

# プログラミング演習

解答例

# 本日のお題

1. スロットマシン
  - 3つの数を生成
2. ジャンケンの勝敗
  - 簡単に判定
3. 確実に数値を入力させるプログラム
  - 違ったら再入力
4. コンピュータが当てる、数当てプログラム
  - 数を当てる手順

# 1. スロットマシン1

- 0~999の乱数を発生し変数Xに代入
- 100の位を変数x1、10の位をx2、1の位をx3に代入
- X と x1,x2,x3を表示

- 例 X=0 なら 0 0 0  
X=234 なら 2 3 4

実行例1

0  
0 0 0

実行例2

235  
2 3 5

# 1.スロットマシン1

- 計算によって1桁ずつとりだす
  - //(商) と %(余り)を使う

```
import random
```

```
X=random.randint(0, 999)
```

```
x1=X//100
```

```
x2=
```

```
x3=X%10
```

ここを  
考える

- 0~999の乱数を発生し変数Xに代入
- 100の位を変数x1、10の位をx2、1の位をx3に代入
- x1,x2,x3を表示

疑似言語(共通テスト用プログラム表記)

- (1)  $X=0\sim 99$ までの乱数
- (2)  $x1=X$ を100で割った商
- (3)  $x2=(X$ を10で割った商) を10で割った余り
- (4)  $x3=X$ を10で割った余り
- (5)  $X$ を表示
- (6)  $x1, x2, x3$ を表示

# 1.スロットマシン1

- 計算によって1桁ずつとりだす
  - //(商) と %(余り)を使う

```
import random
```

```
X=random.randint(0, 999)
```

```
x1=X//100          #100で割った商
```

```
x2=(X//10)%10     #10で割った商を10で割った余り
```

```
x3=X%10           #10で割った余り
```

```
print(X)
```

```
print(x1, x2, x3)
```

# 2.ジャンケンの勝敗

```
import random
A=int(input('0:グー 1:チョキ 2:パーを入力'))
B=random.randint(0,2)
hantei=['あいこ','勝ち','負け']
te=['グー','チョキ','パー']
```

- 0:グー 1:チョキ 2:パー とする
- 勝敗を判定する式を考える

この続きをつくる

1行でできるよ!

自分+相手		相手		
		0:グー	1:チョキ	2:パー
自分	0:グー	0	1	2
	1:チョキ	1	2	3
	2:パー	2	3	4

相手-自分		相手		
		0:グー	1:チョキ	2:パー
自分	0:グー	0	1	2
	1:チョキ	-1	0	1
	2:パー	-2	-1	0

自分-相手		相手		
		0:グー	1:チョキ	2:パー
自分	0:グー	0	-1	-2
	1:チョキ	1	0	-1
	2:パー	2	1	0

## 2.ジャンケン の勝敗

```
import random
A=int(input('0:グー 1:チョキ 2:パーを入力'))
B=random.randint(0,2)
hantei=['あいこ','勝ち','負け']
te=['グー','チョキ','パー']
```

- 0:グー 1:チョキ 2:パー とする
- 勝敗を判定する式を考える
- -1と2が負け、-2と1が勝ち、0があいこ
  - %を使うとうまくいく

この続きをつくる

相手-自分		相手		
		0:グー	1:チョキ	2:パー
自分	0:グー	0	1	2
	1:チョキ	-1	0	1
	2:パー	-2	-1	0

相手-自分+3		相手		
		0:グー	1:チョキ	2:パー
自分	0:グー	3	4	5
	1:チョキ	2	3	4
	2:パー	1	2	3

1行でできるよ!

## 2.ジャンケン の勝敗

- 0:グー 1:チョキ 2:パー とす
- 勝敗を判定する式を考える

		(相手-自分+3)%3		
		相手		
		0:グー	1:チョキ	2:パー
自分	0:グー	0	1	2
	1:チョキ	2	0	1
	2:パー	1	2	0

```
import random
A=int(input('0:グー 1:チョキ 2:パーを入力'))
B=random.randint(0,2)
hantei=['あいこ','勝ち','負け']
te=['グー','チョキ','パー']
print(hantei[(B-A+3)%3])
```

3足してから  
3で割った余り



# 3. 確実に数値を入力させるプログラム

- 数値を入力させたくても文字を入力する人がいる
- それでもエラーを出さずに動かせるプログラムを書く

str.isdigit(文字列)関数  
文字列が数値ならTrue,  
そうでなければFalse

## • 動作イメージ

2桁の数を入力してください

数値ではありません もう一度入力してください

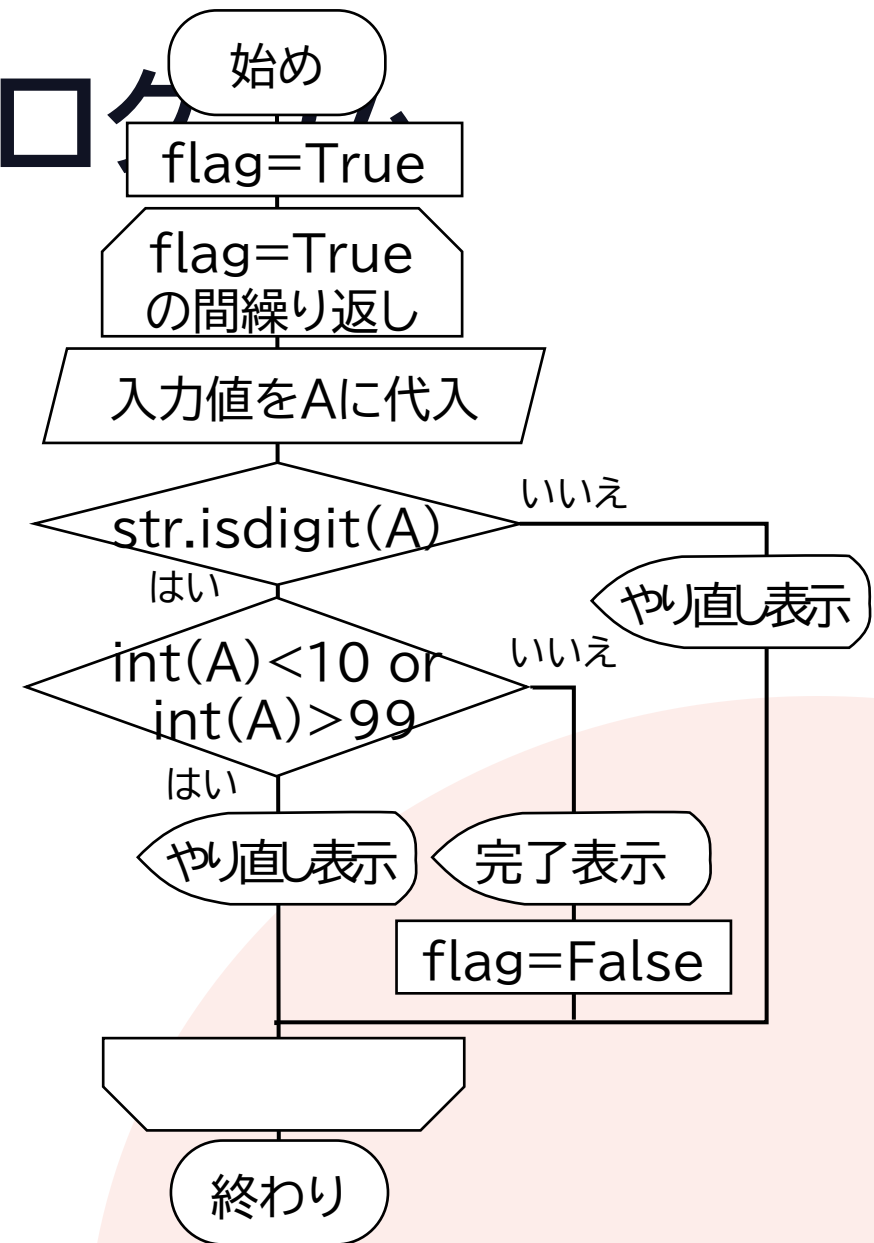
5 ですね 2桁ではありません もう一度入力してください

32 ですね 入力ありがとうございます

# 3. 確実に数値を入力させるプログラム

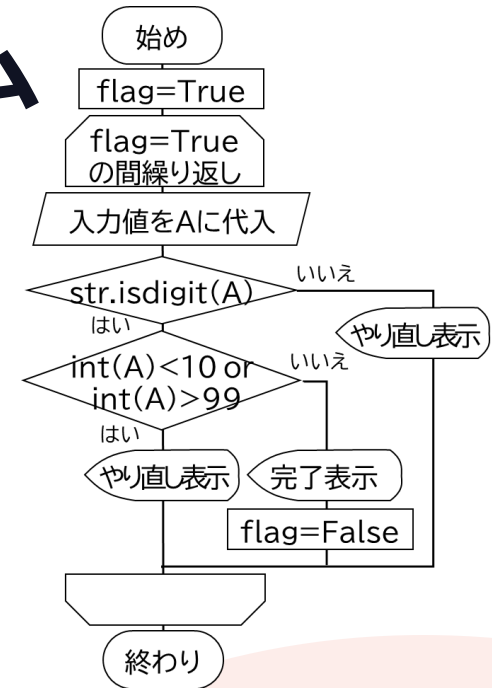
- 数値か確かめる関数
  - `str.isdigit(文字列)`関数  
文字列が数値ならTrue,  
そうでなければFalse

初めて使う関数は動作確認するとよい  
#動作確認のプログラム  
`A=input()`  
`print(str.isdigit(A))`



# 3. 確実に数値を入力させるプログラム

```
flag=True
while flag:
    A=input('2桁の数を入力してください')
    if str.isdigit(A):
        if int(A)<10 or int(A)>99:
            print(A,'ですね 2桁ではありません もう一度入力してください')
        else:
            print(A,'ですね入力ありがとうございます')
            flag=False
    else:
        print('数値ではありません もう一度入力してください')
```



# 4.コンピュータが当てる、数当てプログラム

- 利用者が2桁の数を1つ決める
- 質問をしてこの数を当てるプログラムを書く

- 動作イメージ

あなたが考えた数は20ですね

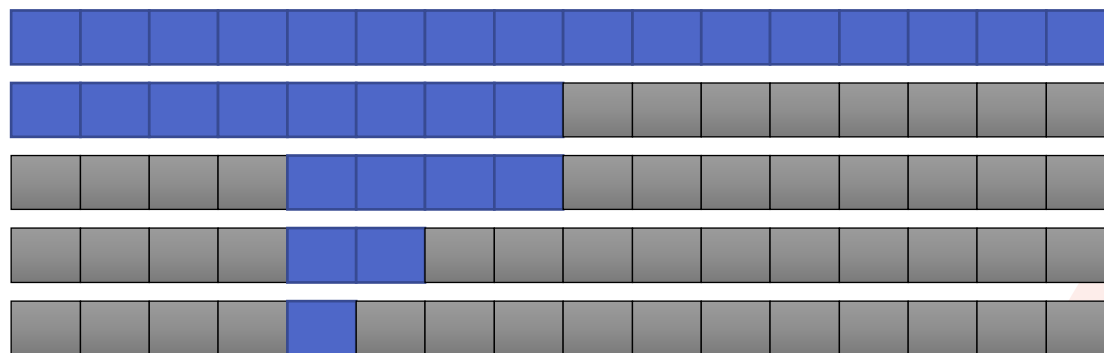
0:はい 1:大きい 2:小さい で回答してください

では、あなたが考えた数は60ですね

0:はい 1:大きい 2:小さい で回答してください

# 4.コンピュータが当てる、数当てプログラム

- 確実に当てる仕組みを考える
- 質問する数を合理的に決定する
  - 7回聞けば当てられる
  - 範囲を半分に、半分に、 $\dots$ として絞り込む



# 4.コンピュータが当てる、数当てプログラム

```
max=99
min=10
atari=False
cnt=0
while atari==False:
    Qnum=int((max+min)/2)
    print('あなたが考えた数は', Qnum, 'ですね')
    Ans=int(input('0:はい 1:大きい 2:小さい で回答してください'))
    cnt=cnt+1
    if Ans==0:
        atari=True
    elif Ans==1:
        min=Qnum
    else:
        max=Qnum
print(Qnum, 'でした', cnt, '回で当たりました')
```