計算物理学B

第4回
配列（特にNumPy）

藤本 悠輝、野垣 康介
（藤本担当回）

質問等あればメールでも受け付けます：
yuki.fujimoto.phys__at__niigata-u.ac.jp
(__at__ を @ に変えてください)

講義予定

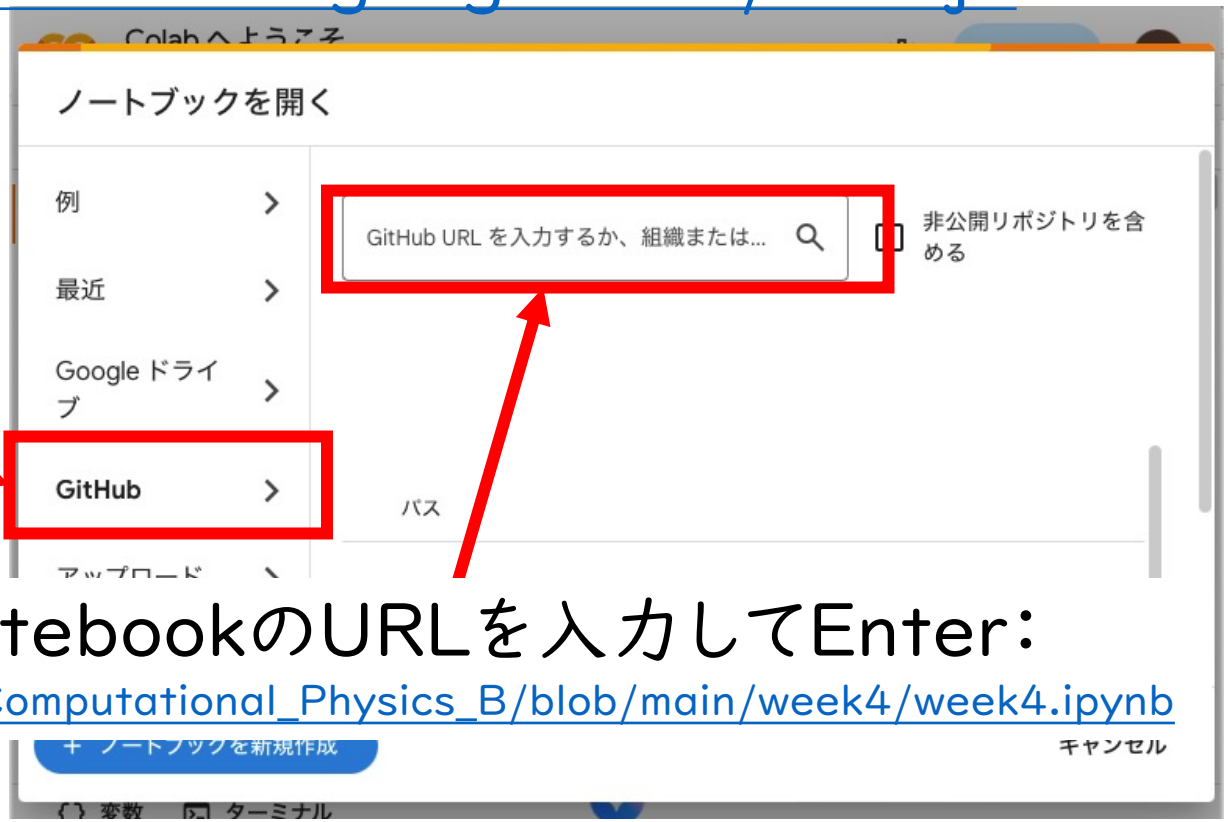
10/07 両名:四則演算
10/14 野垣:制御文(for, if)
10/21 野垣:関数
10/28 藤本:配列(numpy)
11/04 藤本:可視化(matplotlib)
11/11 野垣:数値微分
11/18 藤本:数値積分
～中間レポート～

12/09 野垣:モンテカルロ1
12/16 野垣:モンテカルロ2
12/23 藤本:微分方程式1
01/13 藤本:微分方程式2
01/20 藤本:微分方程式3
01/27 野垣:最適化
02/03 藤本:機械学習
～期末レポート～

あくまで予定なので変更の可能性あり

実習環境

まず、Google Colabを開く:
https://colab.research.google.com/?hl=ja



1. GitHubを選択

2. ここに今週のNotebookのURLを入力してEnter:
https://github.com/nogaki/Computational_Physics_B/blob/main/week4/week4.ipynb

[注意!]この状態では、GitHub上のファイルを開いただけで、
個人のドライブに保存されていません!
[ファイル]→[ドライブにコピーを保存]を実行すること。

第4回　配列（特にNumPy）

- コンテナ型（リスト・タプル・集合・辞書型）
- NumPy
- 統計・線形代数

この講義は次を参考に準備されています：
「実践計算物理学」野本拓也 、是常隆、有田亮太郎 著（共立出版）
https://numpy.org/doc/stable/user/
https://github.com/vlvovch/PHYS6350-ComputationalPhysics

コンテナ型

Pythonのデータ型には、数値(int, float)や文字列(str)のほかに、複合的な型（**コンテナ型**）が存在する

コンテナ=container, 入れ物の意

これらのうち、特に使われるのが

1. リスト
2. タプル
3. 集合
4. 辞書

1. リスト

リストは、要素をカンマで区切って、四角括弧[]で囲む：

```
[1]  ✓ 0.0s                                   Python
     x = [1, 2, 3]
     y = ["Hello", " ", "world"]
```

リストの中身は数値や文字列(str)でも何でも良い

```
[5]  ✓ 0.0s
     print(x[0], y[2])
...  1 world
```

←リストxのn番目の要素にアクセスするには x[n] とする

※Pythonのインデックスは0から始まる
つまり、0番目が最初の要素

1. リスト

リストxとyについて、いくつか便利な演算と操作：

```
x + y       ←xとyを結合
len(x)      ←xの長さを取得
2 in x      ←xが2という要素を含むか（True/False）
x[0] = "A"  ←xの0番目の要素を"A"に置き換える
x.append("X")  ←xの最後に"X"を追加(メソッド)
for a in x:    ←xの要素で繰り返し処理
    print("a=", a) この場合 a=A, a=2, a=3, a=X
                   と連続して出力される
```

あとで実際に手を動かして確認してみよう！

2. タプル tuple

tupleとは？

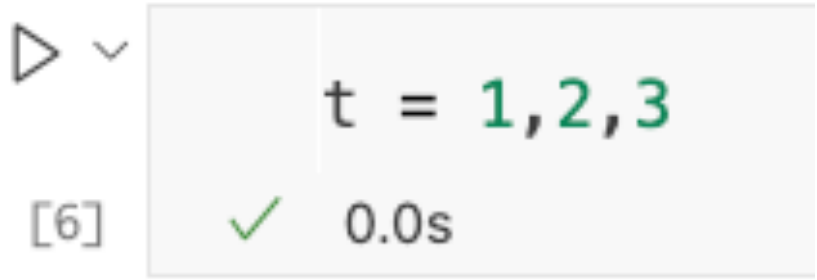
“英語におけるタプルという用語は、シングル (single)、ダブル (double)、トリプル (triple)、さらに4つ組のクアドルプル (quadruple)、5つ組のクインチュープル (quintuple) といった、類似の接尾辞をもつラテン語系の英語の並びを抽象化して n タプル (n-tuple) と呼ぶようになったことに由来する。”

Wikipediaより引用

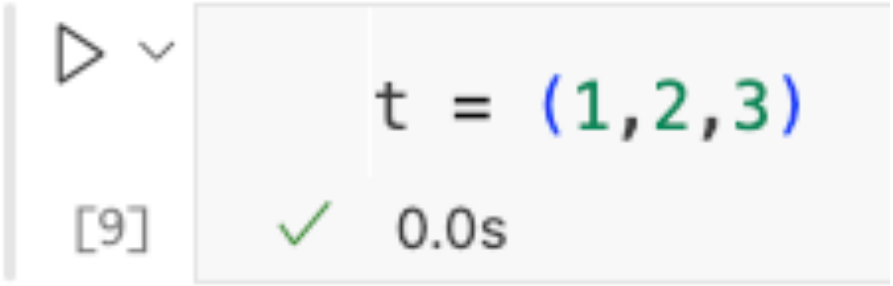
要は（数値や文字列の）組のこと

2. タプル tuple

Pythonでは、カンマで区切られた要素で表現される：



あるいは、丸括弧 ( ) で囲んでも良い：



どちらでも同じ

2. タプル tuple

一度作成されたタプルは修正できない(immutable):

```
t[0] = "A"
```

[8] ⊗ 0.9s

```
TypeError                                 Traceback (most
Cell In[8], line 1
----> 1 t[0] = "A"

TypeError: 'tuple' object does not support item assignment
```

↑タプルtの0番目の要素を置き換えようとしたが、
タプルは修正できないので、エラーを吐く

これがリストとの重要な違い

修正でなければ、再定義は可能:

```
t = "A",2,3
```

[10] ✓ 0.0s

2. タプル tuple

タプルの使い方：

```
a, b = 1, 2   #複数の変数の初期化
b, a = a, b   #変数の値の交換
```

[11] ✓ 0.0s

3. 集合 set

数学の意味での集合。次のようにカンマで区切って
波括弧 { } で囲むか、listをset()に渡すことで定義：

```
s1 = {1,2,3,3,4}
s2 = set([3,1,5,5,5,4])
```

[12] ✓ 0.0s

定義上、重複部分は取り除かれる：

```
print(s1, s2)
```

[15] ✓ 0.0s

```
{1, 2, 3, 4} {1, 3, 4, 5}
```

3. 集合 set

ビット演算子のうち、論理和 OR (|)、論理積 AND (&)、
排他的論理和 XOR (^) を、それぞれ集合論での
和 (∪)、共通部分 (∩)、対称差として使える：

```
s1 = {1, 2, 3, 4} ,   s2 = {1, 3, 4, 5}
s1 | s2 = {1, 2, 3, 4, 5}   ←s1とs2の和集合
s1 & s2 = {1, 3, 4}         ←s1とs2の共通部分
s1 ^ s2 = {2, 5}            ←s1とs2の対称差
```

使用例：listの重複部分を取り除く

4. 辞書

キー(Key)と値(Value)のペアを保持するデータ型

```
# 辞書型：{"key1":value1, ...} のように書く
teisuu = {"hbar":1.055e-34, "c":1.0}
print(teisuu["hbar"]) # Key="hbar"のValueを取得
teisuu["c"] = 2.998e+8 # Key="c"のValueを変更
teisuu["e"] = 1.602e-19 # {"e":1.602e-19}の要素を追加
for key, value in teisuu.items():
    # teisuuに適用された.itemsというメソッドは
    # keyとvalueの組を取得。反復処理に用いる
    print(key, value)
```

コードの可読性を向上させる→本格的な数値計算よりは、データ処理によく用いられる

コンテナ型のまとめ

	リスト	タプル	集合	辞書
変更	✅	❌	✅	✅
順序の保持	✅	✅	❌	❌
要素の重複	✅	✅	❌	❌
反復	✅	✅	✅	✅
要素の種類	何でも	何でも	イミュータブル	キーとバリューの組

Numpyとは？

ライブラリの一種。

Numerical Pythonの略

通常は次のようにインポートされる：

```
import numpy as np
```

※ライブラリ：モジュールやパッケージやそれらをまとめてインストールできるようにしたもの全般への総称
※モジュール：実行したい内容を.pyファイルとして保存したもの。他のPythonコードから呼び出して使う。通常、数種類のクラスや関数などのみを含み簡潔にする。ある程度長いコードを書く場合に用いられる
※パッケージ：モジュールを1つフォルダにまとめて複数束ねて管理したもの

Numpyの配列

ndarrayというクラスのオブジェクト(「**配列**」)がnumpyの肝。
このように定義する:

```
a = np.array([1,2,3])
```

2次元以上の配列(行列など)も次のように定義できる:

```
b = np.array([[1,2,3],[4,5,6]])
```

これ以外にも配列に対しての操作(ソート、線形代数などなど)の便利な機能が色々ある。
配布のノートブックで手を動かして確認してみよう

Numpy以外の便利なライブラリ

- scipy: 科学計算用。この授業ではアルゴリズムの中身を理解することを目的としているため使用しないが、実際の科学計算の場面では多用される。
- matplotlib: グラフなどの可視化用。次回扱う。
- pandas: データ解析では必須。この授業では使用しないが、知っていると便利な場面が多い。
- pytorch: 機械学習でよく使う。他にはtensorflow, keras, scikit-learn, jax, などなど

他にも色々あるので、調べてみよう!

～実習タイム～

(1) np.random.randnで正規分布に従う乱数配列を作り、その平均と分散をnp.meanとnp.varで求めよ。

(2)numpyを使わず、for文を使って平均と分散を求めよ。

(3)さらに、配列のサイズを変えたときに、その時間がどう変化するかをそれぞれの場合について比較してみよ。

※未知の関数については、たとえば
　　np.random.randn?
などと、関数の後に?をつけると、ヘルプ(英語)を参照できる。
もちろんググっても良し、AIに聞いても良し。