

## 多重ループ

これまでの繰り返し処理は 1 つだけでしたが、  
繰り返し処理にも入れ子構造があります。  
繰り返し処理が 1 つしかないものを 1 重ループ、  
2 つあるものを 2 重ループといい、  
2 つ以上あるものを多重ループといいます。

## 多重ループの使い方①

```
for 変数 in 繰り返しの条件:
    for 変数 in 繰り返しの条件:
        処理
```

## 多重ループの使い方③

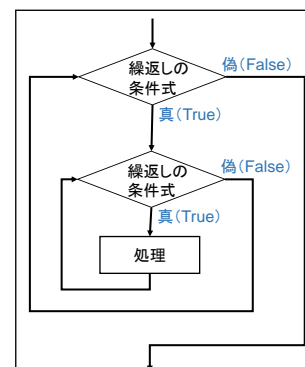
```
for 変数 in 繰り返しの条件:
    while 条件式:
        処理
```

## 多重ループの使い方②

```
while 条件式:
    while 条件式:
        処理
```

## 多重ループの使い方④

```
while 条件式:
    for 変数 in 繰り返しの条件:
        処理
```

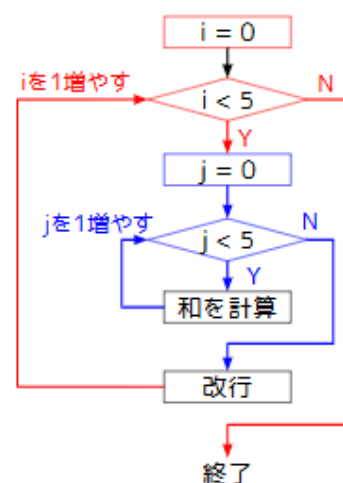


(例 1) 次のプログラムを入力して実行してみよう。

```
1 for i in range(5):
2     for j in range(5):
3         print((i+1)+(j+1), end=" ")
4     print()
```

※これまでの print とは違い、工夫をしています。

print()内の末尾にある「end=" "」ですが、通常 python のプログラムは改行されてしまいます。そこで、「end=" "」で改行をしないようにし、代わりに半角スペースを 1 つ入れています。そして、4 行目の print()は、改行をするために入れています。



2 重ループですが、外側のループが縦方向に対応しており、内側のループが横方向に対応しています。

## 多重ループの使い方①

```
for 変数 in 繰り返しの条件:
    for 変数 in 繰り返しの条件:
        処理
```

(例 1) の  
実行結果

実行 結果		j				
		0	1	2	3	4
i	0	2	3	4	5	6
	1	3	4	5	6	7
	2	4	5	6	7	8
	3	5	6	7	8	9
	4	6	7	8	9	10

(問 1) (例 1) を参考に九九を出力するプログラムを書いてみましょう。

(参考) (例 1) のプログラムを while に書き換えると、次のようになる。入力し、実行してみよう。

```
1 i = 0
2 while (i < 5):
3     j = 0
4     while(j < 5):
5         print((i+1)+(j+1), end=" ")
6         j = j + 1
7     print()
8     i = i + 1
```

## 外側ループのカウンタを内側ループで使う

先ほどの 2 重ループで、内側ループの繰り返し回数に外側ループのカウンタを入れると、どうなるでしょうか。

(例 2) 次のプログラムを入力して実行してみよう。

```
1 for i in range(5):
2     for j in range(i):
3         print("○", end=" ")
4     print()
```

(例 3) 次のプログラムを入力して実行してみよう。

```
1 for i in range(5):
2     for j in range(5-i):
3         print("○", end=" ")
4     print()
```

(問 2) (例 2) と (例 3) を参考に次のような出力になるようにプログラムを書いてみましょう。

```
○ ○ ○ ○ ○
● ○ ○ ○ ○
● ● ○ ○ ○
● ● ● ○ ○
● ● ● ● ○
● ● ● ● ○
```

```
1 for i in range(5):
2     for ? :
3         print("?", end=" ")
4     for ? :
5         print("?", end=" ")
6     print()
```

もちろん、多重ループの中に条件式を入れることもできます。

例えば、対角線の場合に●、それ以外に○を出力するプログラムを見てみましょう。

(例 4) 次のプログラムを実行してみよう。

```
1 for i in range(5):
2     for j in range(5):
3         if(i == j):
4             print("●", end=" ")
5         else:
6             print("○", end=" ")
7     print()
```

**発展** (例 4) を参考にして、キーボードから入力された正の整数の大きさの市松模様を出力するプログラムを作ってみよう。

HINT : 各列・各行で、■と□がどのようなルールで配置されているのか考える。

例) 8 と入力した場合

```
■ □ ■ □ ■ □ ■ □
□ ■ □ ■ □ ■ □ ■
■ □ ■ □ ■ □ ■ □
□ ■ □ ■ □ ■ □ ■
■ □ ■ □ ■ □ ■ □
□ ■ □ ■ □ ■ □ ■
■ □ ■ □ ■ □ ■ □
□ ■ □ ■ □ ■ □ ■
```

例) 3 と入力した場合

```
■ □ ■
□ ■ □
■ □ ■
```

例) 6 と入力した場合

```
■ □ ■ □ ■ □
□ ■ □ ■ □ ■
■ □ ■ □ ■ □
□ ■ □ ■ □ ■
■ □ ■ □ ■ □
□ ■ □ ■ □ ■
```