

Python プログラム実行, Python 環境

(Python 入門)

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

金子邦彦



- **Python**は複数の環境で実行可能
- **Visual Studio Code**ではプログラムの編集と実行が可能
- **Spyder**は**グラフ表示や画像の表示**が簡単にできる。
- **Jupyter ノートブック**では、ノートブック形式でプログラムを編集・実行し、**ドキュメント化が容易で、再現性が高い**です。
- 以上の環境はいずれも、プログラム作成・編集・実行、シンタックスハイライト、自動補完、自動インデント、変数探索などの機能を提供する。
- **オンラインの実行環境**として **Google Colaboratory, trinket, Python Tutor, Repl.it** など多数ある。
- それぞれの環境には特徴があり、適切に選ぶことで作業効率が向上する。

Python プログラムの実行①



① Windows でコマンドプロンプトを使用. プログラムを入れるたびに結果が得られる（対話的実行と言ったりする）.

コマンドプロンプトで
Pythonで開始

```
C:\Users\user>python
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022)
Type "help", "copyright", "credits" or "license()"
>>> x = 100
>>> if (x > 20):
...     print("big")
... else:
...     print("small")
...
big
>>> s = 0
>>> for i in [1, 2, 3, 4, 5]:
...     s = s + i
...
>>> print(s)
15
>>>
```

Python
プログラ
ム
実行
結果

- Python のインストール必要
<https://www.python.org>
- Windows では, **python** コマンドで実行
- 終了は **exit()**

Python プログラムの実行②



② **エディタ**の **Visual Studio Code** を使用. 編集画面でプログラムを編集し, ターミナル (端末) で実行結果を確認

メニュー

実行ボタン

編集

実行結果

```
hello.py - Visual Studio Code
hello.py x
C > Users > user > hello.py > ...
1 x = 100
2 if (x > 20):
3     print("big")
4 else:
5     print("small")
6
7 s = 0
8 for i in [1, 2, 3, 4, 5]:
9     s = s + i
10 print(s)
11
```

問題 出力 デバッグ コンソール ターミナル

```
PS C:\Users\use> c:; cd 'c:\Users\user'; & 'C:\Program Files\Python310\python.exe' 'c:\Users\user\.vscode\extensions\ms-python.python-2023.12.0\pythonFiles\lib\python\debugpy\adapter\..\..\debugpy\launcher' '50351' '--' 'c:\Users\user\hello.py'
big
15
PS C:\Users\user>
```

- **Python** のインストール必要
<https://www.python.org>
- **Visual Studio Code** のインストール必要
<https://www.microsoft.com/ja-jp/dev/products/code-vs.aspx>

※ 「デバッグ」の機能により変数探索も可能

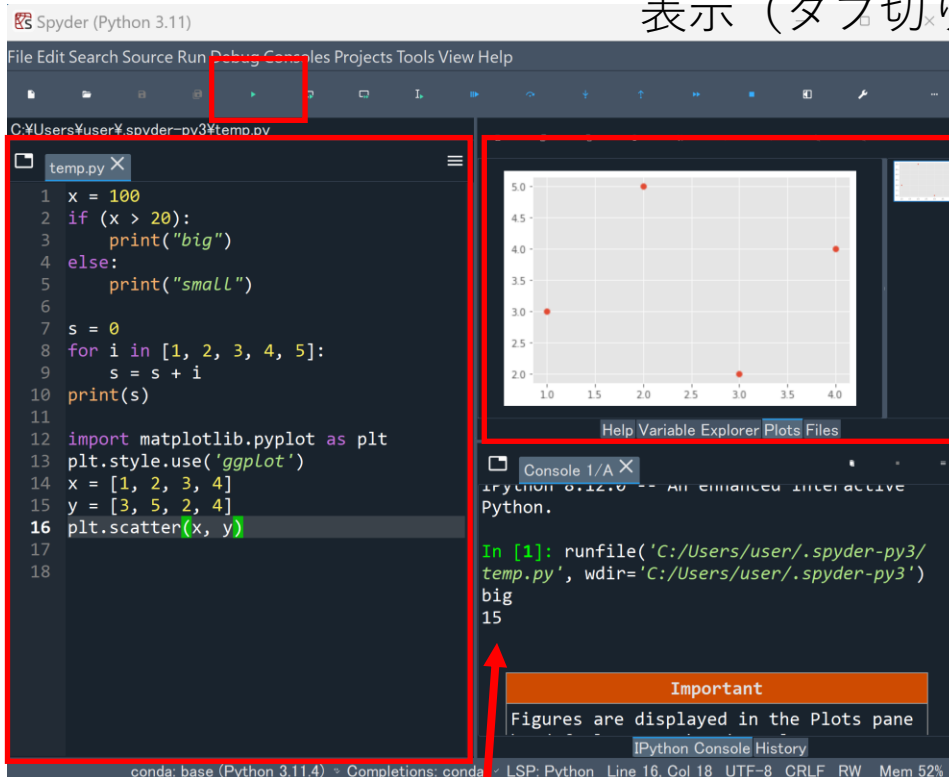
Python プログラムの実行③



③ **Spyder** を使用. 編集画面でプログラムを編集し, コンソールで実行結果を確認. **グラフ表示や画像の表示**が簡単にできる.

実行ボタン

変数の値, 図やグラフの表示 (タブ切り替え)



編集

実行結果

※ 画面右上の「Variable Explorer」タブにより変数探索も可能

- Python のインストール必要
<https://www.python.org>
- spyder は Anaconda に含まれている
<https://www.anaconda.com>

Python プログラムの実行④



④ **Jupyter ノートブック** を使用. ノートブックの画面が開き, コードセルやテキストセルを追加可能. 実行ボタンで実行. **グラフ表示や画像の表示**が簡単にできる. 全体を**ノートブック**として保存可能

The screenshot shows a Jupyter Notebook with three code cells and their outputs. The first cell contains an if-else statement that prints 'big' because x=100 is greater than 20. The second cell contains a for loop that calculates the sum of integers from 1 to 5, resulting in 15. The third cell contains a scatter plot using matplotlib, showing four data points at (1, 3), (2, 5), (3, 2), and (4, 4). The 'Run' button in the toolbar is highlighted with a red box.

```
In [1]: x = 100
if (x > 20):
    print("big")
else:
    print("small")
big

In [2]: s = 0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print(s)
15

In [3]: import matplotlib.pyplot as plt
plt.style.use('ggplot')
x = [1, 2, 3, 4]
y = [3, 5, 2, 4]
plt.scatter(x, y)

Out[3]: <matplotlib.collections.PathCollection at 0x1ff4f198d10>
```

編集

実行結果

編集

実行結果

編集

実行結果

実行ボタン

- **Python** のインストール必要
<https://www.python.org>
- **Jupyter ノートブック** は
Anaconda に含まれている
<https://www.anaconda.com>

※ 変数探索は「%whos」

```
%whos
```

Variable	Type	Data/Info
i	int	5
plt	module	<module 'matplotlib.pyplot' from ...>
s	int	15
x	list	n=4
y	list	n=4

Python開発におけるVisual Studio Code、Spyder、およびJupyter ノートブックの主要な機能

- プログラム作成、編集、実行及び結果表示
- シンタックスハイライト（可読性向上）
- 自動補完（プログラム作成支援）
- 自動インデント（プログラム作成支援）
- 変数探索
- ファイル管理（閲覧、編集、作成）
- マニュアル表示による情報参照

それぞれの特徴



Visual Studio Code

- **豊富な言語対応**

Spyder

- **グラフ表示や画像表示**
- **便利な変数探索**（プログラム終了後も変数確認可能）

Jupyter ノートブック

- **ノートブック**（プログラム, 実行結果, テキスト, 画像, 数式）
- **高い再現性**（実行の一連の手順のノートブックに残り再現可能）
- **グラフ表示や画像表示**

開発環境のメリット



- 1.多様な開発環境の理解**：適切な環境を選択することで作業効率が向上する
- 2.データ分析への理解**：Spyder, Jupyter ノートブックなどは、データ視覚化と分析が容易に可能
- 3.ドキュメンテーションと再現性**：ノートブックである Jupyter ノートブックはドキュメンテーションと結果の再現性に役立つ
- 4.変数探索**：デバッグ機能は効率的なコード開発に役立つ
- 5.効率の向上**：自動補完や自動インデントなどの機能は作業効率とコーディングスキルを向上につながる

オンラインの開発環境



Colaboratory へようこそ
ファイル 編集 表示 挿入 ランタイム ツール ヘルプ 変更を保存できませんでした

+ コード + テキスト ドライブにコピー

```
[6] x = 100
if (x > 20):
    print("big")
else:
    print("small")

big
```

```
[7] s = 0
for i in [1, 2, 3, 4, 5]:
    s = s + i
print(s)

15
```

```
import matplotlib.pyplot as plt
plt.style.use('ggplot')
x = [1, 2, 3, 4]
y = [3, 5, 3, 5]
plt.scatter(x, y)
plt.show()
```

x	y
1	3
2	5
3	3
4	5

MiniatureOldlaceHexagons
kunihikokaneke

main.py

```
1 x = 100
2 if (x > 20):
3     print("big")
4 else:
5     print("small")
6
7 s = 0
8 for i in [1, 2, 3, 4, 5]:
9     s = s + i
10 print(s)
11
12 import matplotlib.pyplot as plt
13 plt.style.use('ggplot')
14 x = [1, 2, 3, 4]
15 y = [3, 5, 3, 5]
16 plt.scatter(x, y)
17 plt.show()
```

Figure 1

Console

```
big
15
```

Google Colaboratory

<https://colab.research.google.com>

ノートブックの画面が開き，コードセルやテキストセルを追加可能。

Repl.it

<https://replit.com>

編集画面でプログラムを編集し，コンソールで実行結果を確認。グラフ表示や画像の表示が簡単にできる。多数の言語をサポート。

オンラインの開発環境



Python Tutor: Visualize code in [Python](#), [JavaScript](#), [C](#), [C++](#), and [Java](#)

Python 3.6
[known limitations](#)

```
1 x = 100
2 if (x > 20):
3     print("big")
4 else:
5     print("small")
6
7 s = 0
8 for i in [1, 2, 3, 4, 5]:
9     s = s + i
10 print(s)
```

Print output (drag lower right corner to)

```
big
15
```

Frames Objects

Global frame	
x	100
s	15
i	5

[Edit this code](#)

→ line that just executed
→ next line to execute

<< First < Prev Next > Last >>

Done running (16 steps)

Ads keep this tool free; we are not responsible for contents of displayed ads
[Move and hide objects](#)

Python Tutor

<https://pythontutor.com>

編集画面でプログラムを編集し、
実行結果を確認。

Home / My Trinkets / Untitled

Save Cancel

main.py

```
1 x = 100
2 if (x > 20):
3     print("big")
4 else:
5     print("small")
6
7 s = 0
8 for i in [1, 2, 3, 4, 5]:
9     s = s + 1
10 print(s)
```

Powered by trinket

```
big
15
```

trinket

<https://trinket.io>

編集画面でプログラムを編集し、
実行結果を確認。

オンラインの開発環境のメリット



オンラインでプログラミングを実行する Google Colaboratory、trinket、Repl.it、Python Tutor など

- 1.アクセスの容易さ** インターネット接続と Web ブラウザを通じてすぐに利用可能で、開発環境のインストール不要.
- 2.プログラムの共有や公開の容易さ:** Google Colaboratory、trinket、repl.it では、複数の利用者による共有やプログラムの公開が容易
- 3.ビジュアルでインタラクティブな実行:** 実行結果をリアルタイムでビジュアルに確認可能

プログラミングを学ぶときや、プログラムの開発プロジェクトでも有用. 多くの場合、**アカウントの作成やログインが必要. 有料の利用料金**が発生する場合もある. よく確認してから利用すること.

オンラインの開発環境のそれぞれの特徴



Google Colaboratory

- Jupyter ノートブックをオンラインで利用可能
- **グラフ表示や画像表示**
- AI に関連する多くのパッケージインストール済み。 **必要なパッケージの追加も可能**
- Google Drive と統合、ノートブックの共有や保存が容易

Repl.it

- **グラフ表示や画像表示**
- **必要なパッケージの追加も可能**
- 組み込まれたシェル (bash) により、様々なコマンドを実行可能
- **リアルタイムの共同編集機能やプロジェクトの公開・共有機能も提供**
- 多くのプログラミング言語をサポート

Python Tutor

- **変数探索やステップ実行をビジュアルに可能**
- 主に学習目的
- Python、JavaScript、C、C++、Java のプログラミング言語をサポート

Trinket

- タートルグラフィックスをサポート
- **作成したプログラムをオンラインで他の人が実行することが容易に可能**
- 主に学習目的
- Python、HTML、JavaScript などのプログラミング言語をサポート

デバッグのための主な機能



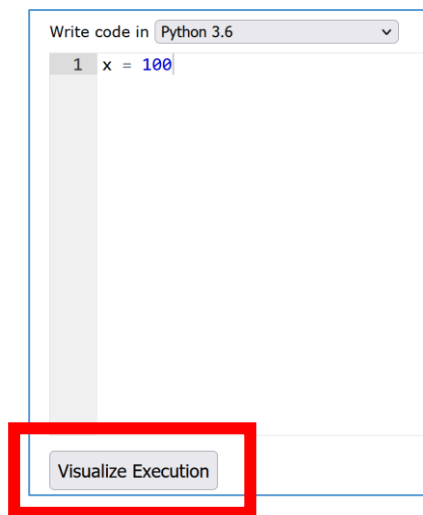
- **ステップ実行**では、**1行ずつの実行**が行われ、そのときの変数の値の変化などを**確認**できる
- **ステップ実行**により、**プログラムの動作を細かく追跡**でき、不具合が発生している箇所の特定、プログラムの学習に役立つ
- **通常実行**は、**プログラムを最初から最後まで一度に実行**するもの（プログラム実行中の変数の値の変化を確認するなどは困難）。**ステップ実行**は、**プログラムを1行ずつ実行し、実行後にプログラムを一時停止**するもの。



Python Tutor でのプログラム実行

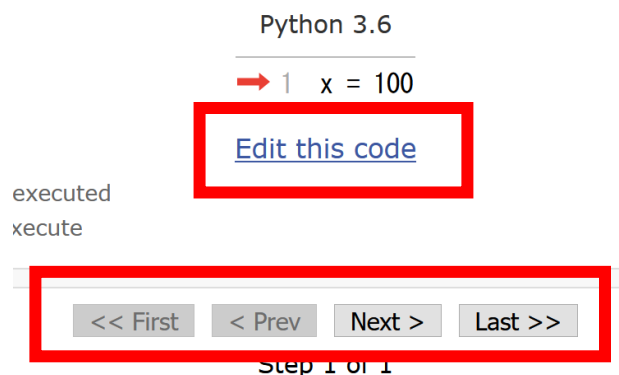
- Python Tutor は Python などのプログラムを書き実行できるサイト。ステップ実行、変数の値表示などの機能がある。
- Python Tutorのウェブサイトアクセス。「Python」を選択
<https://www.pythontutor.com/>

メイン画面で、プログラムを書く



Visualize Execution ボタン

メイン画面に
戻るには
Edit this code



通常実行: Last
ステップ実行: 他のボタン

変数の値を
視覚的に
確認できる

Frames

