

2022年度・鼓山塾

# K-15 デジタル社会の技術（プログラミング） 2. リスト・数式プロット

伊東栄典

九州大学情報基盤研究開発センター  
ito.eisuke.523@m.kyushu-u.ac.jp

# 参考資料



Python ゼロからは始めるプログラミング

著者 : 三谷純

出版社 : 翔泳社

発売日 : 2021/5/24

ISBN : 9784798169460

講義用のスライドも提供

[https://mitani.cs.tsukuba.ac.jp/book\\_support/python/](https://mitani.cs.tsukuba.ac.jp/book_support/python/)

本資料も、上記のスライドを援用しています。

K-15 デジタル社会の技術（プログラミング）

## 2. リスト・数式プロット

### 1. リスト

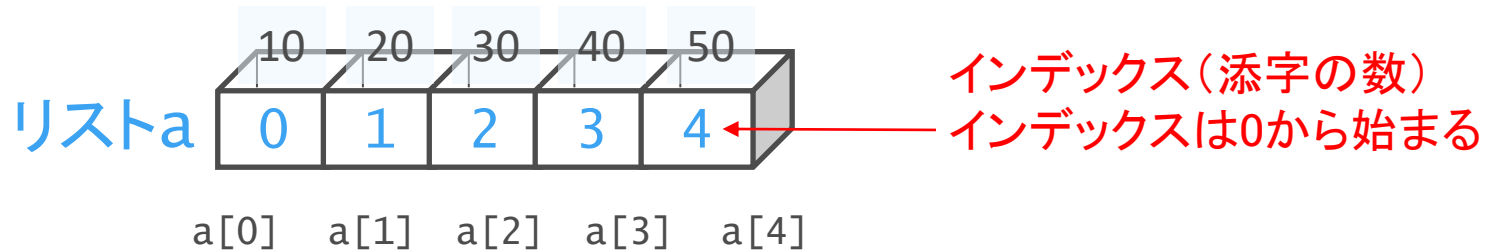
---

1. リスト
2. モジュールの利用
3. グラフ描画

# 1. リスト (List)

- 変数が連なっている構造。配列 (Array) とも言う。
- リストを使って、複数の値をまとめて管理できる
- リストの要素の値は**インデックス (添字の数)** を用いて参照する

```
a = [10, 20, 30, 40, 50]
```



```
▶ a = [10, 20, 30, 40, 50]
  print( a[1] )
```

```
↳ 20
```

a[1] が持つ値の 20 が表示される。

# リスト内の値の変更

- インデックスを指定して値を変更できる

```
▶ a = [10, 20, 30, 40, 50]
  print(a)
```

[10, 20, 30, 40, 50]

リストの全要素を出力

```
▶ a = [10, 20, 30, 40, 50]
  a[0] = 99
  print(a)
```

[99, 20, 30, 40, 50]

先頭の要素を 99 に変更  
(0 番目の要素を 99 に変更)

```
▶ a = [10, 20, 30, 40, 50]
  a[0] = 99
  a[4] = 'Hello'
  print(a)
```

[99, 20, 30, 40, 'Hello']

要素の値を文字列にもできる

## リストの要素数の確認

- len 関数でリストの要素数を取得できる

```
▶ a = [10, 20, 30, 40, 50]
  print(len(a))
```

5

len は、長さを意味する Length の略。

## 問題 1

- 1月から12月までの、各月の日数を格納したリスト `days` を作れ。
  - リストは 0 から始まるので、0番目の値は0日とする。
  - 残りの1~12要素に、各月の日数を入れる
- `input`文で「何月の日数を知りたい？」を尋ね、入力された数字に対応する月の日数を表示するプログラムを作成せよ。

## 問題 1 (解答)

- 1月から12月までの、各月の日数を格納したリスト days を作れ。
  - リストは 0 から始まるので、0番目の値は0日とする。
  - 残りの1~12要素に、各月の日数を入れる
- input文で「何月の日数を知りたい？」を尋ね、入力された数字に対応する月の日数を表示するプログラムを作成せよ。

```
days = [0, 31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31]
x = int(input('何月の日数を知りたい?:'))
print(f'{x}月の日数は{days[x]}日です。')
```



K-15 デジタル社会の技術（プログラミング）

## 2. リスト・数式プロット

## 2. モジュールの利用

---

1. はじめの1歩
2. 変数
3. 型と算術演算子
4. 文字列
5. リスト
6. モジュールの利用

## 6. モジュールの利用

- モジュールとは各種の機能を管理する単位
  - (他の言語ではライブラリ (library) と呼ぶこともある)
- 必要に応じてモジュールを読み込んでプログラムを作る。
- 読み込むことを「モジュールを**インポート**する」という。

```
import モジュール名
```

## 6.1 mathモジュール

Mathematics (数学)のモジュールなので, math

- $\sin$  や  $\cos$  などの数学の関数を使用できる。
- mathモジュールをインポート

```
import math
```

- mathモジュールに含まれる関数の利用

```
math.関数名(引数)
```



```
import math
print(math.sqrt(2))
print(math.floor(12.345))
```

```
1.4142135623730951
12
```

mathモジュールをインポート

2の平方根(square root)を計算して出力

12.345以下の最大整数を計算して出力

## mathモジュールに含まれる関数・定数

関数	説明
<code>ceil(x)</code>	xの値以上の最小整数を返す
<code>cos(x)</code>	xの余弦(cosin)を返す。xの単位はラジアン。
<code>floor(x)</code>	xの値以下の最大整数を返す
<code>exp(x)</code>	e (自然対数の底, ネイピア数) のx乗を返す。
<code>log(x)</code>	xの自然対数を返す
<code>sqrt(x)</code>	xの平方根を返す
<code>sin(x)</code>	xの正弦(sin)を返す。xの単位はラジアン。
<code>tan(x)</code>	xの正接(tangent)を返す。xの単位はラジアン。
<code>radians(x)</code>	角度xをラジアンに変換した値を返す
<code>atan(x)</code>	xの逆正接(arctangent)を返す

定数名	説明
<code>pi</code>	円周率 3.141592653589793
<code>e</code>	ネイピア数 2.718281828459045

# mathモジュールの利用例：定数

## mathモジュールに含まれる定数の利用

math.定数名

```
▶ import math
print(math.pi)
print(math.e)

3.141592653589793
2.718281828459045
```

mathモジュールをインポート

円周率 ( $\pi$ ) を出力

ネイピア数 (e) を出力

## 6.2 randomモジュールの利用

- 乱数を生成するモジュール。何かと便利。

### random モジュールに含まれる関数

関数	説明
random()	0以上1未満の浮動小数点数を返す。
randrange(x)	0から (x-1) までの整数を返す。
choice(list)	list からランダムに要素を1つ選び, その値を返す。
randint(a, b)	a 以上 b 以下の整数をランダムに返す。

# randomモジュールの利用例

```
import random
random.randint(1,6)
4
```

randomモジュールをインポート

1以上, 6未満の整数をランダムに返す

実行のたびに値が変わる

```
import random
janken = ['グー', 'チョキ', 'パー']
random.choice(janken)
'パー'
```

randomモジュールをインポート  
(既にimportしていれば記述不要)

要素を3つ持つリスト

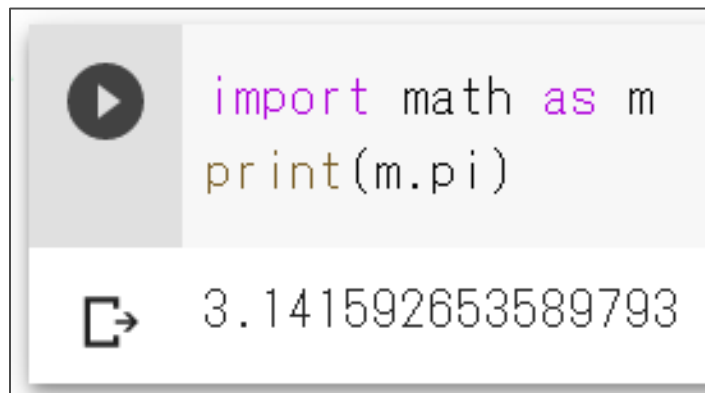
リスト janken が持つ要素からランダム  
に1つ選び, その要素の値を返す

実行のたびに値が変わる

## 6.3 モジュールに別名をつけて使う

モジュール名が長いときは, 別名をつけると便利

```
import モジュール名 as 別名
```



```
▶ import math as m  
print(m.pi)
```

```
↳ 3.141592653589793
```

← mathモジュールの別名をmとする

← 別名を使って記述



# モジュールに含まれる関数を調べる

- Pythonの標準ライブラリのドキュメント

<https://docs.python.org/ja/3/library/index.html>



The screenshot shows the Python 3.9.4 documentation page for the standard library index. The page title is "Python 標準ライブラリ". The main content area contains the following text:

Python 言語リファレンス ではプログラミング言語 Python の厳密な構文とセマンティクスについて説明されていますが、このライブラリリファレンスマニュアルでは Python とともに配付されている標準ライブラリについて説明します。また Python 配布物に収められていることの多いオプションのコンポーネントについても説明します。

Python の標準ライブラリはとても拡張性があり、下の長い目次のリストで判るように幅広いものを用意しています。このライブラリには、例えばファイル I/O のように、Python プログラマが直接アクセスできないシステム機能へのアクセス機能を提供する (C で書かれた) 組み込みモジュールや、日々のプログラミングで生じる多くの問題に標準的な解決策を提供する Python で書かれたモジュールが入っています。これら数多くのモジュールには、プラットフォーム固有の事情をプラットフォーム独立な API へと昇華させることにより、Python プログラムに移植性を持たせ、それを高めるという明確な意図があります。

Windows 向けの Python インストーラはたいがい標準ライブラリのすべてを含み、しばしばそれ以外の追加のコンポーネントも含んでいます。Unix 系のオペレーティングシステムの場合は Python は一揃いのパッケージとして提供されるのが普通で、オプションのコンポーネントを手に入れるにはオペレーティングシステムのパッケージツールを使うことになるでしょう。

標準ライブラリに加えて、数千のコンポーネントが (独立したプログラムやモジュールからパッケージ、アプリケーション開発フレームワークまで) 成長し続けるコレクションとして [Python Package Index](#) から入手可能です。

• [はじめに](#)

- [利用可能性について](#)

• [組み込み関数](#)

• [組み込み定数](#)

- [site](#) モジュールで追加される定数

• [組み込み型](#)

- [真理値判定](#)
- [ブール演算](#) --- `and`, `or`, `not`
- [比較](#)
- [数値型](#) `int`, `float`, `complex`

# モジュールに含まれる関数を調べる

- 「標準ライブラリ」のページ右上の「モジュール」のリンクから「math」を探してみよう。
- ブラウザの「検索」機能も使ってみよう
  - Windows, Chrome book : [Ctrl]+[F]
  - Mac : [コマンド]+[f]

## math --- 数学関数

このモジュールは、C 標準で定義された数学関数へのアクセスを提供します。

これらの関数で複素数を使うことはできません。複素数に対応する必要があるならば、`cmath` モジュールにある同じ名前の関数を使ってください。ほとんどのユーザーは複素数を理解するのに必要なだけの数学を勉強したくないので、複素数に対応した関数と対応していない関数の区別がされています。これらの関数では複素数が利用できないため、引数に複素数を渡されると、複素数の結果が返るのではなく例外が発生します。その結果、こういった理由で例外が送出されたかに早い段階で気づく事ができます。

このモジュールでは次の関数を提供しています。明示的な注記のない限り、戻り値は全て浮動小数点数になります。

### 数論および数表現の関数

`math.ceil(x)`  
 $x$  の「天井」 ( $x$  以上の最小の整数) を返します。  $x$  が浮動小数点数でなければ、内部的に `math.ceil()` が実行され、 `Integral` 値が返されます。

`math.comb(n, k)`  
Return the number of ways to choose  $k$  items from  $n$  items without repetition and without order.  
Evaluates to  $n! / (k! * (n - k)!)$  when  $k \leq n$  and evaluates to zero when  $k > n$ .  
Also called the binomial coefficient because it is equivalent to the coefficient of  $k$ -th term in polynomial expansion of the expression  $(1 + x) ** n$ .  
Raises `TypeError` if either of the arguments are not integers. Raises `ValueError` if either of the arguments are negative.  
バージョン 3.8 で追加。

`math.copysign(x, y)`  
 $x$  の大きさ (絶対値) で  $y$  と同じ符号の浮動小数点数を返します。符号付きのゼロをサポートしているプラットフォームでは、`copysign(1.0, -0.0)` は `-1.0` を返します。

`math.fabs(x)`

## 問題 2

- mathモジュールを利用して、角度  $120^\circ$  のコサイン (cos) 値を求めてください。

## 問題 2 (解答)

- mathモジュールを利用して, 角度  $120^\circ$  のコサイン (cos) 値を求めてください。

```
import math
print(math.cos(math.radians(120)))
```

※ 別解

```
import math
print(math.cos(2 / 3 * math.pi))
```

K-15 デジタル社会の技術（プログラミング）

## 2. リスト・数式プロット

# 3. グラフ描画

---

1. リスト
2. モジュールの利用
3. グラフ描画

### 3. グラフ描画

- Google Colab での Python プログラムでは、数学的なグラフ描画も可能。
- matplotlib モジュールを使う
  - Mathematical (数学的) な plot (描画) をするもの。
- importして別名で使うことが多い

```
import matplotlib.pyplot as plt
```

matplotlibモジュールの「pyplot」だけをインポート。  
それを別名 plt として使う。

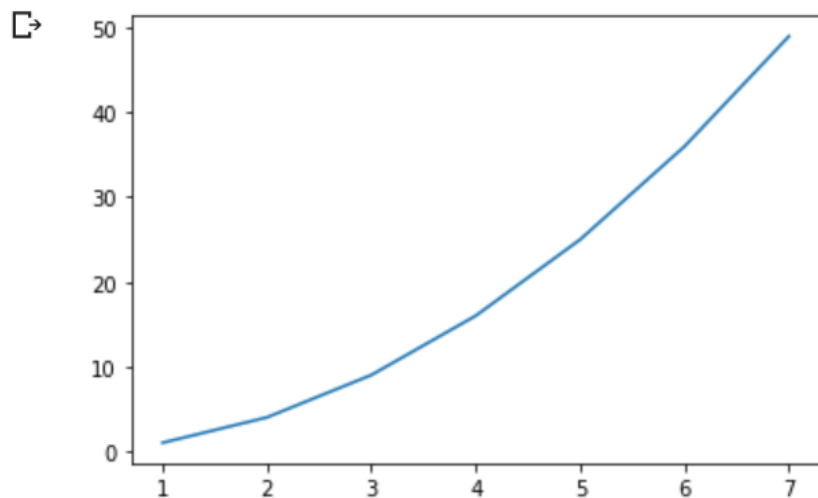
## 簡単なグラフ描画(1)

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5, 6, 7]
y = [1, 4, 9, 16, 25, 36, 49]
plt.plot(x, y)
plt.show()
```

リストでデータを用意

xを横軸, yを縦軸としてデータをプロット  
グラフを表示

```
import matplotlib.pyplot as plt
x = [1, 2, 3, 4, 5, 6, 7]
y = [1, 4, 9, 16, 25, 36, 49]
plt.plot(x, y)
plt.show()
```

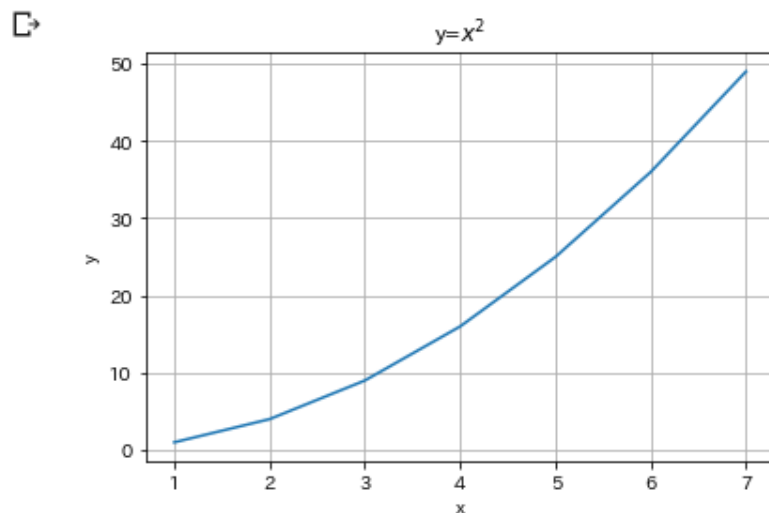


## 簡単なグラフ描画(2)

```
import matplotlib.pyplot as plt
x = [1,2,3,4,5,6,7]
y = [1,4,9,16,25,36,49]
plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.title("y=$x^2$")
plt.grid()
plt.show()
```

x軸のラベル  
y軸のラベル  
グラフのタイトル  
グリッドを表示

```
import matplotlib.pyplot as plt
x = [1,2,3,4,5,6,7]
y = [1,4,9,16,25,36,49]
plt.plot(x, y)
plt.xlabel("x")
plt.ylabel("y")
plt.title("y=$x^2$")
plt.grid(True)
plt.show()
```

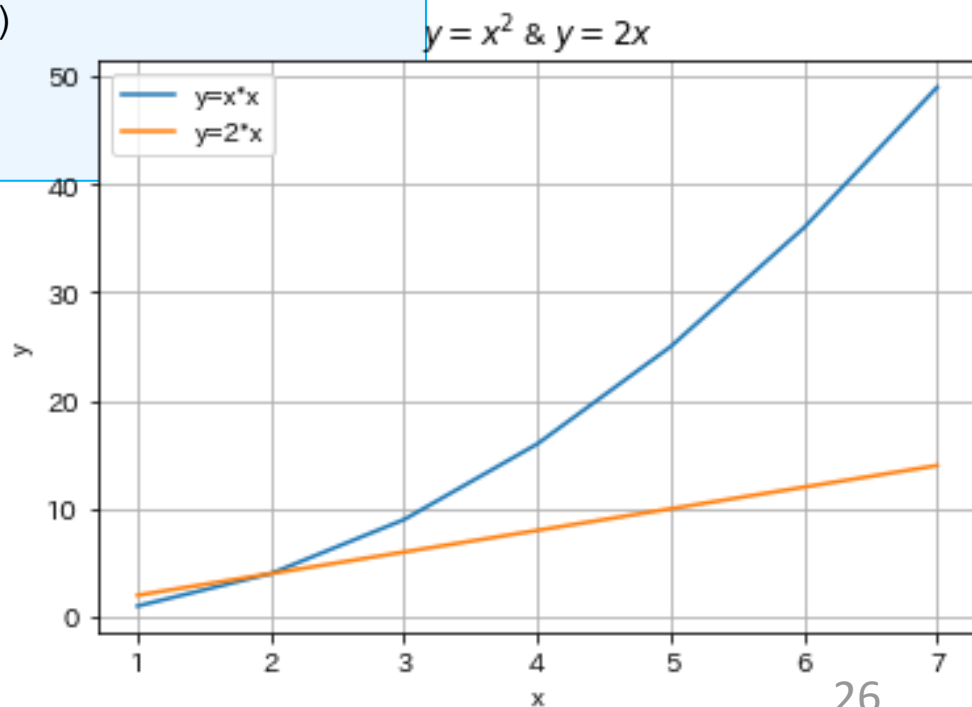




## 簡単なグラフ描画(3)

```
import matplotlib.pyplot as plt
x = [1,2,3,4,5,6,7]
y = [1,4,9,16,25,36,49]
plt.plot(x, y, label="y=x*x")
z = [2,4,6,8,10,12,14]
plt.plot(x, z, label="y=2*x")
plt.legend()
plt.xlabel("x")
plt.ylabel("y")
plt.title("$y=x^2$ & $y=2x$")
plt.grid()
plt.show()
```

プロット1と、そのラベル  
2つ目の系列  
プロット2と、そのラベル  
凡例を表示



## 簡単なグラフ描画(3)

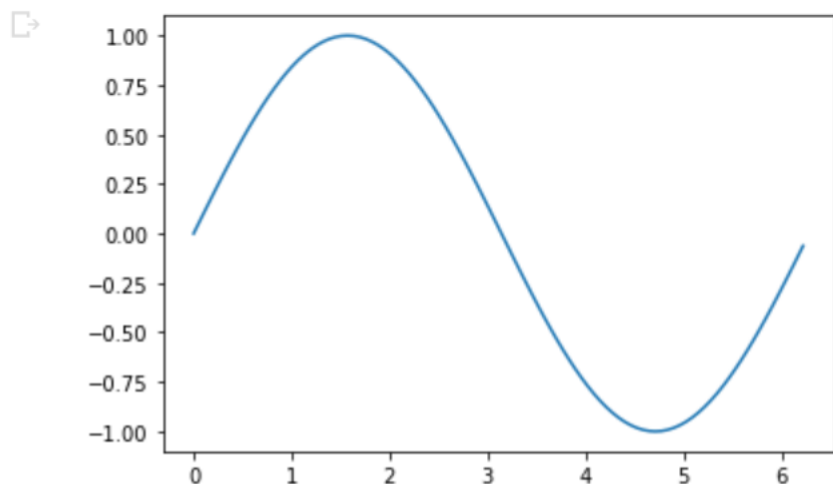
```
import matplotlib.pyplot as plt
import math as m
x = [0.02*i*m.pi for i in range(100)]
y = [m.sin(z) for z in x]
plt.plot(x, y)
plt.show()
```

mathをインポート(別名 m で使う)

0から  $0.02\pi$  ずつ増加する  
100個のデータ  
(繰り返しの資料で説明)

x の各要素に sin関数を適用  
した値のリスト

```
import matplotlib.pyplot as plt
import math as m
x = [0.02*i*m.pi for i in range(100)]
y = [m.sin(z) for z in x]
plt.plot(x, y)
plt.show()
```



## その他の機能

- 線の種類や色も変更可能
- 様々なグラフが描ける
- 詳細は配布や、解説サイトを見て、試してみよう
  - Matplotlib サイト → <https://matplotlib.org/>
  - ギャラリー <https://matplotlib.org/stable/gallery/index.html>