

【講習会】 Python初学者セミナー

第1回 2026年06月29日（月） 12:15-13:15

第2回 2026年07月02日（木） 12:15-13:15



1 . Python 概要

プログラミング言語: Python - データから問題を解決

スマートフォンの
使用は学生の学業
成績に有意な影響
を及ぼすか....？

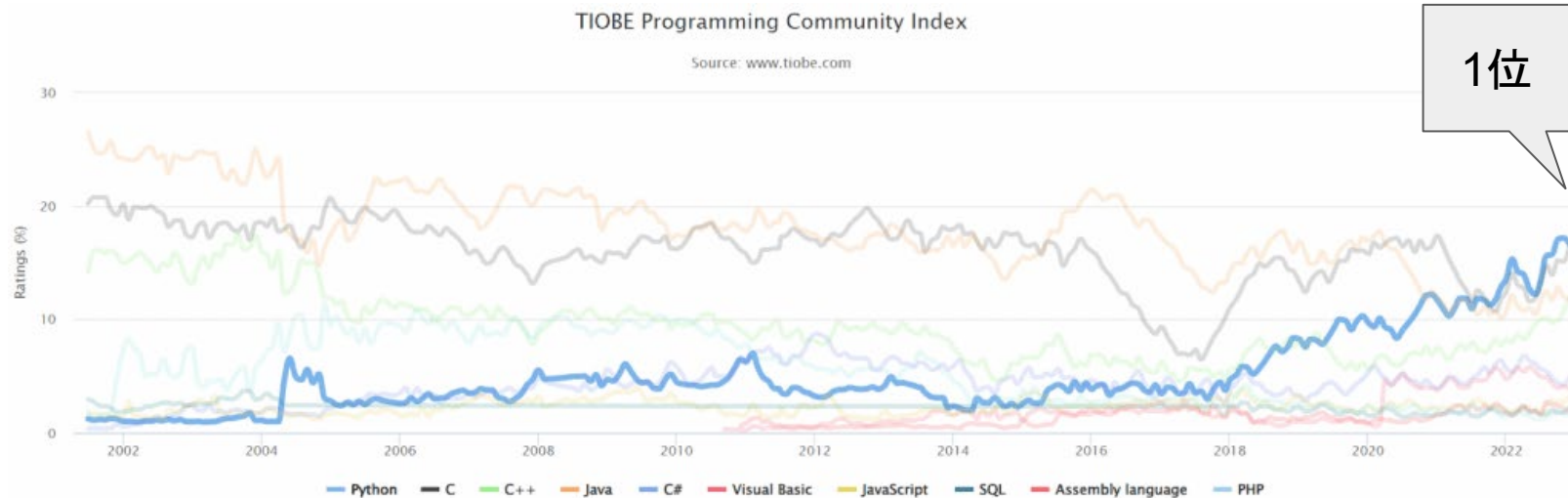
選挙の投票行動に
年代間で差はある
か....？

SNS広告とテレビ
広告が消費者に与
える影響に差はあ
るか....？



Python software foundation, (<https://www.python.org/community/logos/>)
Wikipedia, R言語, (<https://ja.wikipedia.org/wiki/R%E8%A8%80%E8%AA%9E>)
Wikipedia, Julia(プログラミング言語),
(https://ja.wikipedia.org/wiki/Julia_%28%E3%83%97%E3%83%AD%E3%82%B0%E3%83%A9%E3%83%9F%E3%83%B3%E3%82%B0%E8%A8%80%E8%AA%9E%29)

なぜpython? -シェア-



- 高いシェア
→勉強しやすい、調べやすい

CMC JAPAN, “【2023年版】プログラミング言語ランキング | 順位の推移とトレンド”,
(<https://cmc-japan.co.jp/blog/%E9%96%8B%E7%99%BA%E8%A8%80%E8%AA%9E%E3%83%A9%E3%83%B3%E3%82%AD%E3%83%B3%E3%82%B0/>)

なぜpython? -簡潔、便利-

NumPy, (<https://numpy.org/ja/press-kit/>)
Wikipedia, pandas, (<https://ja.wikipedia.org/wiki/Pandas>)

C++

```
#include <iostream>
using namespace std;
```

```
int main(){
    cout<<"Hello
World!"<<endl;
```

```
    return 0;
```

```
}
```

Python

```
print("Hello World")
```

どちらも“Hello World”と出力する文

ライブラリ



pandas

- numpy : 様々な科学計算など
- matplotlib : グラフ描画など
- pandas : データ処理など

⋮

- astropy : 天文学の計算用ライブラリ

他にも、機械学習や文法を修正するものまで目的に応じて様々

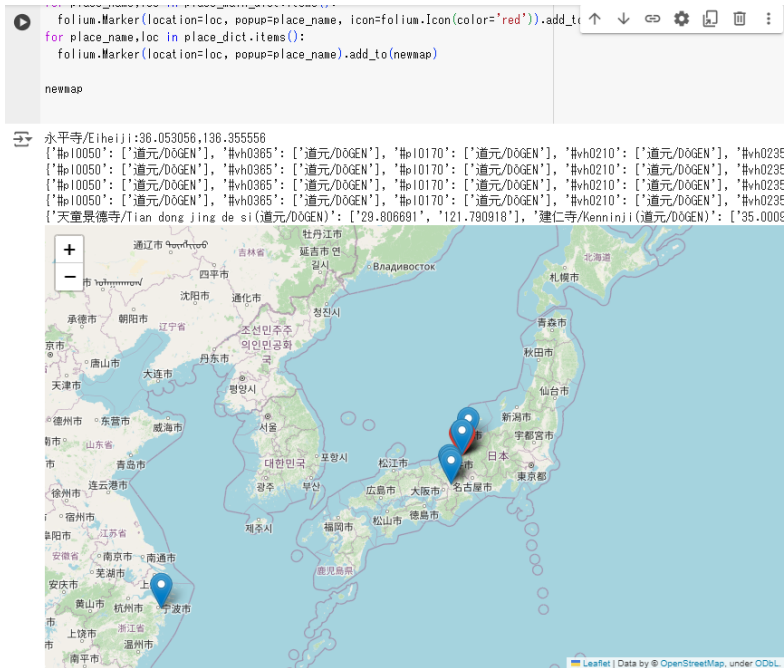
フレームワーク(必要なものがまとめられている)も豊富

Pythonの多彩な活用分野 - 応用例（人文学）

- データ分析、自然言語処理やスクレイピング。

例：

- ・シェイクスピア戯曲の文法的特徴を調べる。
(be動詞の出現回数は？など)
- ・東アジア方面の寺院や僧侶達のデータを地図上にプロット



「人文学のためのテキストデータ構築入門」フォローアップサイト，“寺院や僧侶達のデータ（主に東アジア方面）を地図上にプロット：Python/BeautifulSoup編”，
(<https://colab.research.google.com/drive/14KR1krSNzf0WbackOJaUlhwFODjTHD5?hl=ja#scrollTo=GhRBvVkkV80r&uniqifier=1>)

応用例（社会科学）

■ 業務内容

プロダクトへの経済学的な知見にもとづくアルゴリズム改善/提案など

▼例

<アドテクノロジー領域>

- ・広告のクリック率 / コンバージョン率などの予測モデル
- ・ダイナミックリターゲティング広告での商品レコメンデーション
- ・広告クリエイティブ選択アルゴリズム
- ・広告枠オークションにおける最適入札戦略
- ・実験設計

<小売DX領域>

- ・店舗内施策の因果効果検証
- ・個別因果効果の予測と入札戦略
- ・位置情報や購買データにもとづいた広告最適化

<行政DX領域>

- ・マーケットデザインの応用
- ・ナッジを用いた行動変容促進

職種 / 募集ポジション	データサイエンティスト（経済学）
雇用形態	正社員
給与	応相談 経験・能力を考慮の上、当社規定により優遇致します。
勤務地	150-6121 東京都渋谷区渋谷2丁目24番12号 渋谷スクランブルスクエア22F 地図で確認 ※屋内の受動喫煙対策 有(喫煙室あり)
経験・スキル	◎PythonやRを使った分析・モデル作成・可視化の経験 ◎機械学習の基本的な理解 ◎計量経済学および因果推論の基本的な理解 ◎事業・ビジネスを理解した上で、分析・提案ができること ◎事業課題にあった論文の調査やその論文を読解して再現出来る ◎事業課題に対して経済学の切り口からアプローチができること 【あれば尚良】 いずれかに該当する方 ○計量経済学・因果推論を用いた論文の英文査読誌での採択実績 ○機械学習・人工知能系国際カンファレンスでの採択実績 ○データ分析にもとづいたアルゴリズムの改善提案 ○機械学習アルゴリズムの実装経験

データ分析にむけて

Pythonの動作

Numpy, pandas, Matplotlib, scikit-learnなどデータ処理の基本的なライブラリを解説

・寺田学・辻真吾・鈴木たかのり・福島真太郎 (2018) 『Pythonによるあたらしいデータ分析の教科書』 翔泳社

理工図書館所蔵あり

電子ブック所蔵あり

データ分析の基礎

・安井翔太・株式会社ホクソエム (2020) 『効果検証入門～正しい比較のための因果推論/計量経済学の基礎』 技術評論社

総合図書館所蔵あり

理工図書館所蔵あり

応用例（その他）

- web開発
- ゲーム開発
- 自動化
- 画像認識 ...

例：

- Dropbox：デスクトップ用のアプリケーションや通信先のサーバーなどでPythonが使われている。
- Instagram：サーバーでPythonが用いられている。
- バトルフィールド2：大ヒットシューティングゲーム。Pythonで作られている。

環境構築

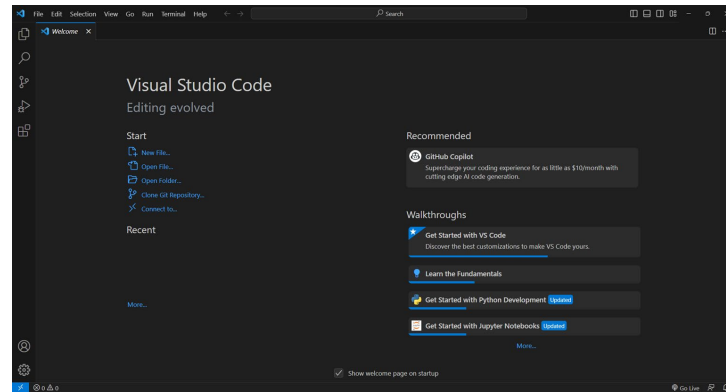
基本的にはエディターや統合的開発環境(IDE)を導入する。

- シンタックスハイライト（色分け表示）や自動補完、エラーの可視化など
- 自分で環境をカスタマイズでき、快適なコーディングに
- プロジェクトを管理しやすい

Visual Studio Code
などが主流。



今回は、ブラウザ上で動く開発環境の
Google Collaboratoryを使用。



環境構築

Google Collaboratoryを立ち上げる。

- ブラウザからGoogle Collaboratory(<https://colab.research.google.com/?hl=ja>)にアクセス
- Google アカウントにログイン(既にログインしていれば問題ない)



2. 基礎知識と構文 | 内容と目的

■内容

- ①変数・データ型 ②構文 ③関数
- 一緒に手を動かしながら覚えていく

■このコーナーで学んだことはどのような場面で役立つか

- 統計分析や自動化、可視化など。
→ライブラリの活用が一般的。プログラムの読解や改良は必須

■目的

- はじめのいっぽを一緒に踏みだし、Pythonに対するハードルを下げる

①変数, データ型

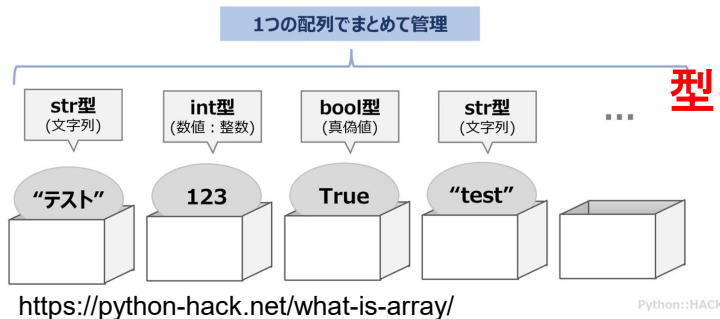
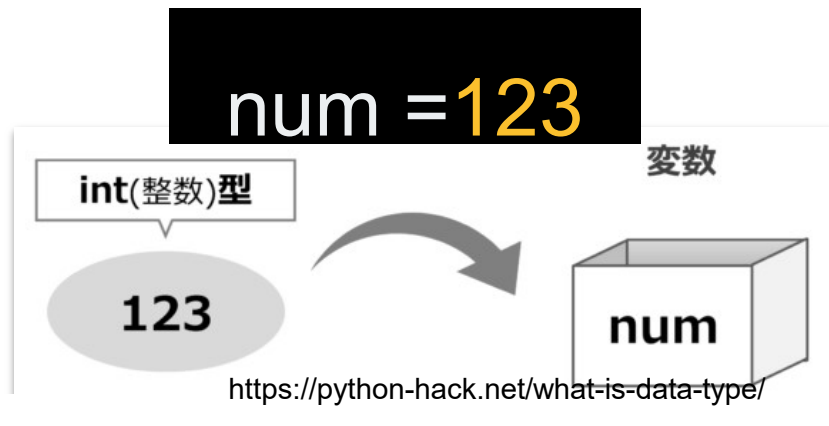
データの分類のこと。→不適切な計算の回避（例：海+1=?）

主なデータ型：

- string型 : 文字列
- int型 : 整数
- float型 : 浮動小数
- bool型 : 真偽値

これらのデータを
格納するコンテナ：

- list型
- tuple型
- dict型
- set型



型の情報も一緒に変数に格納

海+1→エラー！

①変数, データ型

主なデータ型 :

- **string型** : 文字列
- int型 : 整数
- float型 : 浮動小数
- bool型 : 真偽値

Colabの「1-a.str型」を
実行してみよう！
Shift + Enter

※変数名は自由！

■ 文字列は“”で認識される

```
a = "Hello, "  
b = "World!"
```

■ 繋ぎ合わせることできる

```
c = a+b  
print(a+b)
```

■ 型の確認

```
print(type(c))
```

①変数, データ型

主なデータ型 :

- string型 : 文字列
- **int型** : **整数**
- float型 : 浮動小数
- bool型 : 真偽値

Colabの「1-b.int型」を
実行してみよう！
Shift + Enter

■整数は直接タイプ

```
a = 10  
b = 20
```

※文字列を格納していた変数a,b
は整数に上書きされる

■四則演算

```
sum = a+b # 和  
dff = a-b # 差  
prd = a*b # 積  
div = a/b # 商  
print(sum, dff, prd, div)
```

■int型+str型

```
print(a+c)
```

■int型は整数しか表せない。divのデータ型は..?

①変数, データ型

主なデータ型 :

- string型 : 文字列
- int型 : 整数
- **float型** : **浮動小数**
- bool型 : 真偽値

Colabの「**1-c.float型**」を
実行してみよう！
Shift + Enter

■少数を直接タイプ

```
a = 1.4  
b = 1.6
```

■四則演算もできる

```
sum1 = a+b # 和  
print(sum1)
```

■整数型と演算する際には型をそろえる

```
c = 3  
sum2 = sum1 + float(c)  
print(sum2)
```

①変数, データ型

主なデータ型 :

- string型 : 文字列
- int型 : 整数
- float型 : 浮動小数
- **bool型** : **真偽値**

Colabの「1-d.bool型」を
実行してみよう！
Shift + Enter

■先頭大文字。TrueかFalseのみ

```
a = True  
b = False
```

■論理演算

```
print(a == b) # 等しい  
print(a != b) # 等しくない
```

■例) 文字列 x と y は等しいか？

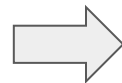
```
x = “ラーメン”  
y = “うどん”  
print(x == y)
```

②構文 | 処理を記述する際のルール

主な構文：**If文（条件分岐）**、for文（反復）

例）カフェ：ご注文はお決まりでしょうか？

- ・ メニュー内の商品の時→かしこまりました。
- ・ メニューにない商品の時→取り扱っておりません。



コンピュータでも
場合毎に処理したい



②if文 | もし○○なら××をする。

```
if 条件1:  
    処理1  
elif 条件2:  
    処理2  
else 条件1,2以外:  
    処理3
```

- if, elif, else
- 条件
 - Bool型の値を入れる。
 - 後ろにコロン「:」を書く
- 処理
 - 実行したいことを書く。
 - 先頭に空白を入れる
(基本的には2つか4つ)

②if文 | もし○○なら××をする。

```
if 条件1:  
    処理1  
elif 条件2:  
    処理2  
else 条件1,2以外:  
    処理3
```

Colabの「2-1.if文」を
実行してみよう！

Shift + Enter

■まずは簡単な例。条件が1つ

```
# パンがあるか  
hasBread = True  
  
if hasBread:  
    print("パンを食べる")  
else:  
    print("ケーキを食べる")
```

■hasBreadを「False」に変えてみよう！

②if文 | もし○○なら××をする。

※フォローします！！

■ 条件式1,2の部分を書き換えてみよう！

```
if 条件1:  
    処理1  
elif 条件2:  
    処理2  
else 条件1,2以外:  
    処理3
```

Colabの「2-1.if文」を
実行してみよう！

Shift + Enter

```
# パンがあるか  
hasBread = "Yes"  
  
if 条件式1: # hasBreadが"Yes"のとき  
    print("パンを食べる")  
elif 条件式2: # hasBreadが"No"のとき  
    print("ケーキを食べる")  
else: # それ以外  
    print("暴れる")
```

②if文 | もし○○なら××をする。

```
if 条件1:  
    処理1  
elif 条件2:  
    処理2  
else 条件1,2以外:  
    処理3
```

Colabの「2-1.if文」を
実行してみよう！

Shift + Enter

■ 解答

```
# パンがあるか  
hasBread = "No"  
  
if hasBread == "Yes": # hasBreadが"Yes"のとき  
    print("パンを食べる")  
elif hasBread == "No": # hasBreadが"No"のとき  
    print("ケーキを食べる")  
else: # それ以外  
    print("暴れる")
```

②for文 | 同じ処理を繰り返す。

主な構文：If文（条件分岐）、**for文（反復）**

例）カフェ：ご注文はお決まりでしょうか？

- ・ メニュー内の商品の時→かしこまりました。
- ・ メニューにない商品の時→取り扱っておりません。

×お客様の数



②for文 | 同じ処理を繰り返す。

```
for 変数 in range(反復回数):  
    処理
```

```
for 変数 in 配列:  
    処理
```

- 処理
 - 実行したいことを書く。
 - 先頭に空白を入れる
(基本的には2つか4つ)

②for文 | 同じ処理を繰り返す。

```
for 変数 in range(反復回数):  
    処理
```

Colabの「2-2.for文」を
実行してみよう！

Shift + Enter

■ 回数を指定して繰り返す

```
for i in range(10):  
    print(i)
```

■ 回数の調整

```
for i in range(5,10,2):  
    print(i)
```

■ ループから抜ける

```
for i in range(20):  
    print(i)  
    if i>10:  
        print("Stop Process")  
        break
```

■ 10,13,16,,,25まで出力する プログラムを書いてみよう！

②for文 | 同じ処理を繰り返す。

for 変数 in 配列:
処理

Colabの「2-2.for文」を
実行してみよう！

Shift + Enter

※配列...複数の値を格納した変数

```
array = ["りんご","ゴリラ","ラッパ"]
```

■ 配列の中身をすべて出力

```
for s in array:  
    print(s)
```

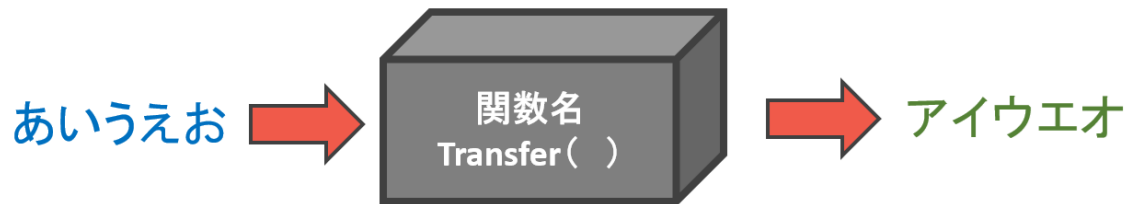
■ “りんご”なら“食べられる”を出力
してみよう！

③関数 | 機能が一つにまとまったもの。

入力された値に円周率を掛けて返す関数



入力された値をカタカナに変換して返す関数



引数を与えると値を返す。

③関数 | 機能が一つにまとまったもの。

■ 最初の問題を関数化してみる

```
def 関数名(引数):  
    処理
```

Colabの「3.関数」を
実行してみよう！

Shift + Enter

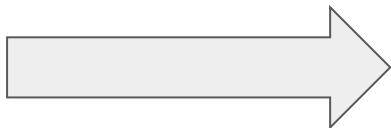
```
# パンがあるか  
hasBread = "No"  
def pan_or_cake(hasBread):  
    if hasBread == "Yes": # hasBreadが"Yes"のとき  
        print("パンを食べる")  
    elif hasBread == "No": # hasBreadが"No"のとき  
        print("ケーキを食べる")  
    else: # それ以外  
        print("暴れる")
```

```
pan_or_cake(hasBread)
```

3. 生成 AI の活用

Pythonのデータ分析ライブラリ (pandas)

	A	B	C
1	ID	Score	Grade
2	1001	89	A
3	1002	86	A
4	1003	75	B
5	1004	68	C
6	1005	88	A
7	1006	93	S
8	1007	48	F
9	1008	86	A
10	1009	78	B
11	1010	61	C
12	1011	67	C
13	1012	64	C
14	1013	72	B
15	1014	60	C
16	1015	68	C
17	1016	59	F
18	1017	84	A
19	1018	99	S



```
import pandas as pd # データ分析ライブラリを使用
```

```
data = pd.read_excel("grade_data.xlsx") # Excelファイルを読み込む
```

```
data.head() # 先頭の行列を確認
```

	ID	Score	Grade
0	1001	89	A
1	1002	86	A
2	1003	75	B
3	1004	68	C
4	1005	88	A

```
data["Score"].max() # 最大値
```

99

```
data["Score"].min() # 最小値
```

48

```
data["Score"].mean().round(2) # 平均値 (小数第2位まで)
```

np.float64(74.24)

```
data["Score"].median() # 中央値
```

72.0

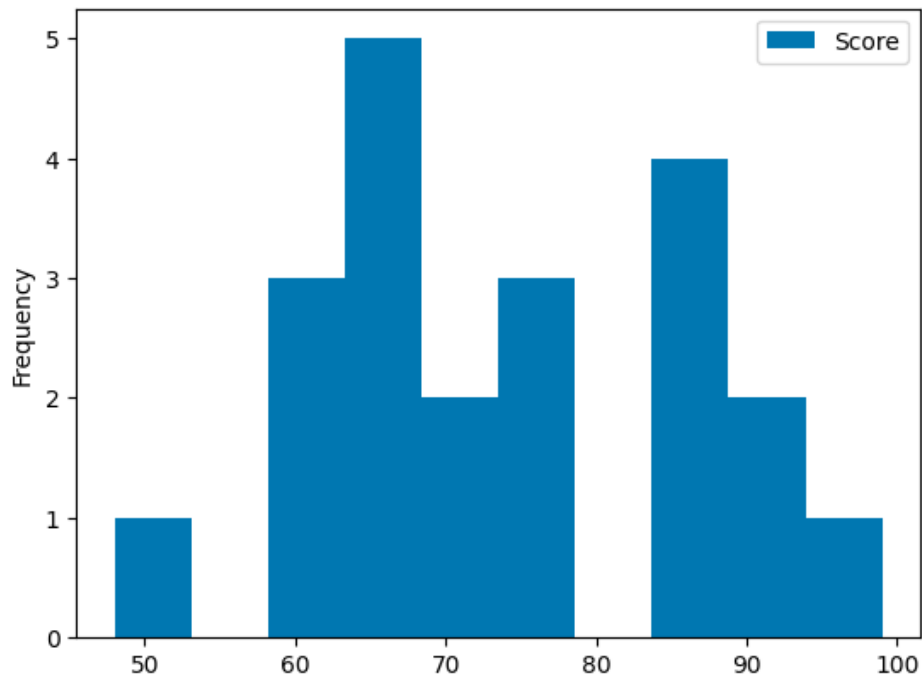
grade_data.xlsx

データ分布の可視化・解析

	A	B	C
1	ID	Score	Grade
2	1001	89	A
3	1002	86	A
4	1003	75	B
5	1004	68	C
6	1005	88	A
7	1006	93	S
8	1007	48	F
9	1008	86	A
10	1009	78	B
11	1010	61	C
12	1011	67	C
13	1012	64	C
14	1013	72	B
15	1014	60	C
16	1015	68	C
17	1016	59	F
18	1017	84	A
19	1018	99	S
20	1019	71	B
21	1020	66	C
22	1021	77	B

```
data.plot.hist(column=["Score"]) # 分布図 (ヒストグラム)
```

```
<Axes: ylabel='Frequency'>
```



```
data["Grade"].value_counts()  
# ユニークな要素の出現回数をカウント
```

	count
Grade	
C	7
A	5
B	5
S	2
F	2

dtype: int64

生成 AI: 大規模言語モデル(LLM)

- ChatGPT (コード生成、オンラインで実行)
- Claude (コード生成)
- Gemini (コード生成)



Pythonの基礎を勉強

有用なPythonの基礎を説明してください。



1. 変数とデータ型

Pythonは動的型付け言語であり、変数に値を代入するときに型を指定する必要はありません。いくつかの基本的なデータ型には、次のものがあります：

- 整数型 (int): `x = 5`
- 浮動小数点数 (float): `y = 3.14`
- 文字列 (str): `name = "Python"`
- ブール値 (bool): `flag = True`

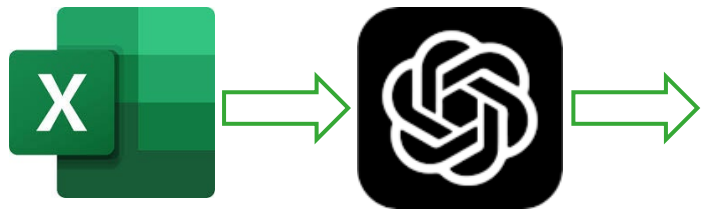
例:

python

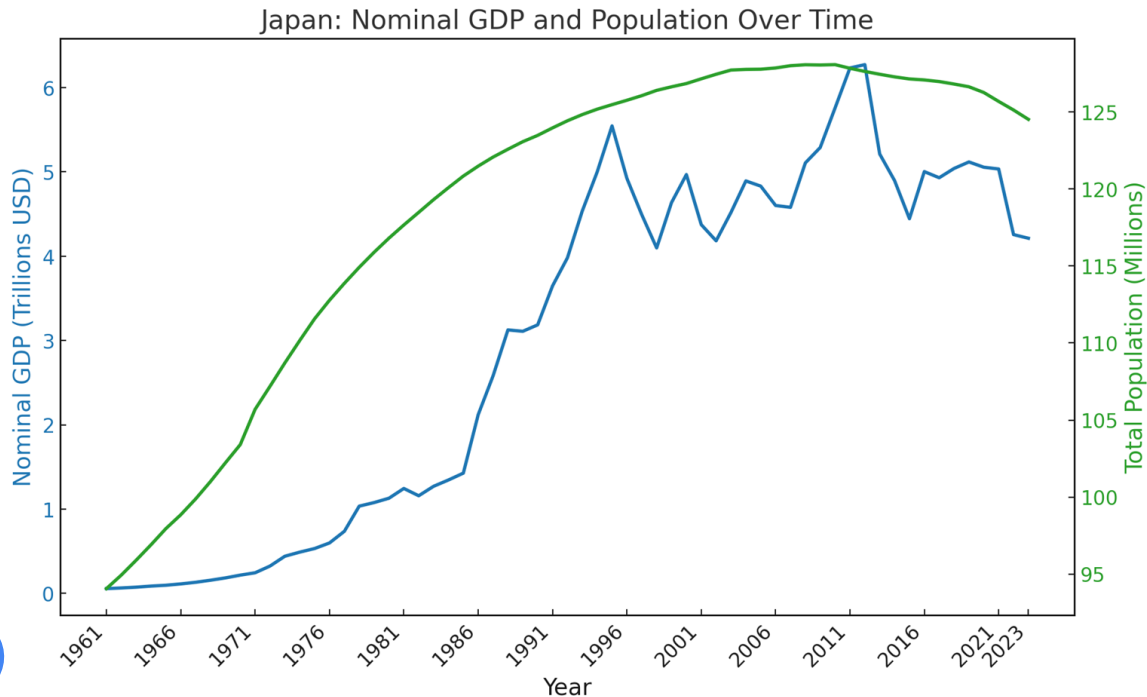
コードをコピーする

```
a = 10      # 整数
b = 3.14    # 浮動小数点数
c = "Hello" # 文字列
d = True    # ブール値
```

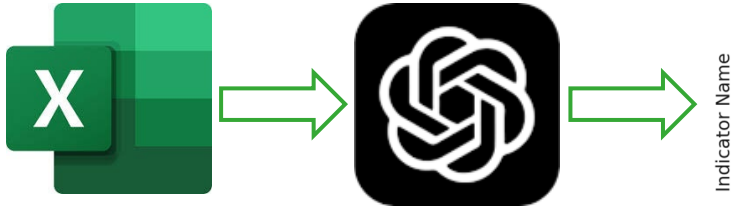
2つのy軸のグラフ



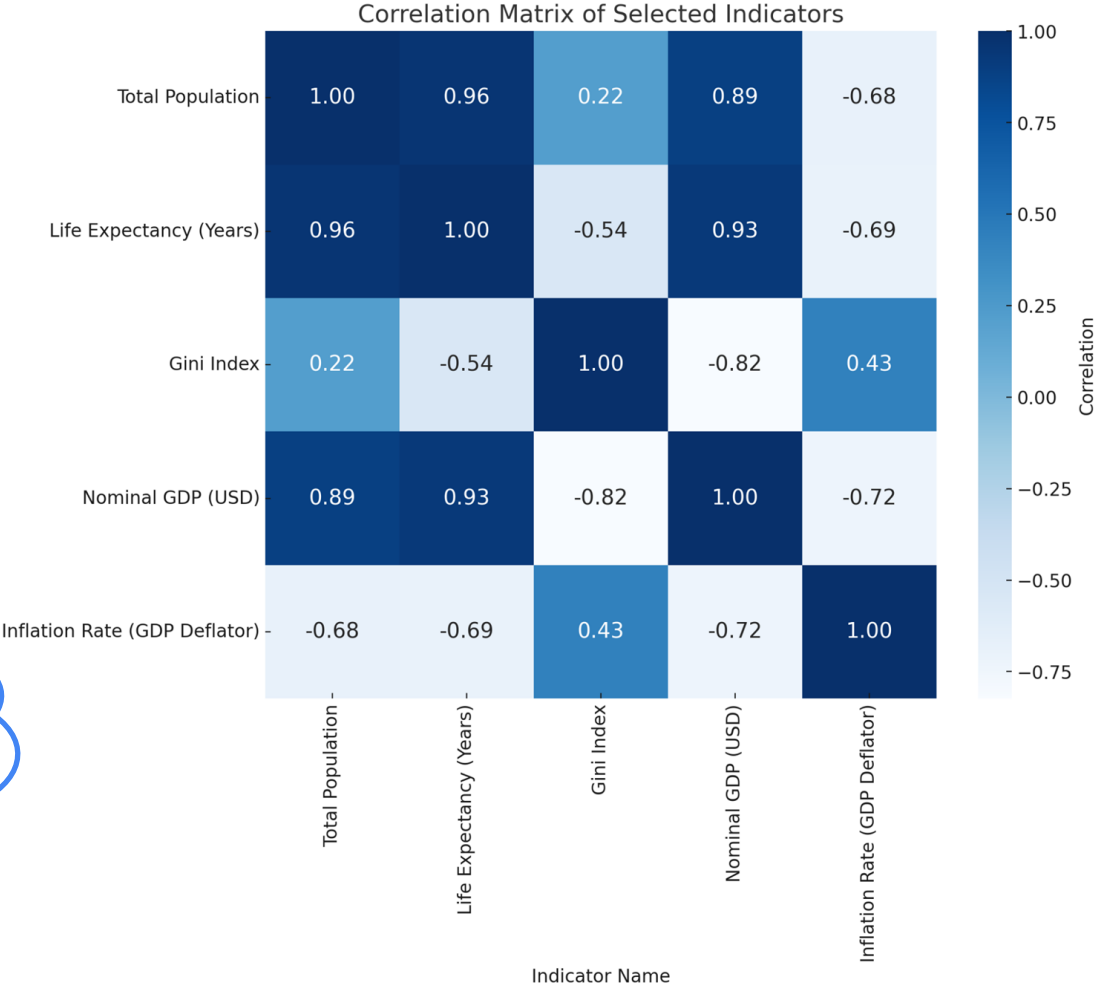
日本の人口と名目
GDPのグラフを作りたい。
x軸は時間に。



相関行列



日本に関するデータを
分析したい。相関行列
を作ってください。



追加質問

線を太くしてほしい。

フォントサイズが小さい。



グラフが正しくない。
x軸は時間にすべき。

データの特徴を説明
して。

```
print("Hello World!")
```

SyntaxError: incomplete input

実行した時エラーが出た。