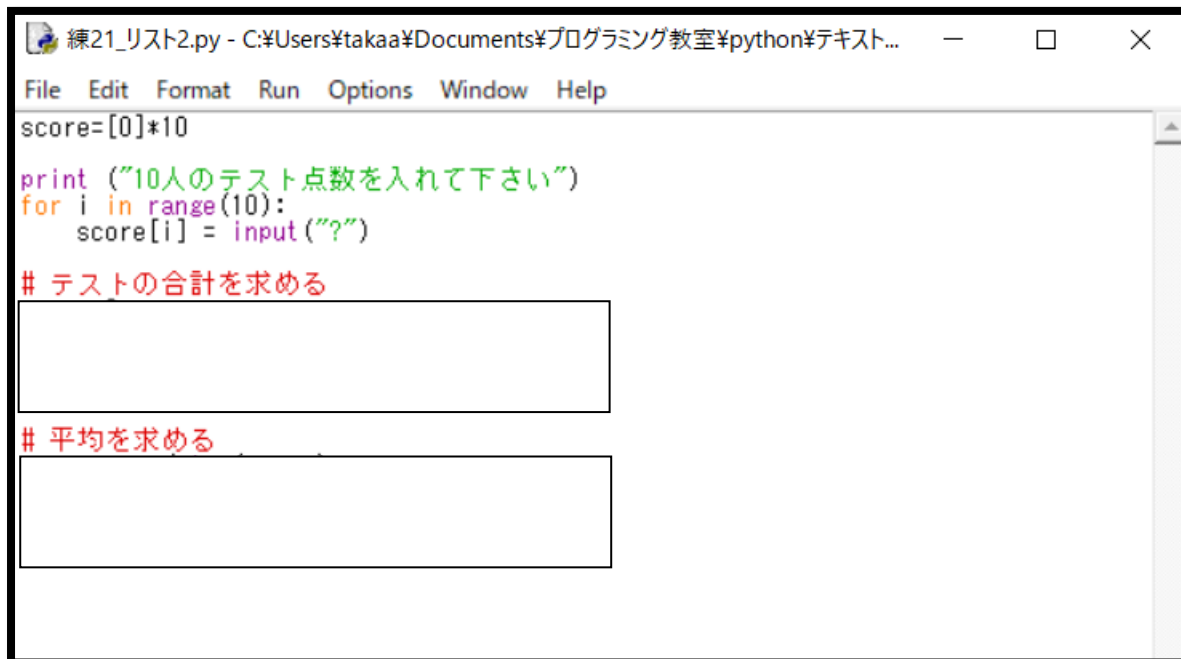
```

Python 3.5.2 Shell
File Edit Shell Debug Options Window Help
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習
練20_リスト1.py ==
88 点を足して合計は、 88
76 点を足して合計は、 164
67 点を足して合計は、 231
43 点を足して合計は、 274
79 点を足して合計は、 353
80 点を足して合計は、 433
91 点を足して合計は、 524
平均点は、 74.85714285714286 点です
>>>

```

## ■練習 21 リストを使おう 2

次はリストに自分で値を入れます。for を使って、リスト score に 10 回値をインプットします。



```

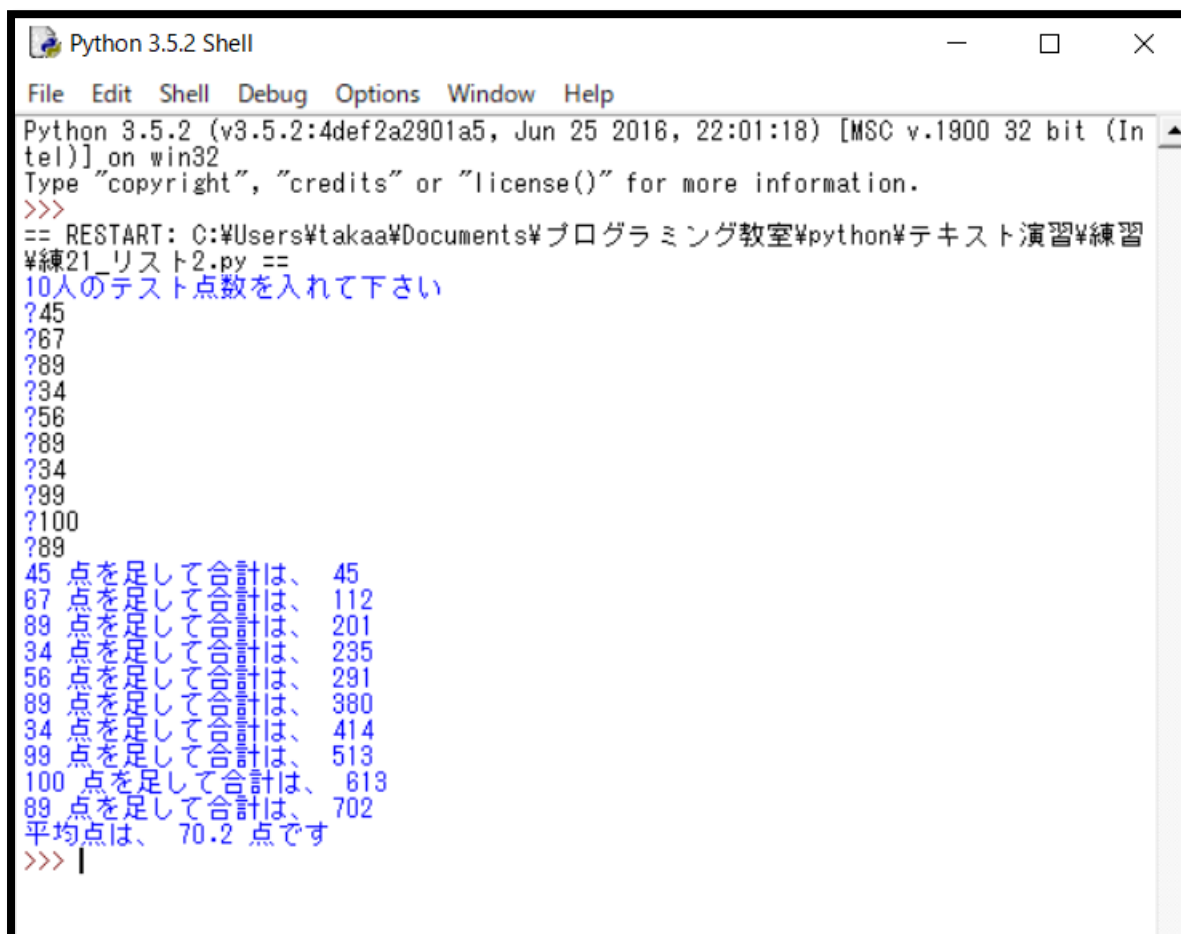
score=[0]*10

print ("10人のテスト点数を入れて下さい")
for i in range(10):
    score[i] = input("?")

# テストの合計を求める

# 平均を求める

```



```

Python 3.5.2 Shell

File Edit Shell Debug Options Window Help

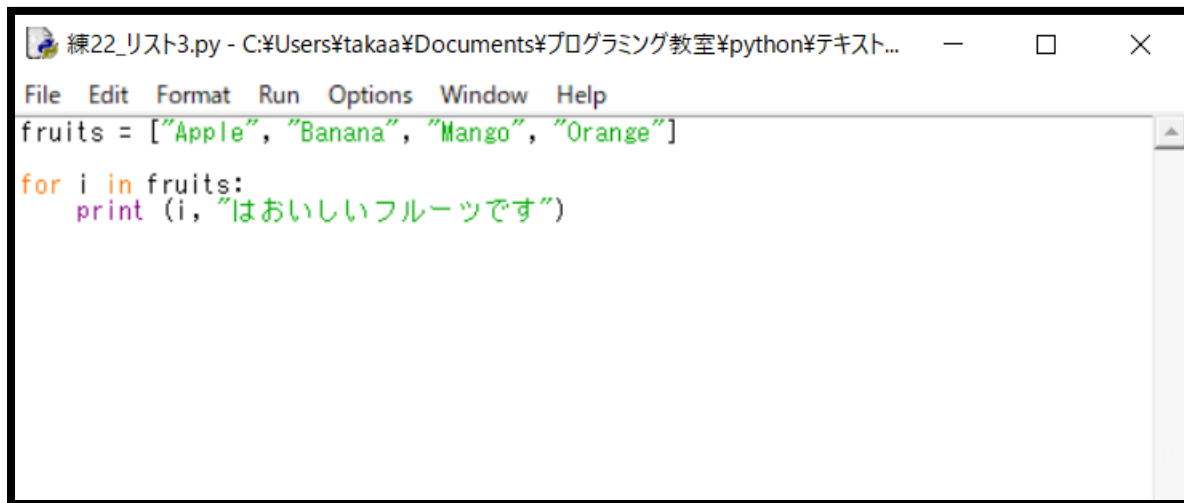
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
== RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習
練21_リスト2.py ==
10人のテスト点数を入れて下さい
?45
?67
?89
?34
?56
?89
?34
?99
?100
?89
45 点を足して合計は、 45
67 点を足して合計は、 112
89 点を足して合計は、 201
34 点を足して合計は、 235
56 点を足して合計は、 291
89 点を足して合計は、 380
34 点を足して合計は、 414
99 点を足して合計は、 513
100 点を足して合計は、 613
89 点を足して合計は、 702
平均点は、 70.2 点です
>>>

```



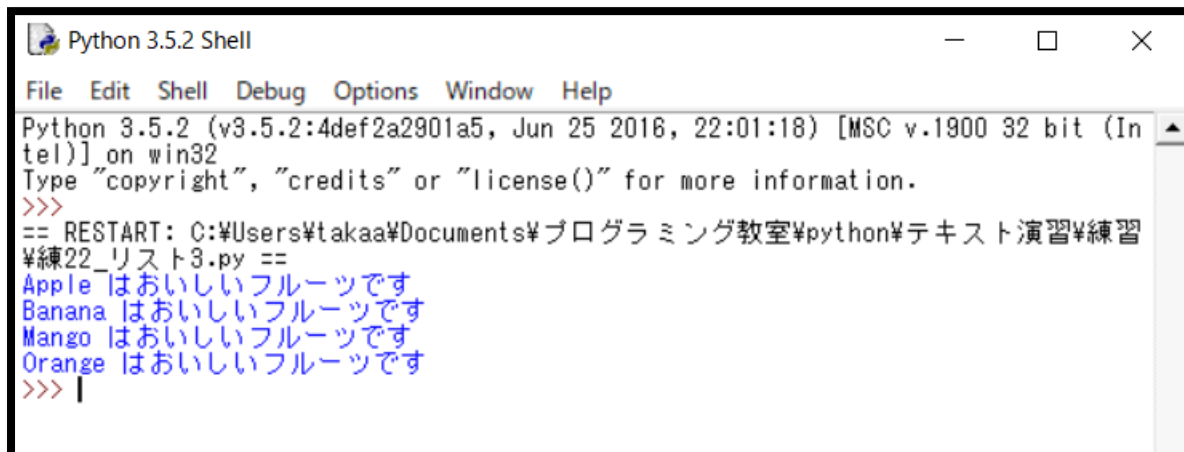
## ■練習 22 リストを使おう 3

リストには文字列を入れることもできます。



The screenshot shows a text editor window titled "練22\_リスト3.py - C:\Users\takaa\Documents\プログラミング教室\python\テキスト...". The menu bar includes File, Edit, Format, Run, Options, Window, and Help. The code in the editor is as follows:

```
fruits = ["Apple", "Banana", "Mango", "Orange"]  
for i in fruits:  
    print(i, "はおいしいフルーツです")
```



The screenshot shows a Python 3.5.2 Shell window titled "Python 3.5.2 Shell". The menu bar includes File, Edit, Shell, Debug, Options, Window, and Help. The output of the script is as follows:

```
Python 3.5.2 (v3.5.2:4def2a2901a5, Jun 25 2016, 22:01:18) [MSC v.1900 32 bit (Intel)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>>  
== RESTART: C:\Users\takaa\Documents\プログラミング教室\python\テキスト演習\練習  
練22_リスト3.py ==  
Apple はおいしいフルーツです  
Banana はおいしいフルーツです  
Mango はおいしいフルーツです  
Orange はおいしいフルーツです  
>>> |
```

□考えよう 12 赤点チェック

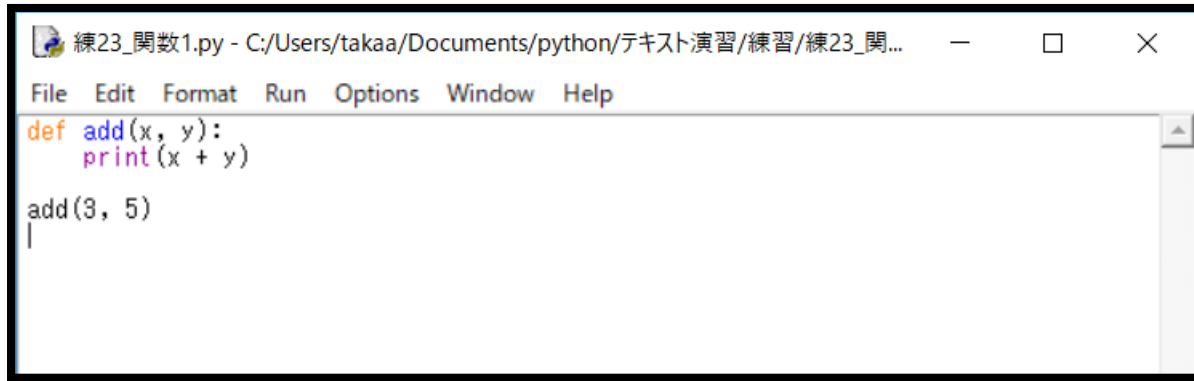
クラス 10 人のテスト点数を入力して、赤点（30 点以下）の人数と一番高得点の点数を表示するプログラムを作りましょう。

□考えよう 13 あいうえお作文

リストを使って、ランダムに組み合わせてできるあいうえお作文を作ろう。

## ■練習 23 関数を使おう 1

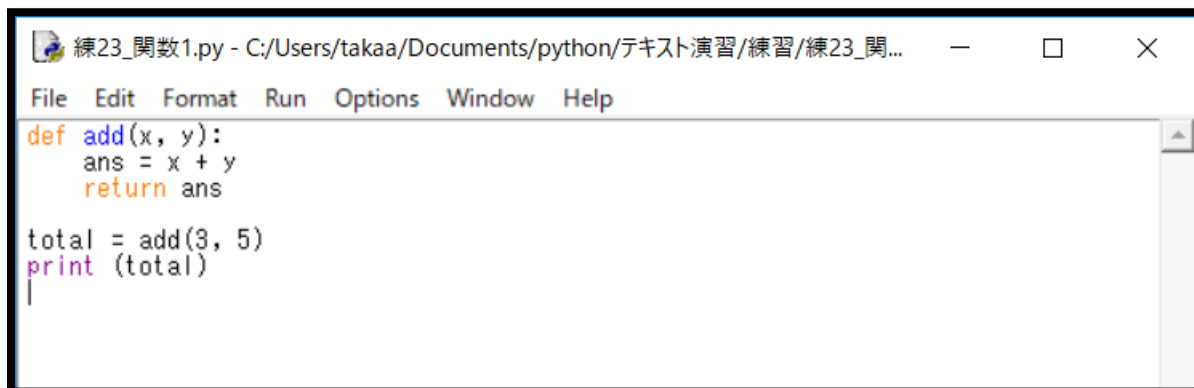
def 文を使って関数を定義することができます。下記の例では、x と y という二つの引数を受け取り、その合計値を表示する関数 add() を定義し、それを呼び出しています。以下のプログラムを実行して、結果を確認してみましょう。



```
練23_関数1.py - C:/Users/takaa/Documents/python/テキスト演習/練習/練23_関...
File Edit Format Run Options Window Help
def add(x, y):
    print(x + y)
add(3, 5)
|
```

return 文 は関数の戻り値を指定します。以下のプログラムを実行して、結果を確認してみましょう。

※変数 ans は、関数 add() の中でしか使えないローカル変数と言います。ローカル変数は return 文でメイン関数に戻すことができます。

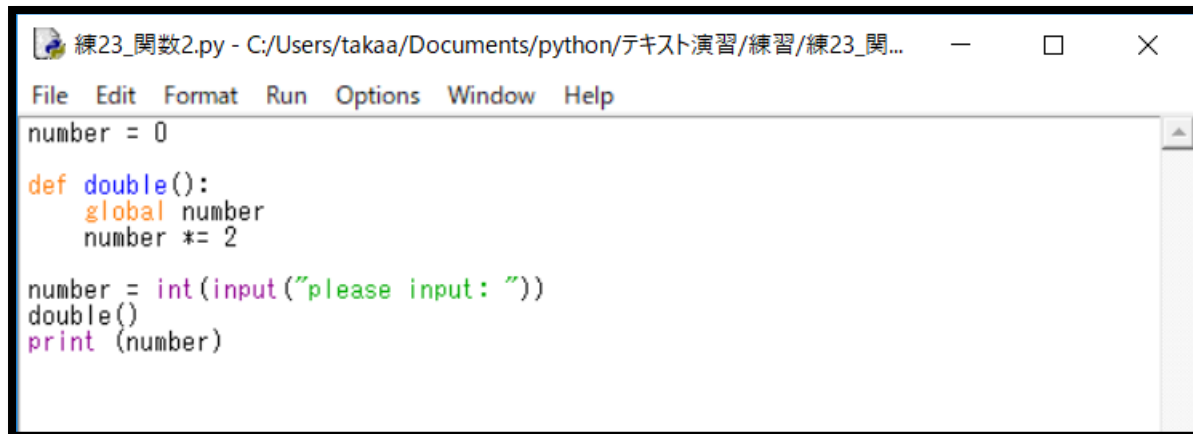


```
練23_関数1.py - C:/Users/takaa/Documents/python/テキスト演習/練習/練23_関...
File Edit Format Run Options Window Help
def add(x, y):
    ans = x + y
    return ans
total = add(3, 5)
print (total)
|
```

## ■練習 24 関数を使おう 2

関数の外部で定義された変数は グローバル変数 として扱われます。関数の中でグローバル変数を参照することはできますが、代入することはできません。代入する場合は、`global` で宣言する必要があります。

以下のプログラムで登場する関数 `double()` は、グローバル変数 `number` を 2 倍にするプログラムです。メイン関数から、入力した値を変数 `number` にセットします。関数 `double()` を呼び出して、変数 `number` がどう変わるかを見てみましょう。



```
練23_関数2.py - C:/Users/takaa/Documents/python/テキスト演習/練習/練23_関...
File Edit Format Run Options Window Help
number = 0
def double():
    global number
    number *= 2
number = int(input("please input: "))
double()
print (number)
```

## □考えよう 14 動物クイズ

1つのプログラムをまとめた関数を作り、メインから呼び出すことで、プログラムをシンプルにできます。関数とグローバル変数を使って、動物クイズを作りましょう。以下のプログラムは、ヒントです。少なくとも下記の機能を付けたプログラムを完成させましょう。

- ・ 5 問以上の問題を作りましょう。
- ・ 入力した答えから当たり、ハズレを表示します。
- ・ 最後にスコアを表示します。

クイズの出典：<https://www.city.tokushima.tokushima.jp/smph/zoo/joho/quiz/index.html>

```
def check(input, kotae):
    global score
    if input == kotae:
        print("当たり")
        score = score + 1

score = 0

print("動物に関するクイズに答えてね。半角の数字で入力してね。")
print("人間の歯は 32 本ですが、ゾウの歯は何本でしょうか？¥n(1) 4 本 (2) 8 本 (3) 12 本 (4) 16 本 (5) 20 本")
input1 = input("")
check(int(input1), 1)
print("次の鳥の中で、実際にいる鳥はどれでしょうか？¥n(1) ムシバ (2) サシバ (3) キンバ (4) ギンバ (5) イレバ")
input2 = input("")
check(int(input2), 2)
```

□考えよう 15 西暦・平成対応表

西暦年と平成年の対応表を作ろう。1989年(平成元年)から2019年までをループで出力し、ループ内で、各西暦年を平成年に変換して両方を表示させよう。

□考えよう 16 誕生日アプリ

生まれた西暦年を入力すると和暦(明治・大正・昭和・平成・令和)と年齢を表示するアプリを製作します。

□考えよう 16 いえべん管理アプリ

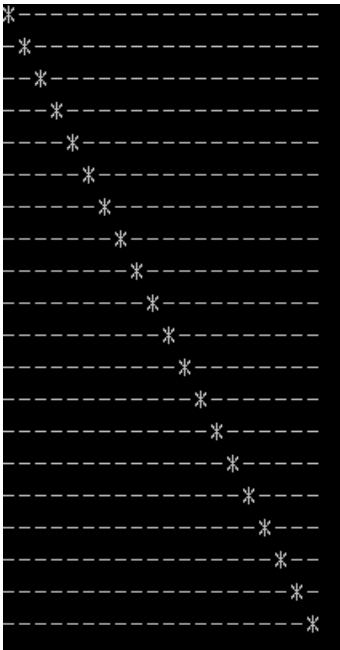
月曜日～日曜日の5教科の勉強時間を入力して、管理するアプリを作りましょう。教科ごとの時間やトータル時間が分かる機能を付けます。入力した時間はテキストファイルを使って管理します。

□考えよう 17 体重管理アプリ

身長と体重を入力して保存するアプリを作ります。実行するとテキストファイルから過去のデータを読み込みます。今日の体重を入力すると、BMI 指数と前回から増減とコメントを表示してくれる機能を付けます。

Python で自由テーマに挑戦：

① for 文を使って作る芸術作品



Python で自由テーマに挑戦：

② スクラッチで作った映画館のレジプログラムを Python で書く

③ 誕生日占い

2018 年+9 月 8 日

↓

$2+0+1+7+9+8=28$

$2+8=10$

$1+0=1$

最終的に出てきた数字が、その年のテーマとなります。

1. スタート、種まき
2. 出会い、人脈づくり
3. 花開く、楽しみ切る
4. 基礎固め、家関係
5. 転機・チャレンジ
6. 自己犠牲、人に尽くす
7. 自己投資、勉強
8. あるがまま、自然体
9. けじめ、集大成、断捨離



## ■ちょっと番外編 tkinter でゲーム作り 1 サンプル①

```
import tkinter

# ウィンドウ作成
root = tkinter.Tk()
root.title("_____")
root.minsize(640, 480)
root.option_add("*font", ["MS P ゴシック", 22])

# 画像表示
canvas = tkinter.Canvas(bg="black", width=640, height=480)
canvas.place(x=0, y=0)
img = tkinter.PhotoImage(file="img/back1.png")
canvas.create_image(320, 240, image=img)

# テキスト表示
question = tkinter.Label(text="_____", bg="white")
question.place(x=100, y=40)

# テキストボックス表示
entry = tkinter.Entry(width=12, bd=4)
entry.place(x=50, y=133)

# 質問ボタン表示
askbutton = tkinter.Button(text="_____",)
askbutton.place(x=260, y=125)

# 答え表示
answer = tkinter.Label(text=".....", bg="white")
answer.place(x=115, y=235)

# イベント設定
def ask_click():
    val = entry.get()
    minutes = float(val)
    hours = round(minutes/60, 2)
    answer["text"] = str(hours) + "時間だね！"

askbutton["command"] = ask_click

# メインループ
root.mainloop()
```

完成イメージ：



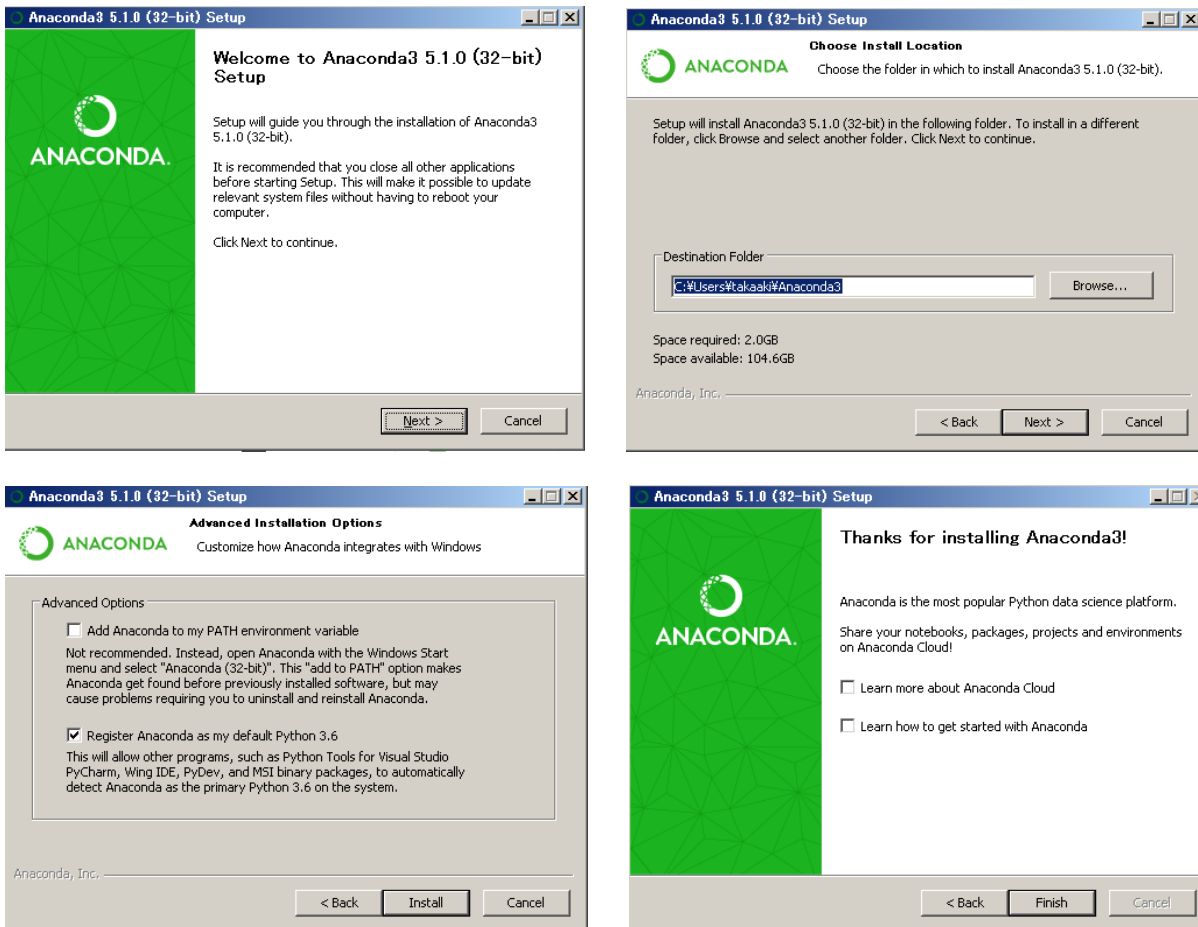
■ちょっと番外編 tkinter でゲーム作り 1 サンプル②  
 サンプル①を応用して、オリジナルのプログラムを作ります。

完成イメージ：



## ■準備 2 Anaconda(アナコンダ)とPygame をインストールしよう

## ①Anaconda のインストール



## ②Pygame のインストール

```
(base) C:\>pip install pygame
```

```
Collecting pygame
```

```
Cache entry deserialization failed, entry ignored
```

```
Downloading pygame-1.9.3-cp36-cp36m-win_amd64.whl (4.2MB)
```

```
100% |????????????????????????????????????| 4.2MB 159kB/s
```

```
Installing collected packages: pygame
```

```
Successfully installed pygame-1.9.3
```

```
(base) C:\>python
```

```
Python 3.6.3 |Anaconda custom (64-bit)| (default, Oct 15 2017, 03:27:45) [MSC v.1900 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
```

```
>>> import pygame
```

```
>>>
```

## ■練習 23 直線を書く (Pygame 編)

Pygame はゲームを作るためのモジュール集です。まずは Pygame のモジュールを読み込み、pygame.draw.line を使って、直線を描いてみましょう。画面の色は黒色(R=0, G=0, B=0)を指定します。

```
# -*- coding:utf-8 -*-
import pygame
from pygame.locals import *
import sys

def main():
    pygame.init()                                # Pygame の初期化
    screen = pygame.display.set_mode((300, 200)) # 大きさ 300*200 の画面を生成
    pygame.display.set_caption("TEST")          # タイトルバーに表示する文字

    while (1):
        screen.fill((0, 0, 0))                  # 画面を黒色に塗りつぶし
        pygame.draw.line(screen, (0, 255, 0), (0, 0), (80, 80), 5) # 直線の描画
        pygame.display.update()                 # 画面を更新
        # イベント処理
        for event in pygame.event.get():
            if event.type == QUIT:               # 閉じるボタンが押されたら終了
                pygame.quit()                   # Pygame の終了(画面閉じられる)
                sys.exit()

if __name__ == "__main__":
    main()
```

練習問題①

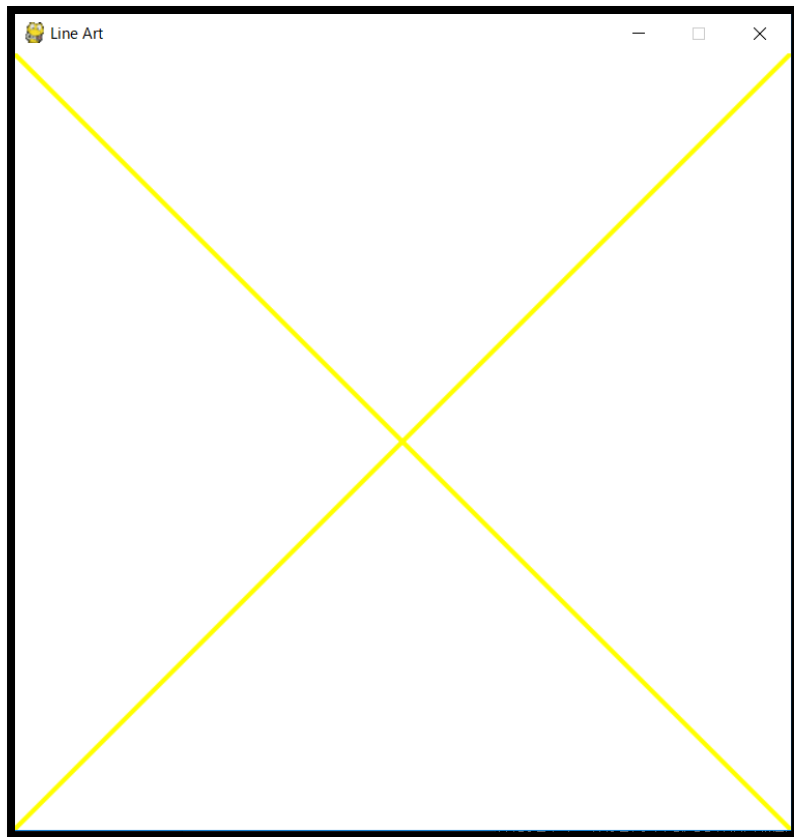
タイトルバーを” LINE Art” に変えてみよう。

練習問題②

画面の大きさを 600\*600、背景を白色に変更しよう。

練習問題③

下図のように画面全体に直線をクロスさせて描こう。色は黄色にしよう。



**■練習 24 四角形を書く (Pygame 編)**

次は、`pygame.draw.rect` を使って、四角形を描いてみましょう。

```
pygame.draw.rect(screen, (0, 80, 0), Rect(10, 10, 80, 50), 5)
```

※左上座標(10, 10)、幅 80px、高さ 50px の長方形を線幅 5px の緑色(R=0, G=80, B=0)で描く。

**練習問題①**

幅 400px、高さ 400px の正方形を画面の中央に配置されるように描いてみよう。

**■練習 25 円を書く (Pygame 編)**

次は、`pygame.draw.circle` を使って、円を描いてみましょう。

```
pygame.draw.circle(screen, (0, 80, 0), (300, 300), 100), 5)
```

※中心座標(300, 300)、半径 100px の円を線幅 5px の緑色(R=0, G=80, B=0)で描く。

**練習問題①**

画面の中央に二重丸を描いてみよう。

## ■練習 26 矢印キーで物体を操作する (Pygame 編)

Pygame のモジュールを読み込み、pygame.event.get を使って、矢印キーで物体を動かしてみましょう。

```
# -*- coding: utf-8 -*-

import pygame

from pygame.locals import *

import sys

def main():
    (w,h) = (400,400)                # 画面サイズ
    (x,y) = (int(w/2), int(h/2))     # 円の中心座標を画面サイズの中心に設定する
    pygame.init()                    # pygame 初期化
    pygame.display.set_mode((w, h), 0, 32)  # 画面設定
    screen = pygame.display.get_surface()  # 画面の情報を変数 screen にセット

    while (1):
        pygame.display.update()       # 画面を更新する
        pygame.time.wait(30)          # 更新時間間隔を設定する
        screen.fill((0, 20, 0, 0))    # 画面の背景色を黒に
        # 円の中心座標 x, y が画面の範囲外にある場合に x, y に画面端の座標をセットする
        if x < 0:
            x = 0
        if x > w:
            x = w
        if y < 0:
            y = 0
        if y > h:
            y = h
        # 円を描画
        pygame.draw.circle(screen, (0, 200, 0), (x, y), 5)
        # イベント処理
        for event in pygame.event.get():
            # 画面の閉じるボタンを押したとき
            if event.type == QUIT:
                pygame.quit()
                sys.exit()
            # キーを押したとき
            if event.type == KEYDOWN:
                # ESC キーなら終了
                if event.key == K_ESCAPE:
                    pygame.quit()
```

```
        sys.exit()

# 矢印キーなら円の中心座標を矢印の方向に移動
if event.key == K_LEFT:
    x -= 1
if event.key == K_RIGHT:
    x += 1
if event.key == K_UP:
    y -= 1
if event.key == K_DOWN:
    y += 1

if __name__ == "__main__":
    main()
```

#### 練習問題①

円を配置する座標を画面の中心から別のところに変更しよう。

#### 練習問題②

矢印キーで円を動かすスピードを速くしてみよう。



## ■練習 27 矢印キー長押しで物体を操作する (Pygame 編)

Pygame のモジュールを読み込み、pygame.event.get を使って、矢印キー長押しで物体を動かしてみましょう。

```
# -*- coding: utf-8 -*-

import pygame

from pygame.locals import *

import sys

def main():
    (w,h) = (400,400)                # 画面サイズ
    (x,y) = (int(w/2), int(h/2))     # 円の中心座標を画面サイズの中心に設定する
    pygame.init()                    # pygame 初期化
    pygame.display.set_mode((w, h), 0, 32)  # 画面設定
    screen = pygame.display.get_surface()  # 画面の情報を変数 screen にセット

    while (1):
        # キーイベント処理(キャラクタ画像の移動)
        pressed_key = pygame.key.get_pressed()
        if pressed_key[K_LEFT]:
            x-=1
        if pressed_key[K_RIGHT]:
            x+=1
        if pressed_key[K_UP]:
            y-=1
        if pressed_key[K_DOWN]:
            y+=1

        pygame.display.update()       # 画面を更新する
        pygame.time.wait(30)          # 更新時間間隔を設定する
        screen.fill((0, 20, 0, 0))    # 画面の背景色を黒に
        # 円を描画
        pygame.draw.circle(screen, (0, 200, 0), (x, y), 5)
        # イベント処理
        for event in pygame.event.get():
            # 画面の閉じるボタンを押したとき
            if event.type == QUIT:
                pygame.quit()
                sys.exit()
            # キーを押したとき
            if event.type == KEYDOWN:
                # ESC キーなら終了
```

```
if event.key == K_ESCAPE:  
    pygame.quit()  
    sys.exit()
```

```
if __name__ == "__main__":  
    main()
```

#### 練習問題①

円の色や大きさを変更してみよう。