



pf-4. リストと繰り返し

(Python プログラミング入門演習, 全6回)

<https://www.kkaneko.jp/cc/pf/index.html>

金子邦彦





アウトライン

4-1 リスト

4-2 リストと繰り返し (ループ)

- リスト, 繰り返し (ループ) を学ぶ.
- ステップ実行により, 変数等の変化, プログラム実行の流れを観察できることを学ぶ.



4-1. リスト



リスト

- リストは、同じ型の要素の並び
- 各要素には、0から始まる番号（添字）が付いている

0	1	2	3	4
8	6	4	2	3

list

0	1	2	3	4
8.8	6.6	4.4	2.2	3.3



実習の指示

- 資料：6～11
- 次のことを理解しマスターする
 - リストを扱う変数

実習



① ウェブブラウザを起動する

② Python Tutor を使いたないので、次の URL を開く

<http://www.pythontutor.com/>

※ Internet Explorer でうまく動かない場合がある

→ うまく動かないときは Google Chrome を試してください

※ 途中で「Server Busy . . .」というメッセージが出る
ことがある。

→ 混雑している。少し（数秒から数十秒）待つと自動で表示が変わる（変わらない場合には、操作をもう一度行ってみる）

※ 日本語モードはない。英語で使う



③ 「Python Tutor」 をクリック

← → ↻ ⓘ 保護されていない通信 | pythontutor.com ☆ APP 2 🔒 📄 👤 | K

VISUALIZE CODE AND GET LIVE HELP

Learn Python, Java, C, C++, JavaScript, and Ruby

[Python Tutor](#) (created by [Philip Guo](#)) helps people overcome a fundamental barrier to learning programming: understanding what happens as the computer runs each line of code.

Write code in your web browser, see it visualized step by step, and get live help from volunteers.

Related services: [Java Tutor](#), [C Tutor](#), [C++ Tutor](#), [JavaScript Tutor](#), [Ruby Tutor](#)

Over five million people in more than 180 countries have used Python Tutor to visualize over 100 million pieces of code, often as a supplement to textbooks, lectures, and online tutorials.

[Visualize your code and get live help now](#)



Get live help!

Start private chat

These Python Tutor users are asking for help right now. Please volunteer to help!
user_041 from Singapore, Singapore needs help with Python3 - 2 people chatting - requested 12 minutes ago)
user_91c from Vélizy-Villacoublay, France needs help with Python3 - [click to help](#) minutes ago)
user_985 from Seattle, Washington, US needs help with Python3 - [click to help](#) (-

「Python 3.6」になっている

Write code in Python 3.6

```
1 |
```

エディタ

Help improve this tool by completing a [short user survey](#)

Visualize Execution

実行のためのボタン
Live Programming Mode

hide exited frames [default] inline primitives but don't nest objects [default]
draw pointers as arrows [default]

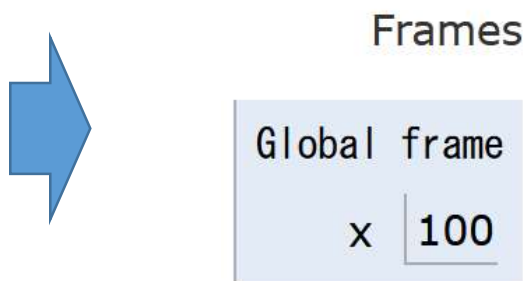
Python Tutor でのプログラム実行手順



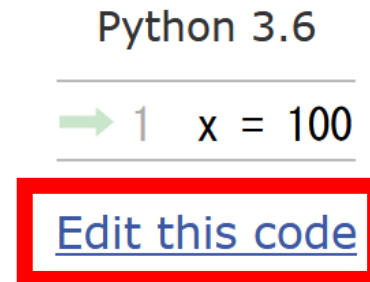
(1) 「**Visualize Execution**」をクリック。



(2) 「**Last**」をクリック。



(3) 結果を確認する。



(4) 「**Edit this code**」をクリックして戻る



④ 次のプログラムを実行し，結果を確認しなさい

リストデータ



```
1 x = [8, 6, 4, 2, 3]
2 print(sum(x))
```



合計を求める

Python 3.6

```
1 x = [8, 6, 4, 2, 3]
2 print(sum(x))
```

Edit this code

Print output (drag lower right corner to resize)

23

Frames

Global frame

x

Objects

0	1	2	3	4
8	6	4	2	3

Done running (2 steps)

結果の
「23」が表示
されるので確認

月の日数



⑤ 次のプログラムを実行し，結果を確認しなさい

月の日数についてのリストデータ

※ うるう年のことは考えないことにする

```
1 days = [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
2 print(days[9])
```

9月について表示

Print output (drag lower right corner to resize)

30

Frames

Objects

Global frame

days

list

0	1	2	3	4	5	6	7	8	9	10	11	12
0	31	28	31	30	31	30	31	31	30	31	30	31

結果の
「30」が表示
されるので確認



4-2. リストと繰り返し (ループ)

繰り返し

- 繰り返し（ループ）では、同じ処理や操作を繰り返す

繰り返しのプログラム例



リストの
組み立て

```
1 x = [8, 6, 4, 2, 3]
2 y = [0, 0, 0, 0, 0]
3 for i in [0, 1, 2, 3, 4]:
4     y[i] = x[i] * 1.1
```

「 $y[i] = x[i] * 1.1$ 」を
 i の値を変えながら
5回繰り返す



実習の指示

- 資料：16～27
- 次のことを理解しマスターする
 - for による繰り返し（ループ）

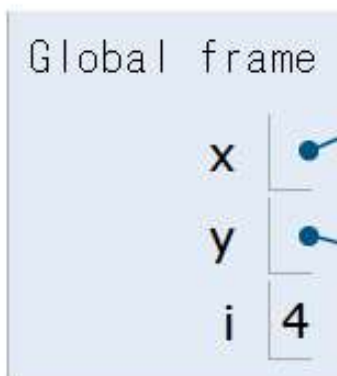


① 次のプログラムを実行し，結果を確認しなさい

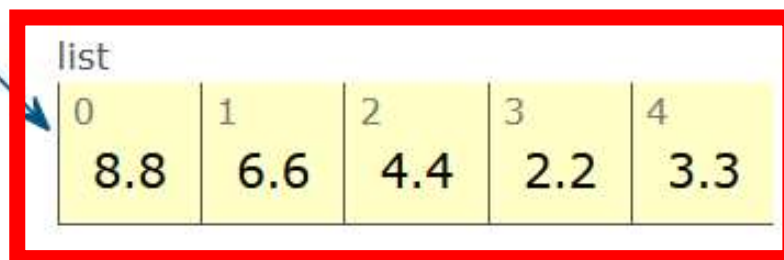
```
1 x = [8, 6, 4, 2, 3]
2 y = [0, 0, 0, 0, 0]
3 for i in [0, 1, 2, 3, 4]:
4     y[i] = x[i] * 1.1
```

Frames

Objects



オブジェクト x は
5 個の要素が入った
リスト



オブジェクト y は
5 個の要素が入った
リスト



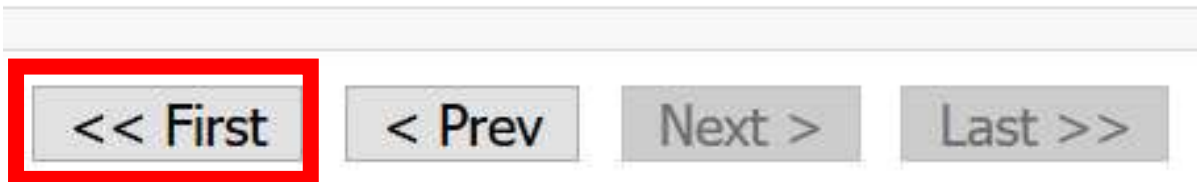
② 「First」 をクリックして, プログラム実行を先頭
に戻す

Python 3.6

```
1 x = [8, 6, 4, 2, 3]
2 y = [0, 0, 0, 0, 0]
→ 3 for i in [0, 1, 2, 3, 4]:
4     y[i] = x[i] * 1.1
```

[Edit this code](#)

ected
ite



Done running (13 steps)



- ③ 「13 steps」と表示されているので、
全部で、ステップ数は 13 あることが分かる
(ステップ数と、プログラムの行数は違うもの)

Python 3.6

```
1 x = [8, 6, 4, 2, 3]
2 y = [0, 0, 0, 0, 0]
→ 3 for i in [0, 1, 2, 3, 4]:
4     y[i] = x[i] * 1.1
```

[Edit this code](#)

Output
ite

<< First < Prev Next > Last >>

Done running (13 steps)



④ ステップ実行したいので、「Next」をクリックしながら、矢印の動きを確認しなさい。

※ 「Next」 ボタンを何度か押し、それ以上進めなくなったら終了

Python 3.6

```
1 x = [8, 6, 4, 2, 3]
2 y = [0, 0, 0, 0, 0]
3 for i in [0, 1, 2, 3, 4]:
4     y[i] = x[i] * 1.1
```

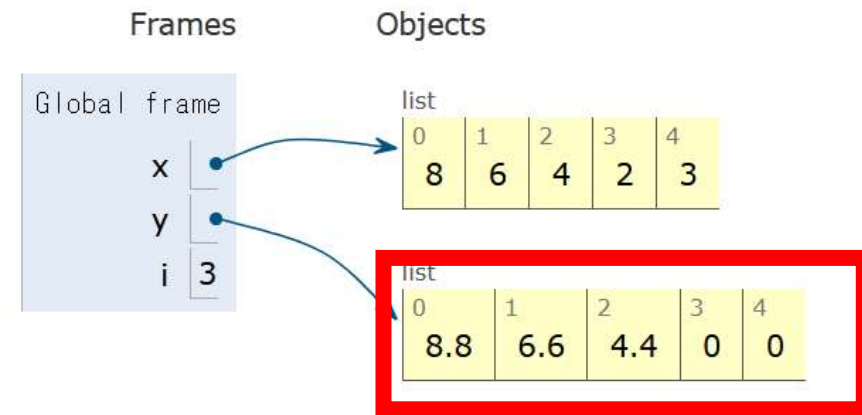
[Edit this code](#)

Executed
Execute

<< First < Prev Next > Last >>

Step 10 of 13

見どころ
3行目, 4行目が
繰り返される



実行が進むと、
y の中身が更新される



⑤ 終わったら「Edit this code」をクリックして、エディタ画面に戻る

Python 3.6

```
1 x = [8, 6, 4, 2, 3]
2 y = [0, 0, 0, 0, 0]
→ 3 for i in [0, 1, 2, 3, 4]:
4     y[i] = x[i] * 1.1
```

[Edit this code](#)

uted
:e

<< First

< Prev

Next >

Last >>

Done running (13 steps)

リストと繰り返し



① Python Tutor のエディタで次のように書きなさい

物体を落とすと $9.8 \times (\text{時間})^2 \div 2$ の分, 落ちていく. (空気抵抗は無視する)

Write code in Python 3.6

```
1 x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2 for t in x:
3     print( (9.8 / 2) * t * t )
```

for t in x の直後に「:」
(コロン)

字下げも正確に!
print の前に, 「タブ」を1つだけ



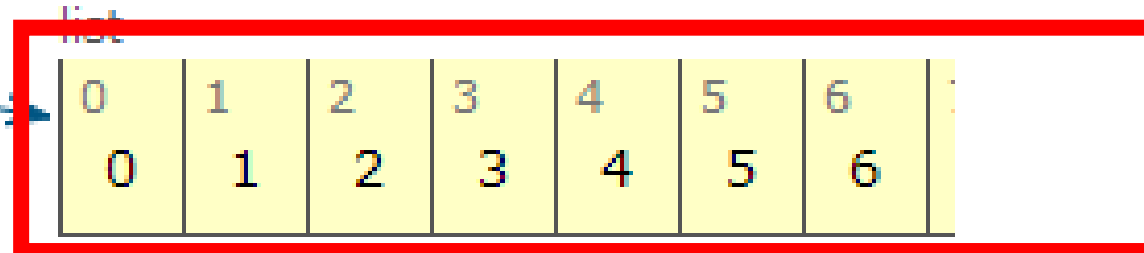
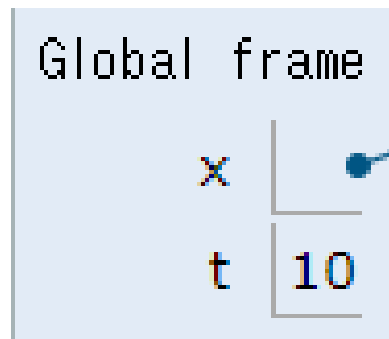
② 実行し，結果を確認しなさい。

Print output (drag lower right corner to resize)

```
122.5
176.4
240.10000000000000002
313.6
396.90000000000000003
490.0
```

Frames

Objects



オブジェクト x は 11 個の要素が入ったリスト

- 実行結果の確認では、必要に応じて、
- 表示枠を広げなさい

Print output (drag lower right corner to resize)

```
0.0  
4.9  
19.6  
44.1  
78.4  
122.5  
176.4  
240.1  
313.6  
396.9  
490.0
```



ここをドラッグすると、
表示枠が広がる



③ 「First」 をクリックして，最初の行に戻しなさい

Python 3.6

```
1 x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
→ 2 for t in x:
3     print( (9.8 / 2) * t * t )
```

[Edit this code](#)

that has just executed

t line to execute

ne of code to set a breakpoint; use the Back and Forward buttons to jump there.

<< First

< Back

Program terminated

Forward >

Last >>



- ④ 「Step 1 of 24」と表示されているので、
全部で、ステップ数は24あることが分かる
(ステップ数と、プログラムの行数は違うもの)

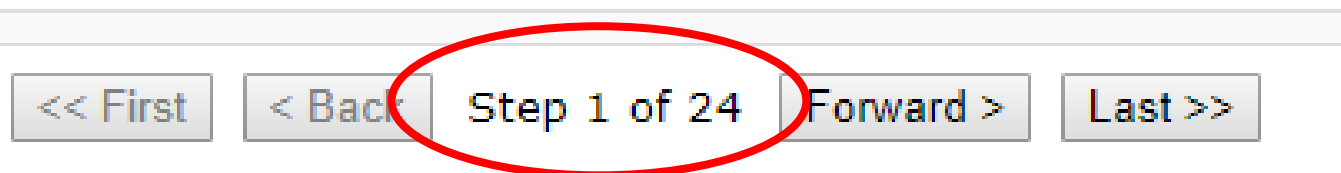
Python 3.6

```
→ 1 x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
   2 for t in x:  
   3     print( (9.8 / 2) * t * t )
```

[Edit this code](#)

has just executed
to execute

code to set a breakpoint; use the Back and Forward buttons to jump there.





⑤ ステップ実行したいので、「Forward」をクリックしながら、緑の矢印の動きを確認しなさい。

※ 「Forward」 ボタンを何度か押し、それ以上進めなくなったら終了

Python 3.6

```
1 x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
2 for t in x:
3     print( (9.8 / 2) * t * t )
```

[Edit this code](#)

It has just executed
Click here to execute

Click here to set a breakpoint; use the Back and Forward buttons to jump there.

<< First < Back Step 11 of 24 **Forward >** Last >>

Support with a [small donation](#).

Help improve this tool by clicking whenever you learn something:

Print output (drag lower right corner to resize)

```
0.0
4.9
19.6
44.1
```

Frames

Global frame	
x	→
t	4

Objects

list	0	1	2	3	4
	0	1			

見どころ)
2行目と3行目を
進んだり、戻ったりする
ところ



⑥ 終わったら「Edit this code」をクリックして、元の画面に戻る

```
Python 3.6  
1 x = [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10]  
→ 2 for t in x:  
3     print( (9.8 / 2) * t * t )
```

[Edit this code](#)

that has just executed
t line to execute

line of code to set a breakpoint; use the Back and Forward buttons to jump there.

<< First < Back Program terminated Forward > Last >>

id by @pgbovine. Support with a [small donation](#).

Help improve this tool by clicking whenever you learn something:

Print output (drag lower right corner to resize)

```
0.0  
4.9  
19.6  
44.1  
78.4  
122.5  
176.4  
240.10000000000002  
313.6  
396.90000000000003  
490.0
```

Frames

Objects

Global frame

x	
t	10

list

0	1	2
0	1	2