Python で遊ぶ

おみくじ

```
import random

# おみくじの結果リスト
fortunes = ['大吉', '吉', '凶', '大凶']

# ランダムにおみくじの結果を選ぶ
fortune = random.choice(fortunes)

# 結果を表示する
print(fortune)
```

変数

僕たちが使っている数字とか、文字は定数（じょうすう、ていすう）と呼ばれます。
それに対して、名前だけを用意して、数字とか、文字を入れられるものを変数と呼びます。

次の命令の左側を変数といいます

```
fortune =  [ ・・・]
```

さいころをふる

```
import random

saikoro = random.randint( 1, 6)

print ("saikoro:" , saikoro)
```

解説

ゲームでは、でたらめな数（＝ 乱数）が欲しいときがあります。

AからBの間の整数（＝int）ででたらめな数が欲しいときは、次のように書きます

```
変数 = random.randint( A, B)
```

数当てゲームをつくる

1から100までの数当てゲームを作ります

- でたらめな数を１から１００で作り、変数の中に入れる
- メッセージを出して「数字を入れてください」と聞く
- 繰り返しの始まり
  - 数字を入力し、1より小さい、もしくは、100より大きいときは、メッセージ「1から100までの数字を入れてください」と表示する
  - 入力した数字が小さいときには、「もっと大きい数字です」と表示する
  - 入力した数字が大きいときには、「もっと小さい数字です」と表示する

コード

さあ、プログラムを打ってみましょう。

「#」から始まる行は打たなくてもかまいません。

```
import random

def guess_number():
    global secret_number
    # ランダムな数字を生成
    secret_number = random.randint(1, 100)

    print("1から100までの数字を当ててください！")

    # ユーザーが当てるまでループ
    while True:
        try:
            # 文字「あなたの予想：」を表示して数字を入れる
            # 入力された数字は、guessに格納される
            guess = int(input("あなたの予想: "))
            if guess < 1 or guess > 100:
                print("1から100までの数字を入力してください。")
                # continue命令は、残りの部分を飛ばします。
                continue
            if guess < secret_number:
                print("もっと大きい数字です！")
            elif guess > secret_number:
                print("もっと小さい数字です！")
            else:
                print("正解です！")
                # break命令は繰返しをやめる命令です。
                break
        except ValueError:
            print("有効な数字を入力してください。")

# ゲームを実行
guess_number()
```

Step up

得点をつけましょう。

得点は、変数Scoreとしましょう。

最初は、100点です。 これは、最初に書きます。

```
score = 100
```

1回間違えると、得点は半分にしましょう。　これは間違えた時に書きます。
今回は2か所、大きい場合と、小さい場合ですね

```
score = score / 2
```

得点を表示するには、次のように書きます。

```
print( "得点", score )
```

3つのパーツをプログラムの中に付け加えます。
この時、インデント（字の始まる位置に注意してください）

ボールを反射するプログラムをつくる

今回は、プログラムの書き方を見ていきます。
インデントが大事ということを見てもらいたいと思います。

```
from tkinter import *

ball = {
 “dirx” : 15,
 “diry” : -15,
 “x” : 350,
 “y” : 300,
 “w” : 10,
}

win = Tk()
cv = Canvas(win, width=600, height=400)
cv.pack()

def draw_objects():
cv.delete( ‘all’ )
cv.create_oval(
ball[ “x” ] - ball[ “w” ], ball[ “y” ] - ball[ “w” ],
ball[ “x” ] + ball[ “w” ], ball[ “y” ] + ball[ “w” ],
fill=” green” )

def move_ball():
bx = ball[ “x” ] + ball[ “dirx” ] by = ball[ “y” ] + ball[ “diry” ] if bx < 0 or bx > 600:
ball[ “dirx” ] *= -1
if by < 0 or by > 400:
```

```
ball[ “diry” ] *= -1
if 0 <= bx <= 600:
ball[ “x” ] = bx
if 0 <= by <= 400:
ball[ “y” ] = by

def game_loop():
draw_objects()
move_ball()
win.after(50, game_loop)

game_loop()
win.mainloop()
```

修正後のコードです

Pythonは汚い書き方をさせないプログラム言語です。
きれいに書いていくと動きますという例なので、真似をして修正してください。

```
from tkinter import *

ball = {
    "dirx": 15,
    "diry": -15,
    "x": 350,
    "y": 300,
    "w": 10,
}

win = Tk()
cv = Canvas(win, width=600, height=400)
cv.pack()

def draw_objects():
    cv.delete('all')
    cv.create_oval(
    ball["x"] - ball["w"], ball["y"] - ball["w"],
    ball["x"] + ball["w"], ball["y"] + ball["w"],
    fill="green")

def move_ball():
    bx = ball["x"] + ball["dirx"]
    by = ball["y"] + ball["diry"]
    if bx < 0 or bx > 600:
        ball["dirx"] *= -1
    if by < 0 or by > 400:
        ball["diry"] *= -1
    if 0 <= bx <= 600:
        ball["x"] = bx
    if 0 <= by <= 400:
        ball["y"] = by
```

```
def game_loop():
    draw_objects()
    move_ball()
    win.after(50, game_loop)

game_loop()
win.mainloop()
```

改造です。

ボールを滑らかに動かすには、移動量を少なくします。
xとyがボールの位置です。この変化を小さくします。

```
    bx = ball["x"] + ball["dirx"] /3
    by = ball["y"] + ball["diry"] /3
```

改造２

スピードアップをします。

win.after()の最初の数は画面を書き換える間隔です。
この値を小さくすると、早く書き換わります。

ちなみに残像が見えるのは、Pythonの画面の書き換えが遅いので、限界です。

```
    win.after(10, game_loop)
```

改造３

キーボードからの操作で動きを変える
次のコードをgame_loop()の前に追加します。

上矢印キーを押すたびに、移動の向きを反転させるプログラムです。

```
def keyevent( e):
    key = e.keysym
    if key == "Up":
        ball["diry"] *= -1

win.bind( "<KeyPress>", keyevent)
```

改造４

4方向の矢印の動きを改造してみましょう