Python によるプログラミング入門④繰り返し　その１

目次

【この単元でやりたいこと】

- 「名前の一覧」を表示するプログラムを作ります。
- これを実現する方法として、「繰り返し」の書き方を学びます。また、「リスト」という型についても学習します。

1. For 文の基本的な書き方

- For 文を書いてみよう

【課題1：入力してみよう】

```
for i in range(5):
    print(”Hello!”)
```

数字を変えると、どう表示されるだろうか？

【実行結果】

```
Hello!
Hello!
Hello!
Hello!
Hello!
```

- for 文の基本的な書き方

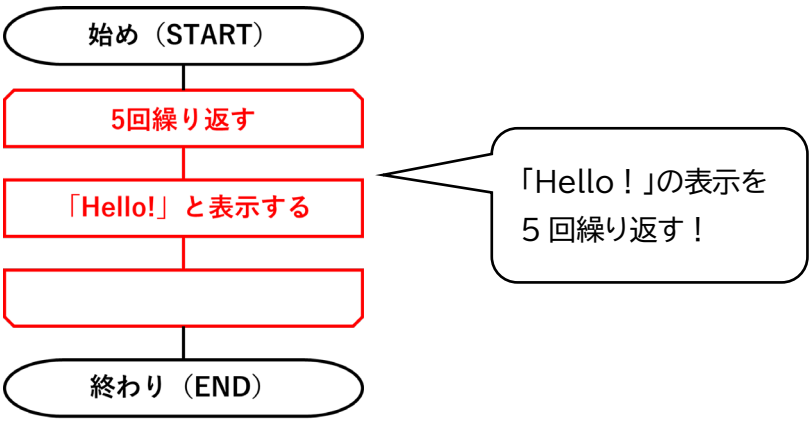
```
for カウンタ変数 in 繰り返し回数:
    繰り返したい処理
    ・・・・・・・
```

コロンを忘れずに！

「繰り返したい処理」の前には、インデントも忘れずに！

```
for i in range(5):      ← 5 回繰り返す:
    print(”Hello!”)     ← “Hello!” と表示する
```

【フローチャートとの比較】



- カウンタ変数とは？

【課題2：入力してみよう】

```
for i in range(5):
    print(i)
```

カウンタ変数を出力
何が表示された？繰返し回数も変えてみよう

【実行結果】

```
0
1
2
3
4
```

【フローチャートとの比較】

  - カウンタ変数の正体：回数やものを数えるときの「指」のようなもの。人間だと、数えるときは 1 から指折り数えるが、Python の場合は、0 から指折り数える。そのため、カウンタ変数は 0 からスタートする。

- range 関数について
  - range 関数の引数は、3 つまで入れることができる。それにより、カウンタ変数に代入される数が異なってくる

【課題 3-1：入力してみよう】

```
for i in range(0,8):
    print(i)
```

range(8)のときと
同じ結果！

【実行結果】

```
0
1
2
3
4
5
6
7
```

【課題 3-2：3-1 のコードを変えてみよう】

```
for i in range(1,8):
    print(i)
```

【実行結果】

```
1
2
3
4
5
6
7
```

スタートの数が変わった！
つまり・・・
range（はじまりの数，おわりの数-1）という意味！
→i の値は、1 から 7 まで代入される

【課題 3-3：3-2 のコードを変えてみよう】

```
for i in range(1,8,2):
    print(i)
```

【実行結果】

```
1
3
5
7
```

ひとつ飛ばしで表示されている！
つまり・・・
range（はじまりの数，おわりの数-1，いくつとばしか）という意味！
→i の値は、1 から一つ飛ばしで 7 まで代入される！

2. 練習問題その１

- 問題 1:画面例のように、For 文を使い、0 から 10 を表示するプログラムを作ってください

【画面例】

```
0
1
2
3
4
5
6
7
8
9
10
```

- 問題 2: 画面例のように、For 文を使い、1 から 10 まで表示するプログラムを作ってください。

【画面例】

```
1
2
3
4
5
6
7
8
9
10
```

- 問題 3:画面例のように、For 文を使い、10 から 1 を表示するプログラムを作ってください

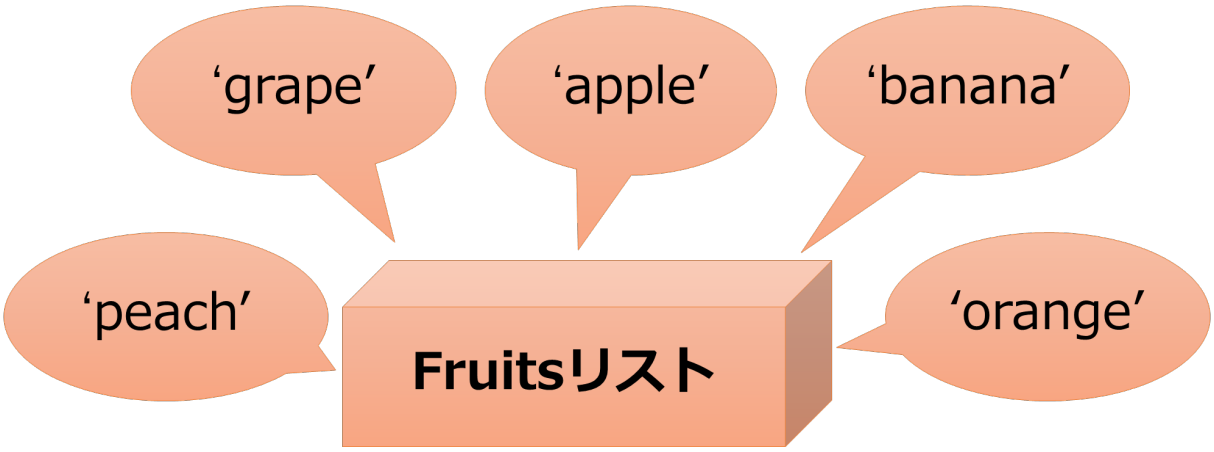
【画面例】

```
10
9
8
7
6
5
4
3
2
1
```

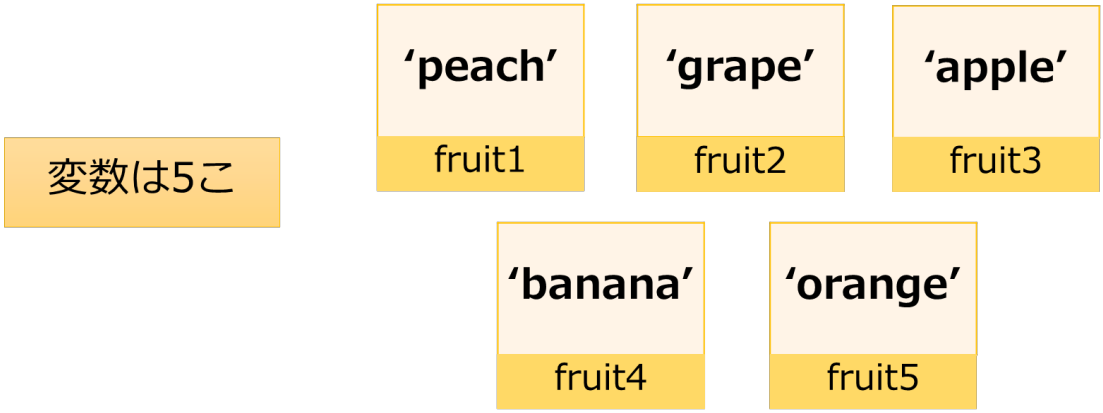
※ヒント
range 関数の 3 つめの数は、マイナスの数（-1, -2 など)を入れることもできます。

3. リスト

- リストとは？：型のひとつ。1 つの変数の中に、たくさんのデータが入れられるもの。



- リストを使えば、1 つの変数に、データをまとめることができる。リストの中は、数千・数万のデータを入れることができるので、大量のデータを扱うときは特に、一つ一つ変数の名前を考えて代入するよりも、扱いが楽になる。



- リストの代入の仕方

【課題1：入力しよう】

```
numlist=[1,2,3,4,5,6,7]
alphabet=["a","b","c","d","e","f"]
print(numlist)
print(alphabet)
```

カンマで区切ったデータを[]で囲めば OK！
データは、どんな型でもよい

【実行結果】

```
[1, 2, 3, 4, 5, 6, 7]
['a', 'b', 'c', 'd', 'e', 'f']
```

- リストの要素番号
  - 変数名[要素番号]で、リスト内のデータを参照することができる。最初のデータの番号は 0 となる。
  - 要素番号のことを、「添字(そえじ)」と言う。

【課題2：2 行目以降を入力しよう】

```
alphabet=["a","b","c","d","e","f"]
print(alphabet [0])
print(alphabet [3])
```

どのデータが表示されるだろうか？

【実行結果】

```
a
d
```

alphabet [0]	alphabet [1]	alphabet [2]	alphabet [3]	alphabet [4]	alphabet [5]
a	b	c	d	e	f

  - リストの各要素の値は書き換えることができる

【課題3：2 行目以降を入力しよう】

```
numlist=[1,2,3,4,5,6,7]
print(numlist[3])
numlist[3]=10
print(numlist[3])
```

numlist[3]は 4

numlist[3]に 10 を代入し、出力する

【実行結果】

```
4
10
```

- リストのメソッド

【本題に入る前に･･･】
リストに、新しいデータを追加することはできるの？→メソッドを使えばできる！

【課題 4：入力してみよう】

```
name=["Yamada","Sato","Kobayashi","Ito","Matsumoto"]
name.append("Abe")
print(name)
```

どんな結果になるだろうか？

【実行結果】

```
['Yamada', 'Sato', 'Kobayashi', 'Ito', 'Matsumoto', 'Abe']
```

- メソッドとは？
  - 型に特有の便利な機能
  - 変数の後にドット(.)をつけたあとの命令がメソッドにあたる(上記の例では、append())
  - メソッドはたくさんある。ただし、型によって準備されているメソッドは異なる(型によって、やれることと、やれないことがあるため)
  - リストメソッドの例:
    「Python リストのメソッド」https://qiita.com/shin_09/items/c89e002bc44343b387dc
  - リスト以外の型にもメソッドはある。
    ※文字型メソッドの例:
    「文字列処理メソッドのまとめ」http://motw.mods.jp/Python/str_methods.html

4. リストと for 文

- リストの便利なところ:for 文と組み合わせ、リストの各要素を処理することができる。

【課題 1:入力してみよう】

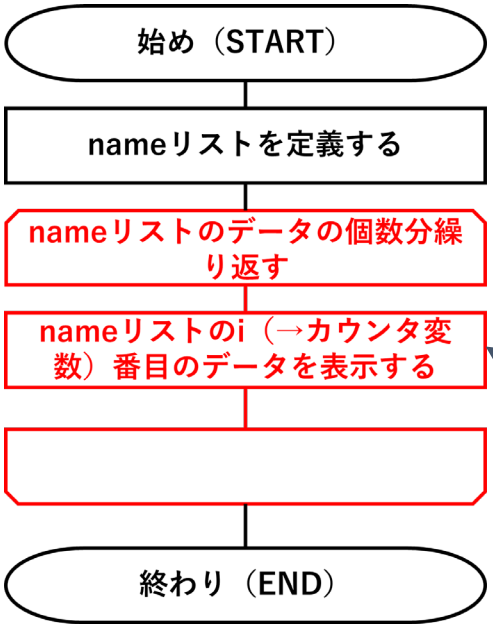
```
name=["Yamada","Sato","Kobayashi","Ito","Matsumoto"]
for i in name:
    print(i)
```

in の後にリスト型の変数を書くことができる！
どんな結果になるだろうか？

【実行結果】

```
Yamada
Sato
Kobayashi
Ito
Matsumoto
```

【フローチャートとの比較】



in の後にリスト型の変数を書くと、
データの個数分繰り返すことができる！

回数	i(カウンタ変数)の値	i 番目のデータ
1 回目	0	Yamada
2 回目	1	Sato
3 回目	2	Kobayashi
4 回目	3	Ito
5 回目	4	Matsumoto

- リストの append メソッドをつかうと、このようなこともできる。

【課題 2-1：2 行目以降を入力しよう】

```
name=["Yamada","Sato","Kobayashi","Ito","Matsumoto"]
new=input( '新しい名前を入力してください:' )
name.append(new)
for i in name:
    print(i)
```

新しい名前を、name リストの末尾に追加する

【実行結果】

```
新しい名前を入力してください：Yoshida
Yamada
Sato
Kobayashi
Ito
Matsumoto
Yoshida
```

【課題 2-2:3 行目以降を入力しよう】

```
name=["Yamada","Sato","Kobayashi","Ito","Matsumoto"]
new_name=["Yoshida","Kondo","Nakanishi"]
for i in new_name:
    name.append(i)
print(name)
```

teachers リストに、new_teachers リストの i 番目の中身を追加している

繰り返し処理が終わったら、teachers リストを表示する

【実行結果】

```
['Yamada', 'Sato', 'Kobayashi', 'Ito', 'Matsumoto', 'Yoshida', 'Kondo', 'Nakanishi']
```

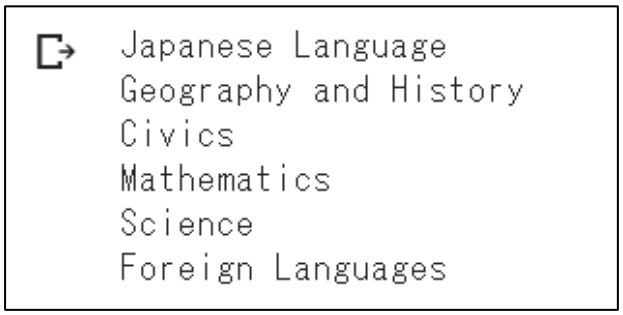
【フローチャートとの比較】

回数	i の値	new_name リスト i 番目のデータ	name リスト
1 回目	0	Yoshida	["Yamada","Sato","Kobayashi","Ito","Matsumoto","Yoshida"]
2 回目	1	Kondo	["Yamada","Sato","Kobayashi","Ito","Matsumoto","Yoshida","Kondo"]
3 回目	2	Nakanishi	["Yamada","Sato","Kobayashi","Ito","Matsumoto","Yoshida","Kondo","Nakanishi"]

5. 練習問題その 2

- 問題 1:画面例のように、For 文を使い、「科目リスト」を表示させてください。「科目リスト」はプログラム 1 行目に書かれているものを使ってください。

【画面例】



- 問題 2:画面例のような出力ができるように、プログラムを修正してください。

【プログラム】

```
course=[”iGrobal””Liberal Arts”,”Business Design”,”Life Design”]

for i in course:
print(course):
```

【画面例】