

プログラミング (Python)

第5回 リストの応用

早稲田大学本庄高等学院 2020年度版

飯島涼



前回の内容

- 繰り返し文
 - while文: 条件を満たす限り繰り返す
 - for 文: リストの長さだけ繰り返す
 - 中断
 - break 文: 中断して繰り返しから抜ける
 - continue 文: 中断して繰り返しの先頭に戻る
 - 繰り返し文のネスト, ネストされた繰り返し文
 - くりかえしの中にさらにくりかえしを書くこと

while文

while (条件):

イ	命令文1
ン	命令文2
デ
ン
ト	

カッコ内で指定した条件を満たす間
インデントした命令文を繰り返す

例3 while文とif文の組み合わせ

while文の中にif文を組み合わせて使うことができる

3回 0-1の間でランダムな数を振って、0.7以上が出れば「チャレンジ成功」と出力するプログラム

tは繰り返しの回数を表す

if文, else文で実行したい命令は, if, elseの列よりも1回分インデントする.

```
1 import random
2
3 t = 0
4 while (t < 3):
5     x = random.random()
6     if (x < 0.7):
7         print("チャレンジ失敗")
8     else:
9         print("チャレンジ成功")
10
11     t += 1
```

↳ チャレンジ失敗
チャレンジ失敗
チャレンジ成功

for文

書き方

for **変数** in **リスト**:

イ
ン
デ
ク
ス

命令1

命令2

... ..

... ..

リストの長さだけ黄色の四角
の中を繰り返す

リストの中の数が、先頭から順に
変数の中に入る

実行例

リストを用いる例



```
1 for x in [0, 1, 2, 3, 4]:  
2     x *= 2  
3     print(x)
```



```
0  
2  
4  
6  
8
```

等差数列を使う場合の例



```
1 for x in range(5):  
2     x *= 2  
3     print(x)
```



```
0  
2  
4  
6  
8
```

break / continue 文

- 繰り返し文の中で用いられる特殊な命令
 - break 文
 - この文にたどり着いたら、**繰り返しを中断して、繰り返しから抜ける。**
 - continue 文
 - この文にたどり着いたら、**今の繰り返しを中断して、次の繰り返しにうつる。**

break 文の例

本来なら、 t が10になるまで繰り返される。
($t < 10$)

チャレンジ成功した場合、
break文にたどり着くので、そこで繰り返しが強制終了される。

- 他の事例
 - 成功したらゲームを次に進める
 - パスワードがあっていたらログインさせる (3回も入力させない)
 - 体力がなくなったらGame Overと表示して繰り返しを終わらせる

```
▶ 1 import random
   2
   3 t = 0
   4 while (t < 10):
   5     x = random.random()
   6     if (x < 0.7):
   7         print("チャレンジ失敗")
   8     else:
   9         print("チャレンジ成功")
  10         break
   11
   12     t += 1
  13
  14 print("Finish.")
```

↳ チャレンジ失敗
チャレンジ失敗
チャレンジ失敗
チャレンジ失敗
チャレンジ成功
Finish.

continue 文の例

```
1 import random
2 t = 0
3 HP = 1
4 while (t < 5):
5     t += 1
6     print("現在のHP:", HP)
7     x = random.random()
8     if (x < 0.3):
9         print("チャレンジ失敗")
10        continue
11
12    print("チャレンジ成功, HPを3回復します.")
13    HP += 3
```

現在のHP: 1

チャレンジ成功, HPを3回復します.
現在のHP: 4

チャレンジ失敗

現在のHP: 4 <= HPも3増えていない

チャレンジ成功, HPを3回復します.
現在のHP: 7

チャレンジ成功, HPを3回復します.
現在のHP: 10

チャレンジ失敗

チャレンジ失敗したら, continue文にたどり着き,
これ以上先のwhile文内の命令(12, 13行目)
を実行せずに, 条件文(4行目)へ戻る.

【やや難&重要】繰り返し文のネスト，ネストされた繰り返し文

- 繰り返し文の中に，さらに繰り返しを書くこと

①: $i=0$ が代入された状態で，
 $j=0, j=1$ の繰り返しが行われる

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

➡

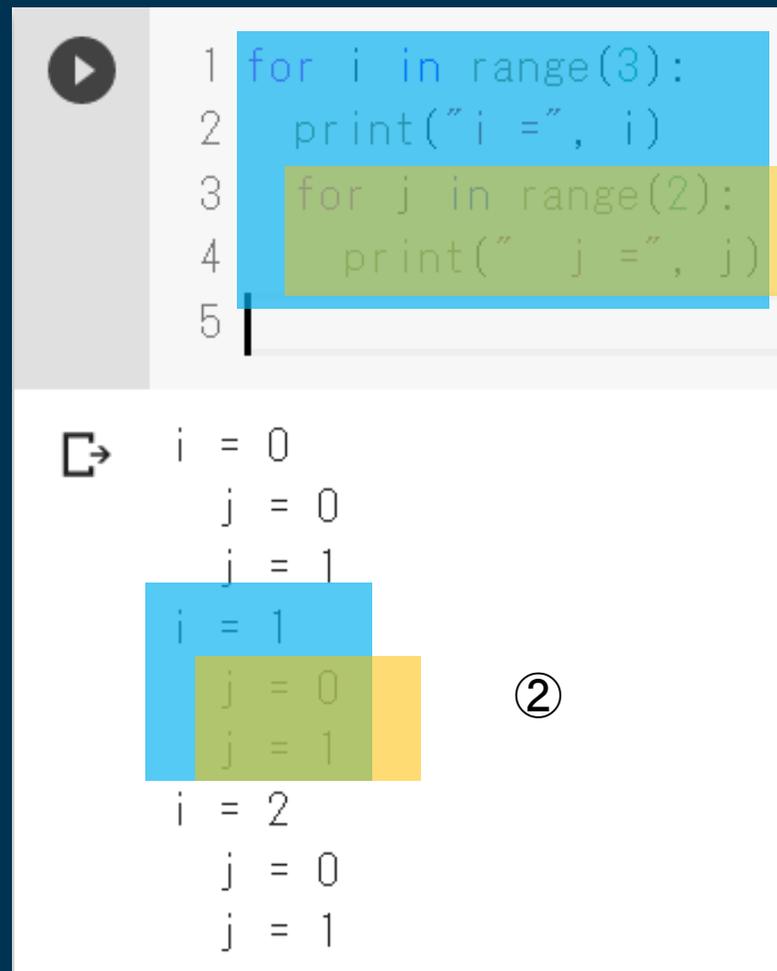
```
i = 0
j = 0
j = 1
i = 1
j = 0
j = 1
i = 2
j = 0
j = 1
```

①

【やや難&重要】繰り返し文のネスト，ネストされた繰り返し文

- 繰り返し文の中に，さらに繰り返しを書くこと

②: $i=1$ が代入された状態で，
 $j=0, j=1$ の繰り返しが行われる



The image shows a code editor with a play button icon on the left. The code is as follows:

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

Below the code, the execution output is shown. The first three lines are $i = 0$, $j = 0$, and $j = 1$. The fourth line is $i = 1$, which is highlighted in blue. The fifth and sixth lines are $j = 0$ and $j = 1$, which are highlighted in green. The seventh line is $i = 2$, followed by $j = 0$ and $j = 1$. A circled number 2 is placed to the right of the $j = 0$ and $j = 1$ lines under $i = 1$.

【やや難&重要】繰り返し文のネスト, ネストされた繰り返し文

- 繰り返し文の中に, さらに繰り返しを書くこと

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

↳

```
i = 0  
    j = 0  
    j = 1  
i = 1  
    j = 0  
    j = 1  
i = 2  
    j = 0  
    j = 1
```

③

- ③: $i=2$ が代入された状態で,
 $j=0, j=1$ の繰り返しが行われる

本日の内容

- リストの応用 (p.142-181)
 - 要素の追加・削除
 - リストとfor文・if文の組み合わせ
 - リスト内包表記
 - split()機能の利用
 - input()で受け取った文字列のリスト化

リストをなんのために使うのか？

数字や文字の連なりを表現するため

- 統計解析
 - 1日ごとの平均外出者数: [120000, 130000, 125000, 300000, … …]
 - 株価・GDPなど
- 連なりを表す必要があるもの（時系列データという）
 - 楽譜・MIDIデータ・コード進行等: [“F”, “G”, “Em”, “Am”, … …]
 - 電車の駅名: [“神保原”, “本庄”, “岡部”, “深谷”, … …]

リストの要素の追加・削除

- 追加

- `append(値)`: 末尾に値を追加する.
- `insert(追加位置, 値)`: 指定した追加位置に, 値を追加する.

- 削除

- `pop()`: 末尾の値を削除する.
- `pop(位置)`: 指定した位置の値を削除する.

リストへの要素の追加例

- 追加

- `append(値)`: 末尾に値を追加する.
- `insert(追加位置, 値)`: 指定した追加位置に, 値を追加する.

```
▶ 1 data = [] # 空のリスト
2
3 data.append(10)
4 print(data)
5
6 data.append(20)
7 data.append(40)
8 print(data)
9
10 data.insert(2, 30)
11 print(data)
```

```
↳ [10]
   [10, 20, 40]
   [10, 20, 30, 40]
```

リストの要素の削除例

- 削除

- `pop()`: 末尾の値を削除する.
- `pop(位置)`: 指定した位置の値を削除する.

```
1 data = [10, 20, 30, 40]
2 print(data)
3
4 #末尾の要素を削除する pop()
5 data.pop()
6 print(data)
7
8 #1番目の要素を削除する pop(1)
9 data.pop(1)
10 print(data)
```

```
↳ [10, 20, 30, 40]
   [10, 20, 30]
   [10, 30]
```

リストの要素追加・削除の使い方

使用例

ランダムにさいころを振った時の結果を入れていく。



```
1 import random
2
3 results = []
4
5 for i in range(10):
6     x = random.randint(1, 6)
7     results.append(x)
8     print(results)
```

```
[3]
[3, 1]
[3, 1, 1]
[3, 1, 1, 2]
[3, 1, 1, 2, 4]
[3, 1, 1, 2, 4, 5]
[3, 1, 1, 2, 4, 5, 6]
[3, 1, 1, 2, 4, 5, 6, 4]
[3, 1, 1, 2, 4, 5, 6, 4, 4]
[3, 1, 1, 2, 4, 5, 6, 4, 4, 4]
```

シミュレーション入門

さいころを1000回ふったら本当に均等に数が現れるか？

Pythonで代わりにさいころを1000回振ってもらおう

使用するモジュール: `collections`

使用する機能: `Counter()`

リスト内に現れる要素の個数をカウントする

シミュレーション結果

```
▶ 1 import random
2 import collections
3
4 results = []
5
6 for i in range(1000):
7     x = random.randint(1, 6)
8     results.append(x)
9
10 print(collections.Counter(results))
11
12
13
```

```
↳ Counter({3: 175, 6: 174, 1: 168, 4: 167, 2: 160, 5: 156})
```

大まかに結果がそろっている

【やや難】リスト内包表記

リストを作成するための便利技

リスト内包表記の書き方

[式 for 変数 in リスト]

リストから要素を一つずつ取り出して変数に代入し、その変数を含む式で計算された値を要素として、新しくリストを作る。

リスト内包表記の利用例

[式] for 変数 in リスト]

```
▶ 1 # リスト内の, 0-1 の確率をすべて100%表記に直したい.  
2  
3 L = [0.324, 0.452, 0.335, 0.562, 0.583, 0.213]  
4  
5 new_L = [per*100 for per in L]  
6  
7 print("new_L =", new_L)  
↳ new_L = [32.4, 45.2, 33.5, 56.2, 58.3, 21.3]
```

L から要素を一つずつ取り出して変数 per に代入し、その per を含む式 per*100 で計算された値を新しい要素として、リスト new_L を作る。

split(“区切り文字”)

指定した区切り文字で、文字列を分割して、リストを作る。

リストの要素はすべて文字列となる。

具体例

```
1 # , (カンマ)区切りの文字列を分ける
2
3 stations = "Jan, Feb, Mar, Apr, May, Jun, Jul, Aug, Sep, Oct, Nov, Dec"
4
5 print(stations.split(","))
```

```
['Jan', 'Feb', 'Mar', 'Apr', 'May', 'Jun', 'Jul', 'Aug', 'Sep', 'Oct', 'Nov', 'Dec']
```

```
1 # 空白区切りの時には, split()のかっこ内は省略可
2
3 foods = "rice bread noodle"
4
5 print(foods.split())
```

```
['rice', 'bread', 'noodle']
```

split(“区切り文字”) の利用用途

input()と組み合わせて使う

⇒ 複数の値を一気に受け取りたいとき

使用例

```
▶ 1 S = input("好きな食べ物を半角スペースで区切って入力してください: ")
2
3 print(S) # すべて1つの文字列として扱われてしまう
4
5 foods = S.split()
6
7 print(foods) # リストとして出力される
```

```
↳ 好きな食べ物を半角スペースで区切って入力してください: gobou noodle fish meet
gobou noodle fish meet
['gobou', 'noodle', 'fish', 'meet']
```

split() input() リスト内包表記を合わせた 使用例

input()で受け取った数字をすべて足した結果を出力

```
▶ 1 S = input("合計したい数字をすべて半角数字で入力してください: ")
   2
   3 L = S.split()
   4 print(L)
   5
   6 ans = ""
   7 for i in L:
   8     ans += i
   9
  10 print(ans)
  11
```

```
↳ 合計したい数字をすべて半角数字で入力してください: 1 2 3 4 5 6 7 8 9
   ['1', '2', '3', '4', '5', '6', '7', '8', '9']
   123456789
```

シングルクォーテーションで
囲まれているものは文字の扱い

要素1つ1つが文字として
扱われてしまい、計算ができない

⇒ 文字を数字に変換した
新しいリストを作成する.

⇒ リスト内包表記を利用

リスト内包表記を用いて修正した結果

```
▶ 1 S = input("合計したい数字をすべて半角数字で入力してください: ")
2
3 L = S.split()
4 print("文字を要素としたリストL = ", L)
5
6 # 文字を数字に変換する機能 int() をリストLの要素1つ1つに適用する.
7 new_L = [int(n) for n in L]
8 print("数字を要素としたリストnew_L = ", new_L)
9
10 ans = 0
11 for i in new_L:
12     ans += i
13
14 print(ans)
```

int(): 文字列を数字として扱えるように変更する

```
↳ 合計したい数字をすべて半角数字で入力してください: 1 2 3 4 5 6 7 8 9
文字を要素としたリストL = ['1', '2', '3', '4', '5', '6', '7', '8', '9']
数字を要素としたリストnew_L = [1, 2, 3, 4, 5, 6, 7, 8, 9]
45
```

自由課題（今回は復習がメイン）

[1] 授業で示した例以外で、リストの使い道を教えてください。過去に自由課題で使用したプログラムを例に挙げて構いません。

[2] TRPGでは、 $1-M$ の整数のさいころを N 回振ることを NDM と表現します（例：2D100ならば、 $1-100$ までのさいころを2回振る）。NDMという文字列を受け取って、目的通りのさいころを回数分振った結果を、リストresultsに格納して表示してください。

自由課題（今回は復習がメイン）

[3] 合計したい数字を半角数字で入力すると、その合計結果を示すプログラムを作成してください。

[4] コインを1000回投げて、表(H)と裏(T)の回数を表示するプログラムを作成してください。

リストは使用してもしなくてもOKだが、使用すると授業中使ったものを応用できて楽

前回の課題を使いまわしてもOK