

# **一般教育演習**

**プログラミングで問題を解く：**

**集計から人工知能まで**

**瀧川一学**

**工学部 情報理工学コース**

# 次回からPythonを掘り下げたい

## 宿題：

- USBメモリからJupyter Notebookを起動する手順を確認復習しておくこと
- 特に毎回必要となるコマンドプロンプト作業も確認復習しておくこと
- Jupyter Notebookにいろいろ打ち込んでエラー出してみたり遊んでみること
- 余裕ある人はPythonの教材もチェック

# 自分のPCを使いたい人へ

演習ではNotebookをダウンロードして作業することがあるためネットワークの設定が必要です。下記を読んで自分のPCで北大内の無線LANが使えるよう設定をしてください。

<https://www.hines.hokudai.ac.jp/wlan/>

USBメモリは必要なく、下記の作業のみです。

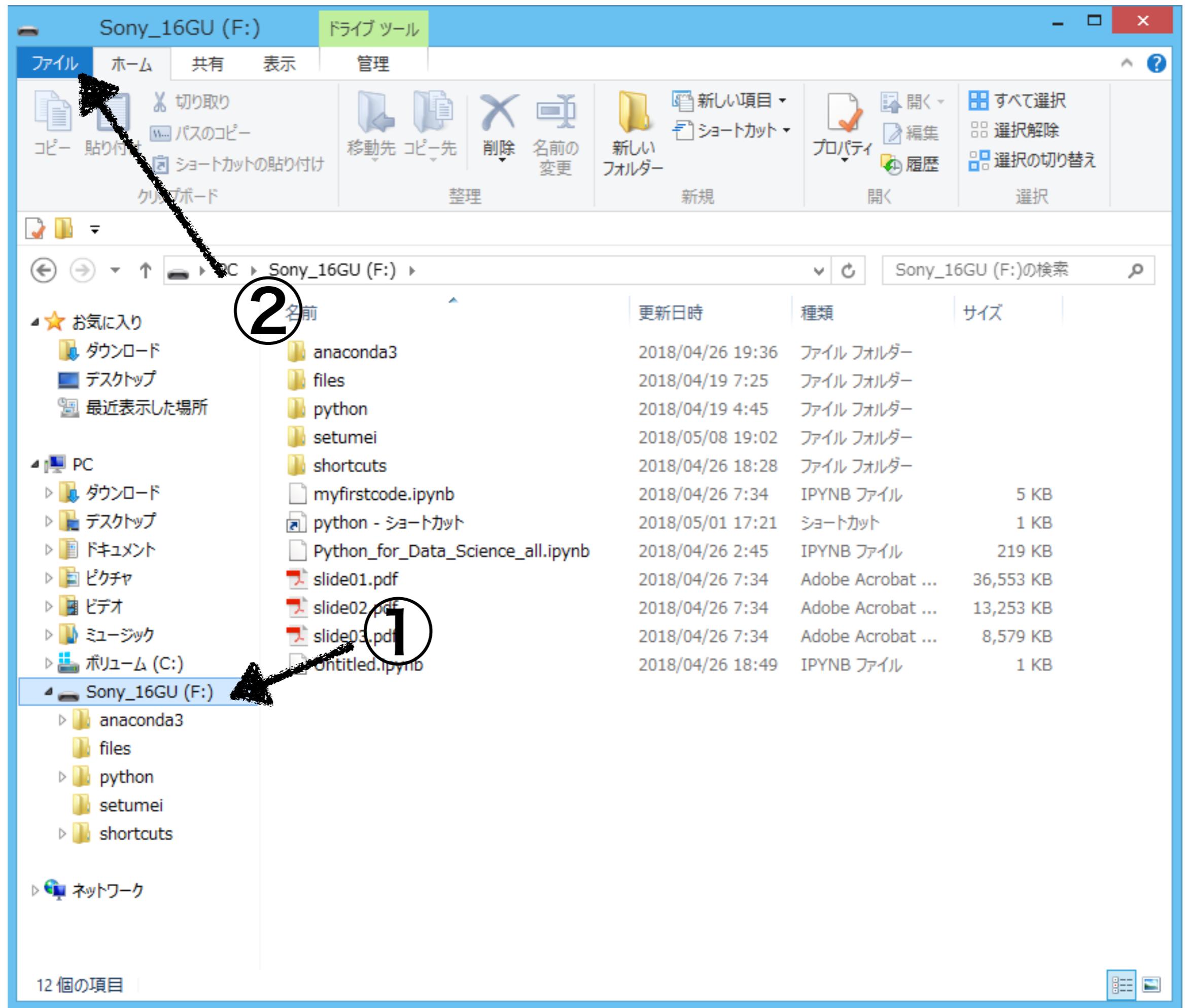
- 1) Anacondaをダウンロードしインストーラの指示通りに通常インストールします。
- 2) Jupyter Notebookの起動アイコンができるので普通にそこから起動して使うだけです。

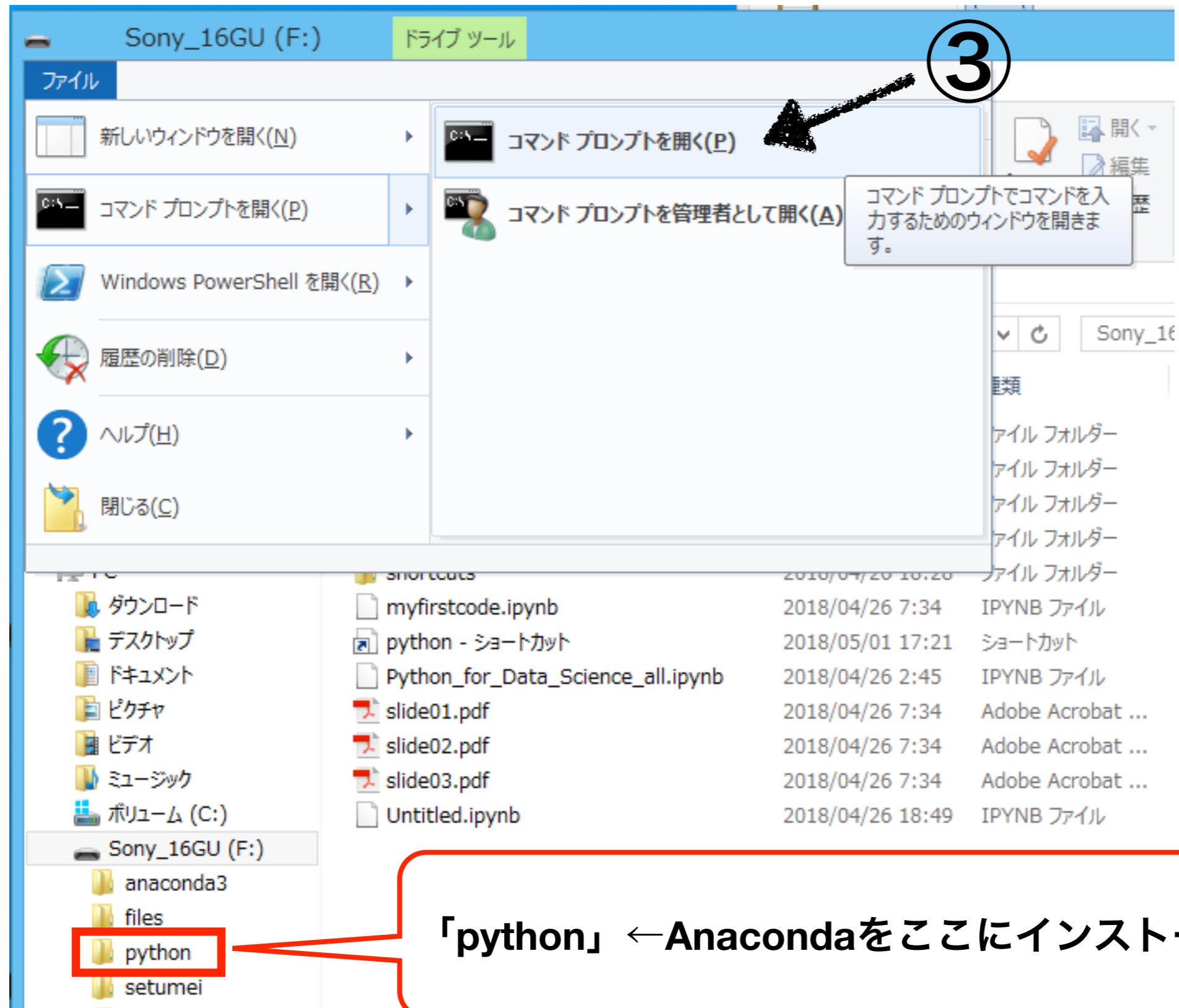
Macでも同じです。

# USBメモリのPython環境を使う：

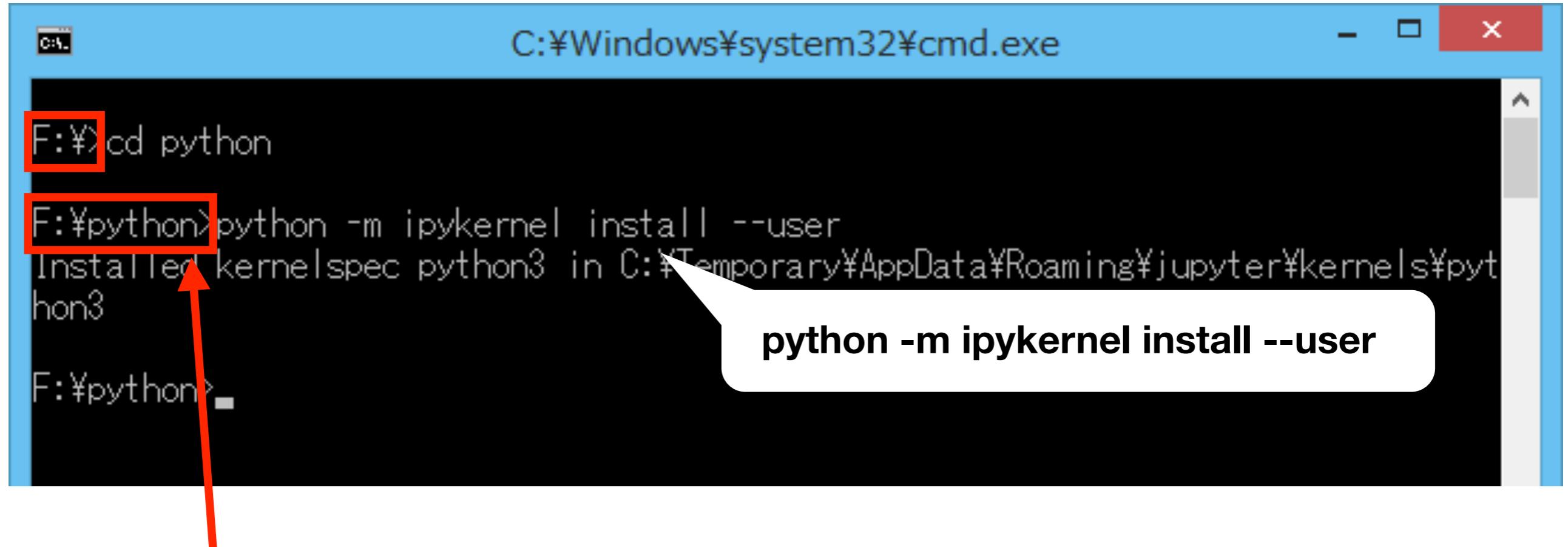
**毎回**演習開始時に以下を実行

1. USBのフォルダ「python」を選択
2. コマンドプロンプトを起動
3. 「python -m ipykernel install --user」を入力





「python」 ← Anacondaをここにインストールしたとする



C:\Windows\system32\cmd.exe

```
F:>cd python
F:>python -m ipykernel install --user
Installed kernelspec python3 in C:\Temporary\AppData\Roaming\jupyter\kernels\python3
F:>
```

A red box highlights the command `python -m ipykernel install --user`. A red arrow points from this highlighted text to a callout bubble containing the same command.

ここにどのフォルダを見ているかが表示されるので...

最初にAnacondaをインストールしたフォルダ(ここでは「**python**」という名前とする)を選択してから「コマンドプロンプト」を起動してたらここに「F:>**python**>」と出ているので、そのまま「`python -m ipykernel install --user`」と入力

「F:>」など、別であれば「`cd F:>python`」として見ているフォルダをかえ表示が「F:>**python**>」となるか見る

**ここまで毎回準備！**

# Jupyter Notebookの起動の仕方

先ほどの準備作業をした後

## その1：一番直接的なやり方

1. USBメモリのAnacondaを入れたフォルダを開く
2. そのフォルダの中にある 「**Scripts**」 を開く
3. 「**jupyter-notebook**」 を起動

(ファイル名は「jupyter-notebook.exe」だが  
たぶんexeは通常表示されない)

# Jupyter Notebookの起動の仕方

その2：ショートカットを作つておき  
それから起動

<http://art.ist.hokudai.ac.jp/~takigawa/prog/>

講義ウェブサイトから「Jupyter\_Notebook.lnk」を  
ダウンロードし、USBに保存しておく

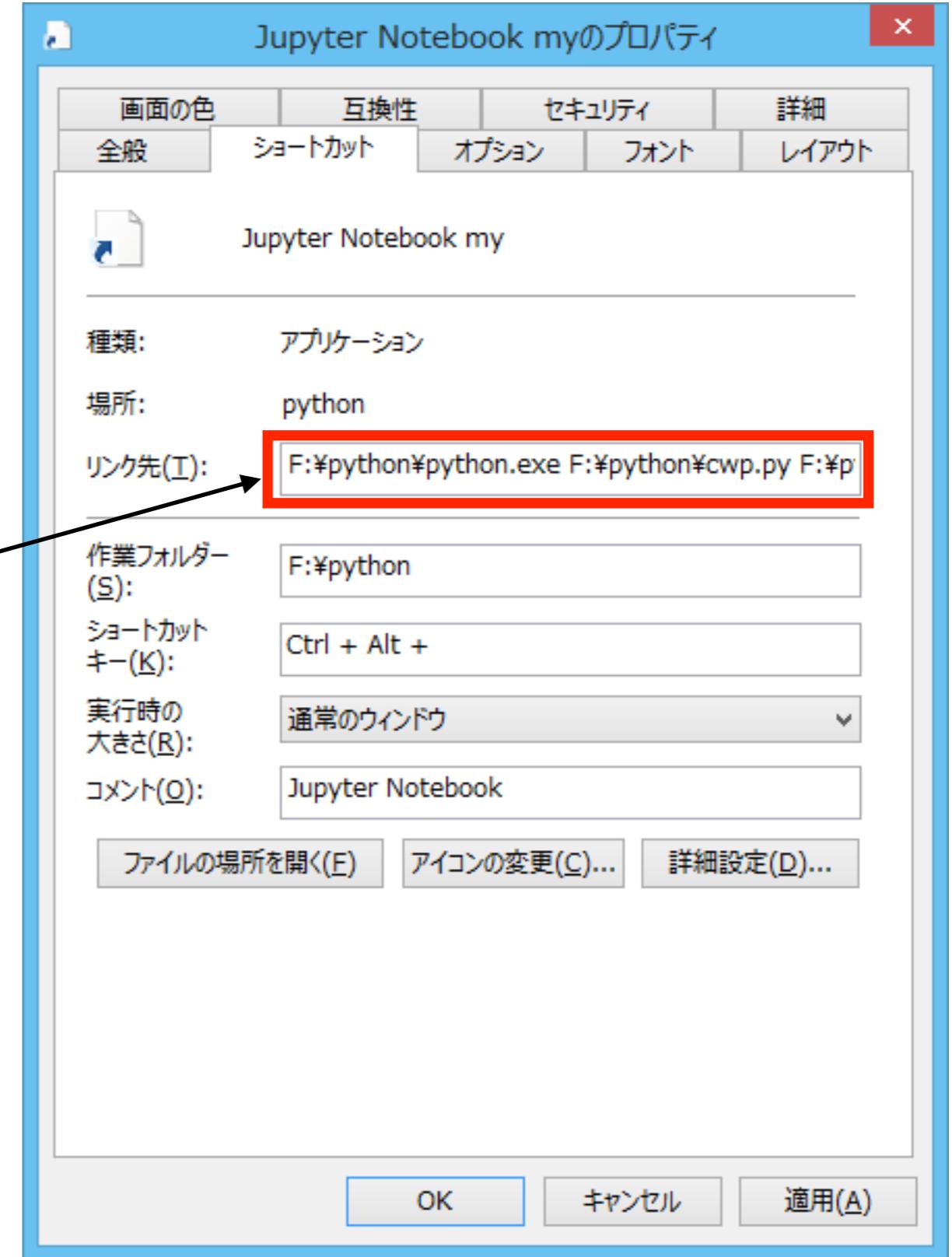
**毎回講義開始時に先ほどの準備作業をした後、  
このファイルをダブルクリックで起動**

# 起動しない場合、色々修正する必要あり

Jupyter\_Notebook.lnkの  
アイコンを右クリックし、  
プロパティを選択

ショートカットを選び  
右のような画面が出たら  
「リンク先」の部分の  
「F:¥python」をすべて  
自分の環境と同じに修正

「コマンドプロンプト」が  
「F:¥」以外ならそれも



A screenshot of a web browser displaying the Jupyter Notebook interface at `localhost:8888/tree/Desktop/mypython`. The browser's top bar includes standard navigation icons (Home, Back, Forward, Refresh, Home) and a title bar showing the URL. Below the title bar, there are several toolbar icons: a book, a star, a star with a line through it, a magnifying glass, a link, a Microsoft logo, and a camera. A "Logout" button is also present.

The main content area features the Jupyter logo and a navigation bar with three tabs: "Files" (selected), "Running", and "Clusters". Below the tabs, a message says "Select items to perform actions on them." On the right side of this message are three buttons: "Upload", "New ▾", and a refresh icon.

The central part of the interface shows a file list. It includes a checkbox for selecting items, a count of "0" items, a dropdown arrow, a folder icon, and the path "Desktop / mypython". To the right of the path are two sorting buttons: "Name" with a downward arrow and "Last Modified". Below the file list, there is a small folder icon followed by "…". On the far right, the text "数秒前" (several seconds ago) is displayed.

At the bottom of the interface, a message states "The notebook list is empty."

# Jupyter Notebookの終了の仕方

**起動時に開いた黒いウインドウ  
(コマンドプロンプト)を閉じる**

# 今日のお題：Pythonに慣れよう

- 前回の復習とふりかえり
- 演習室を使うための**毎回の準備作業**について
- Jupyter Notebookの動かしかた
- Notebookをダウンロードして実行してみる
- Python言語の基本を学ぶ  
print文、変数、数字の計算
- Markdownで文書を飾ってみよう
- ミニレポートタイム

# 先週のお題：Pythonを始めよう

- 前回の復習とふりかえり
- paiza.IOでPythonに触れてみる
- Python言語の基本を学ぶ + 最初はどうやって勉強すればいいか？
- Pythonインタプリタと実行環境について
- USBメモリのPython環境の使い方 (改)
- Jupyter Notebookの動かしかた
- (補足) なぜ前回動かなかったか?
- (補足) USBメモリについて
- ミニレポートタイム

# 大雑把なビジョン

- この演習でやりたいことは大きく2つ
  1. 「プログラミング」とは何かを理解しそれをどうやって学んでいけば良いかの指針を与える
  2. 「Python」というプログラミング言語の基礎と実習を実際にやってみる

# まずはprintと電卓

## 1. 文字を出力

`print('出力したい文字')`

`print("Don't do it!")` ←'があるときは"でくくる

## 2. 四則演算

足し算 + 引き算 - 掛け算 \* 割り算 /

割った余り % べき乗 \*\*

`round(1.2345, 3)` ←小数点以下3桁に丸める

(四捨五入とちょっと違う)

# print文をもう少し

```
print('答えは', 3, 'です')
```

```
print('答えは', 3, 'です', sep=":")
```

```
print('答えは', 3, 'です', sep=":", end=";")
```

```
print('答えは', 3, 'です', sep=":", end=";")
```

```
print()
```

```
print('答えは', 3, 'です', sep=" ", end="\n")
```

バックスラッシュは  
日本語キーボードでは「¥」

# 変数と代入

```
a = 3
```

```
b = 4
```

```
hoge = 5
```

```
kotae = a * b + hoge
```

```
print('答えば', kotae, 'です', sep=')
```

バックスラッシュは  
日本語キーボードでは「¥」



# 練習環境：Jupyter Notebook

## ▼ 1 お絵かきしてみよう！

下のセルを選択して、CtrlキーとEnterキーを同時に押してください。

```
In [40]: import matplotlib.pyplot as plt
%matplotlib inline

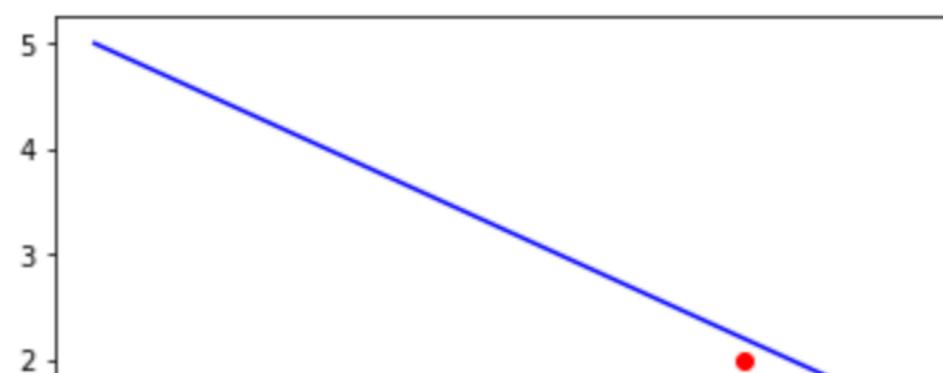
def line(p1, p2, c='blue'):
    plt.plot([p1[0], p2[0]], [p1[1], p2[1]], color=c)

def point(p, c='red'):
    plt.plot(p[0], p[1], 'o', color=c)
```

上はふたつの命令「line」と「point」を定義しました。これを使ってお絵かきしてみましょう。

下記のセルを自由に編集して命令を並べてください。実行はCtrlキーとEnterキーの同時押しか、上部のRunアイコンで！

```
In [45]: line([0, 0], [1, 1])
line([1, 1], [2, 1.5])
line([2, 1.5], [-3, 5])
point([0, 1])
point([1, 2])
```



<http://art.ist.hokudai.ac.jp/~takigawa/prog/>

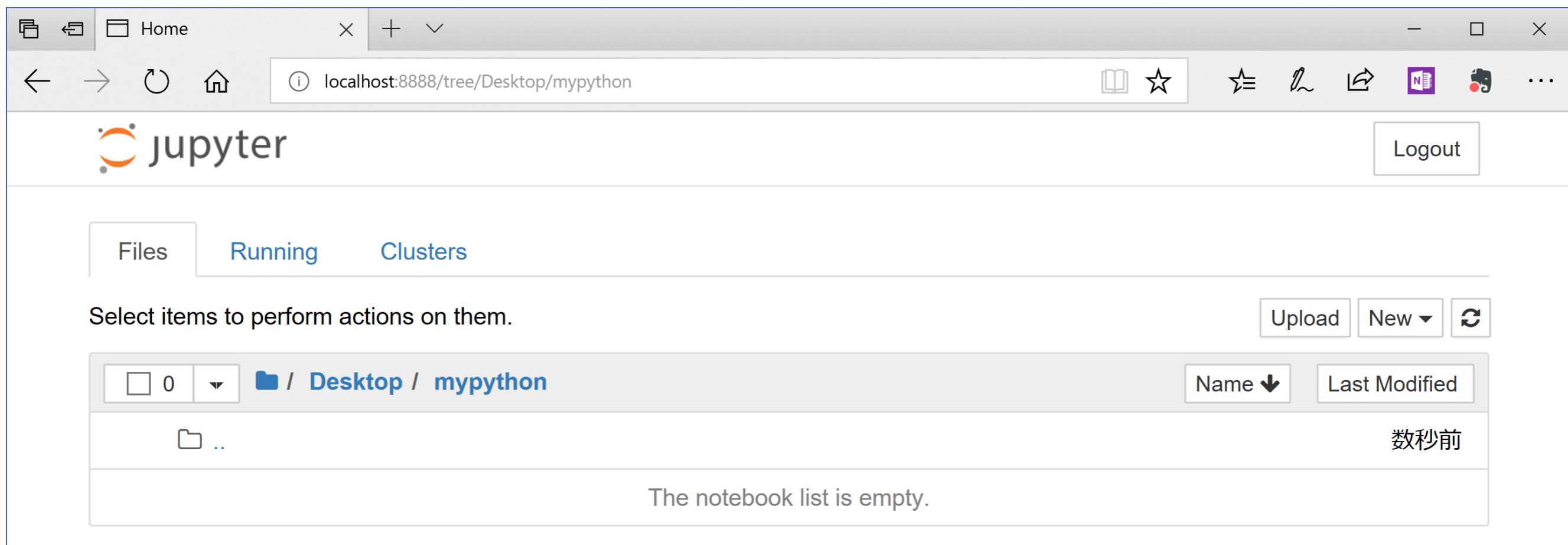
講義ウェブサイトから「Jupyter\_Notebook.lnk」を  
ダウンロードし、USBに保存しておく

毎回講義開始時に先ほどの作業をした後、USBから  
このファイルをデスクトップにコピーしダブルクリック  
すれば「Jupyter Notebook」が起動する

ただし先ほどの表示が「F:」で始まっていること、  
USBにanacondaを入れたフォルダ名が「python」  
であること、は必要

# 演習環境：Jupyter Notebook

起動すると下記のような画面に



The screenshot shows the Jupyter Notebook interface running at `localhost:8888/tree/Desktop/mypython`. The 'Files' tab is active. A context menu is open over an empty notebook list, with arrows pointing to the 'New' button and the 'Folder' option.

**新規作成**

**フォルダなども作成できる**

Select items to perform actions on them.

0 / Desktop / mypython

The notebook list is empty.

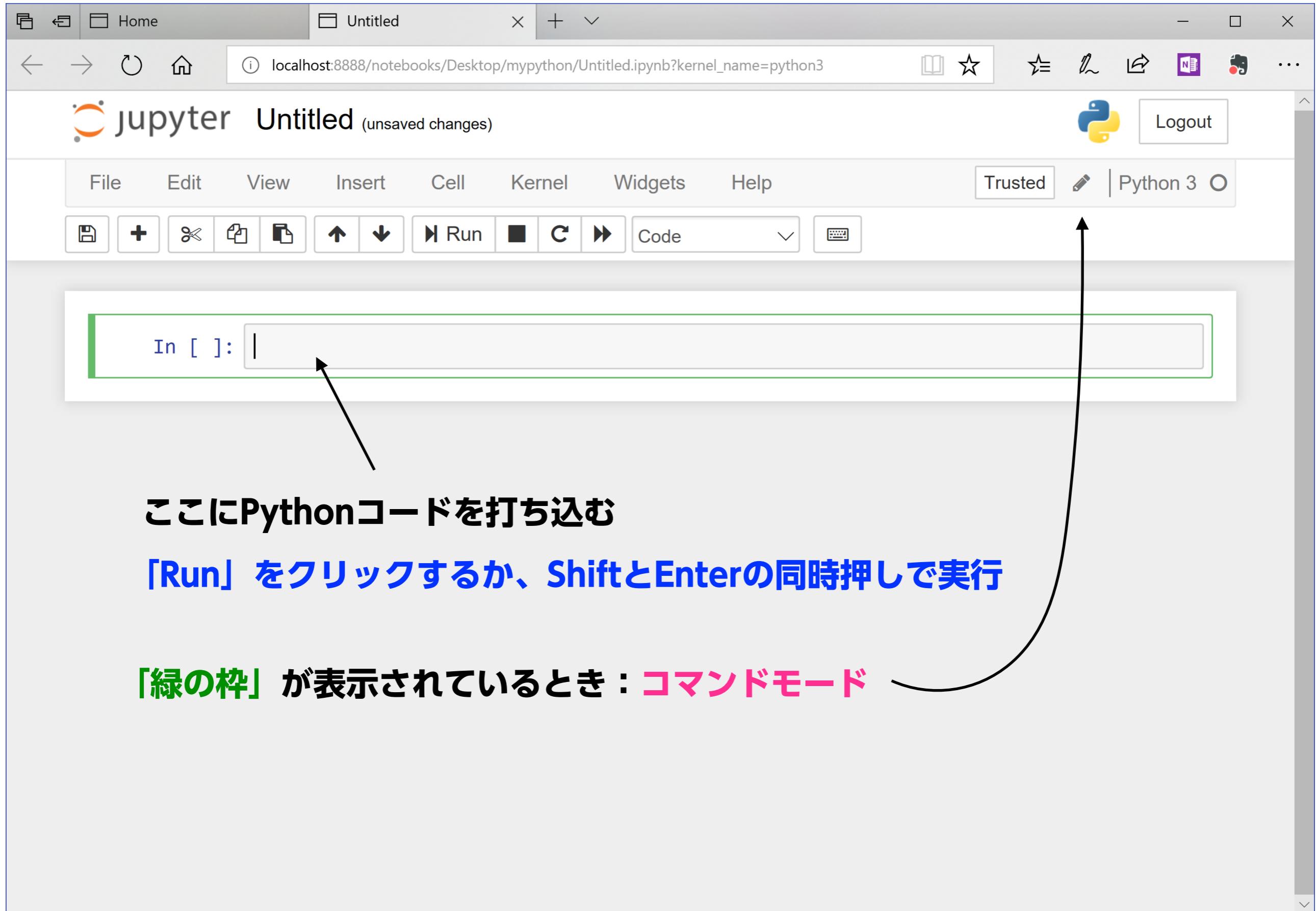
Upload New ▾

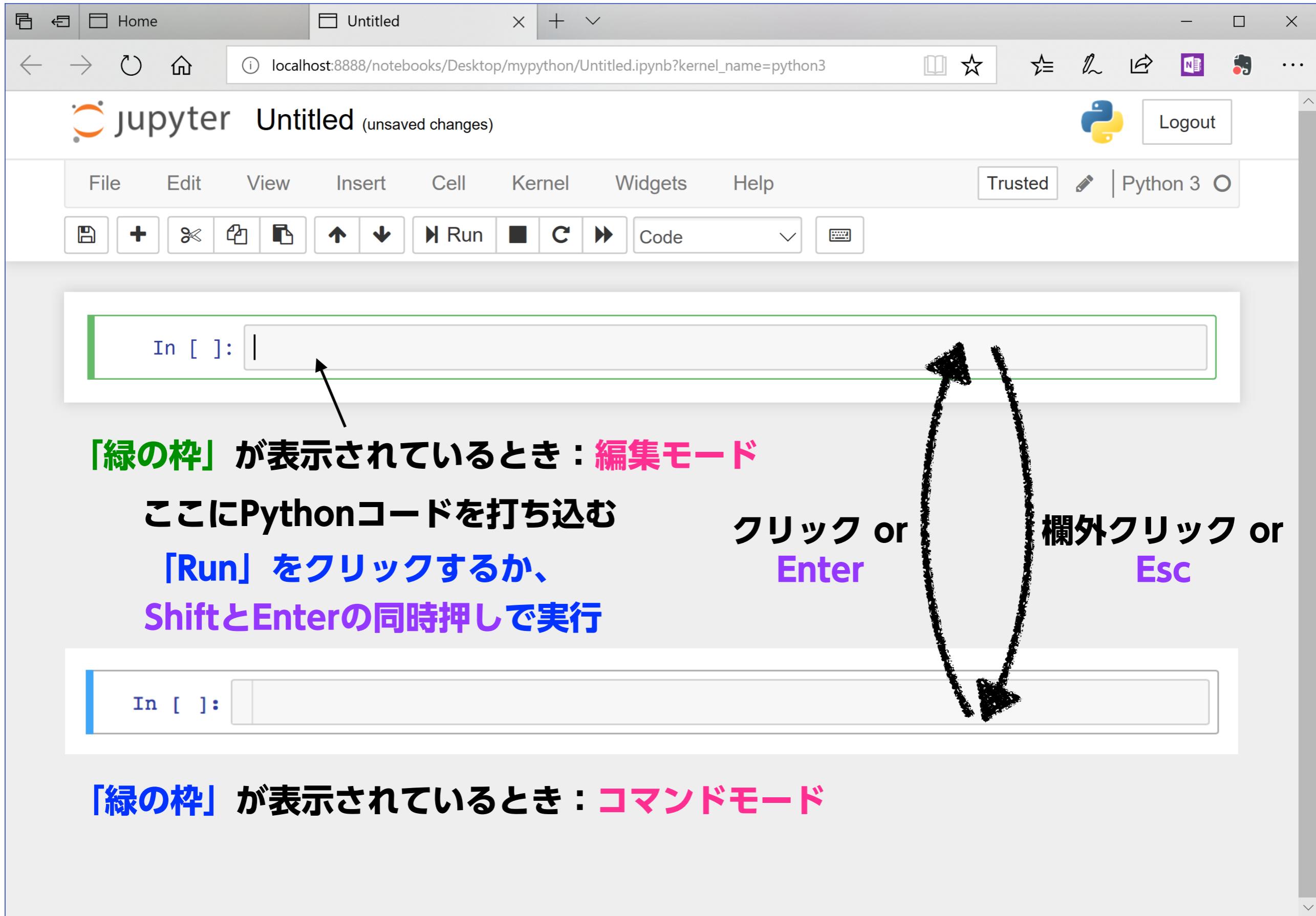
Notebook:

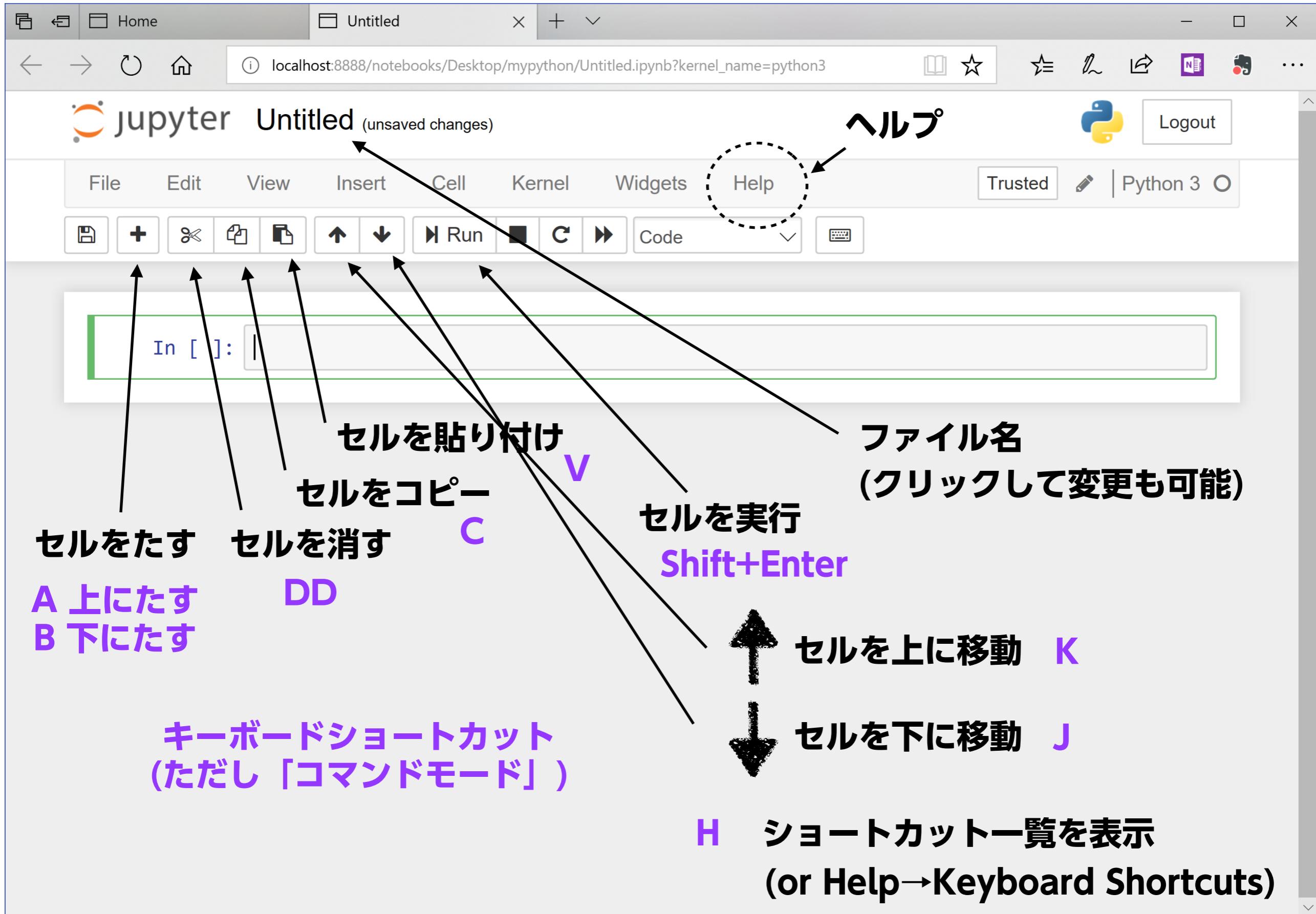
- Python 3

Other:

- Text File
- Folder
- Terminal







# 変数の情報を表示

```
In [4]: z = 19
```

```
In [5]: z?
```

「?」を最後につける

```
Type: int
String form: 19
Docstring:
int(x=0) -> integer
int(x, base=10) -> integer
```

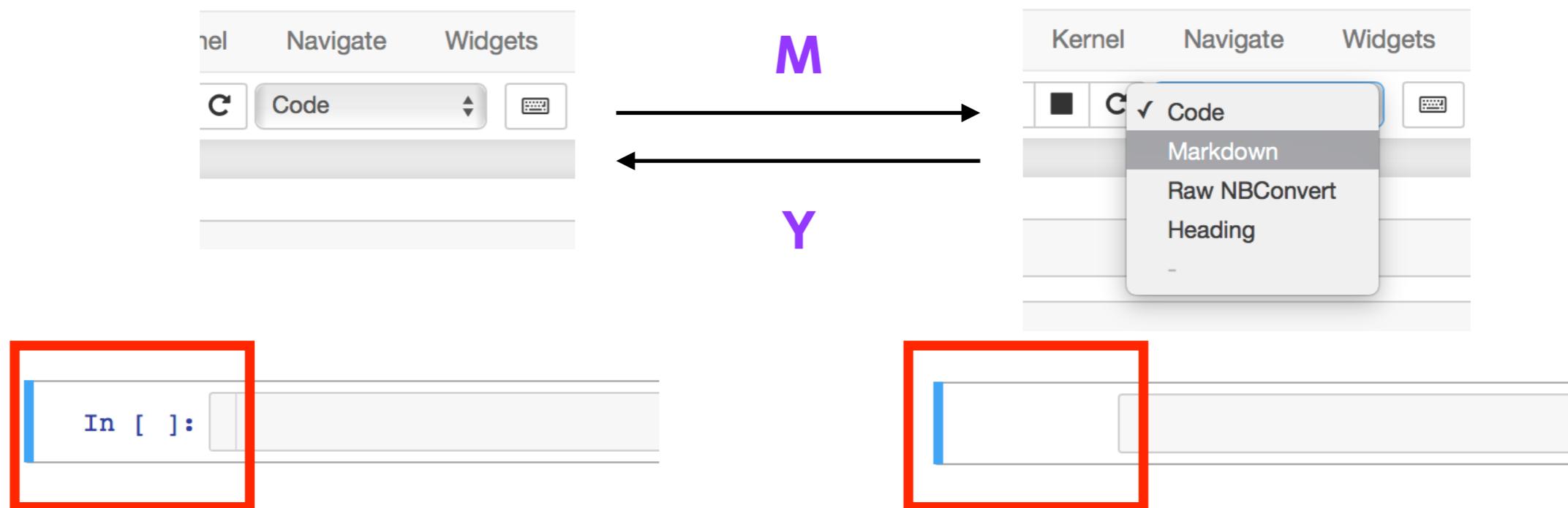
```
In [4]: z = 19
```

```
In [5]: z Shift+Tab
```

```
Type: int
String form: 19
Docstring:
int(x=0) -> integer
```

# Markdown

注：これはpythonとは直接関係ありませんが...



このように

色々書いてみると

- どういう感じに
- なるのでしょうか

Enter

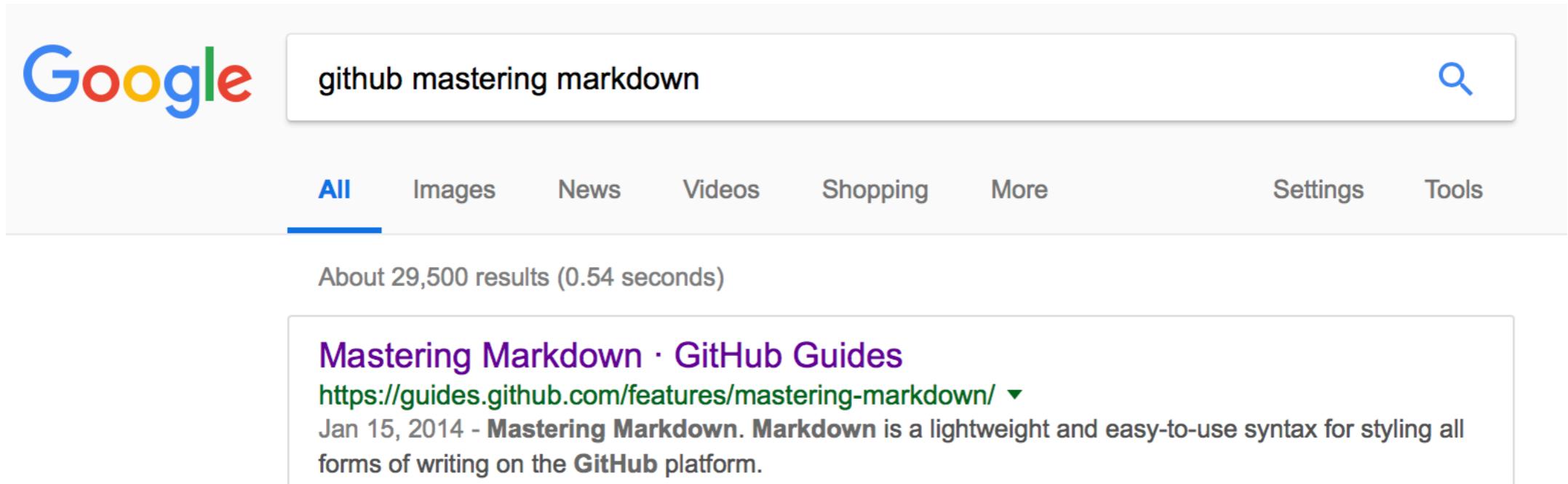
Shift+Enter

# このように

色々\*\*書いて\*\*みると

- どういう感じに
- なるのでしょうか|

# Markdownのルール



A screenshot of a Google search results page. The search query "github mastering markdown" is entered in the search bar. The "All" filter is selected. Below the search bar, there are links for Images, News, Videos, Shopping, and More. The results section shows a summary: "About 29,500 results (0.54 seconds)". The top result is a link to "Mastering Markdown · GitHub Guides" with the URL <https://guides.github.com/features/mastering-markdown/>. The snippet below the link reads: "Jan 15, 2014 - Mastering Markdown. Markdown is a lightweight and easy-to-use syntax for styling all forms of writing on the GitHub platform."

GitHub Guides

Video Guides



# Markdown記法

# これはヘディング

## これはサブヘディング

普通の文はこのように書いておく。  
強調したければ\*\*こう\*\*という風にかく。

箇条書きは

- その1
- その2

などとハイフンを使って書いておく

Pythonコードもこのようにかける。

```
```python
print('Hello!')
````
```

例えば、表もこのようにしてかける。

札幌 | 旭川

--- | ---

このような | 感じです

例えば | こう

(ちなみに)LaTeXという数式表記法も使えるようになっている(理工系の人向け)。

```
$$ \sum_{i=1}^n \frac{1}{n} $$
```

# フォーマット後

これはヘディング

これはサブヘディング

普通の文はこのように書いておく。 強調したければこういう風にかく。

箇条書きは

- その1
- その2 などとハイフンを使って書いておく

Pythonコードもこのようにかける。

```
print('Hello!')
```

例えば、表もこのようにしてかける。

札幌 旭川

このような 感じです

例えば こう

(ちなみに)LaTeXという数式表記法も使えるようになっている(理工系の人向け)。

$$\sum_{i=1}^n \frac{1}{n}$$

コードセルの説明などをMarkdownセル  
として入れておくと実行もでき結果も含み  
説明もあるインタラクティブ文書となる！

**作業：授業のホームページから「myfirstcode.ipynb」や  
「ex1.ipynb」「ex2.ipynb」をダウンロードして実行し  
てみよう。**

**<http://art.ist.hokudai.ac.jp/~takigawa/prog/>**

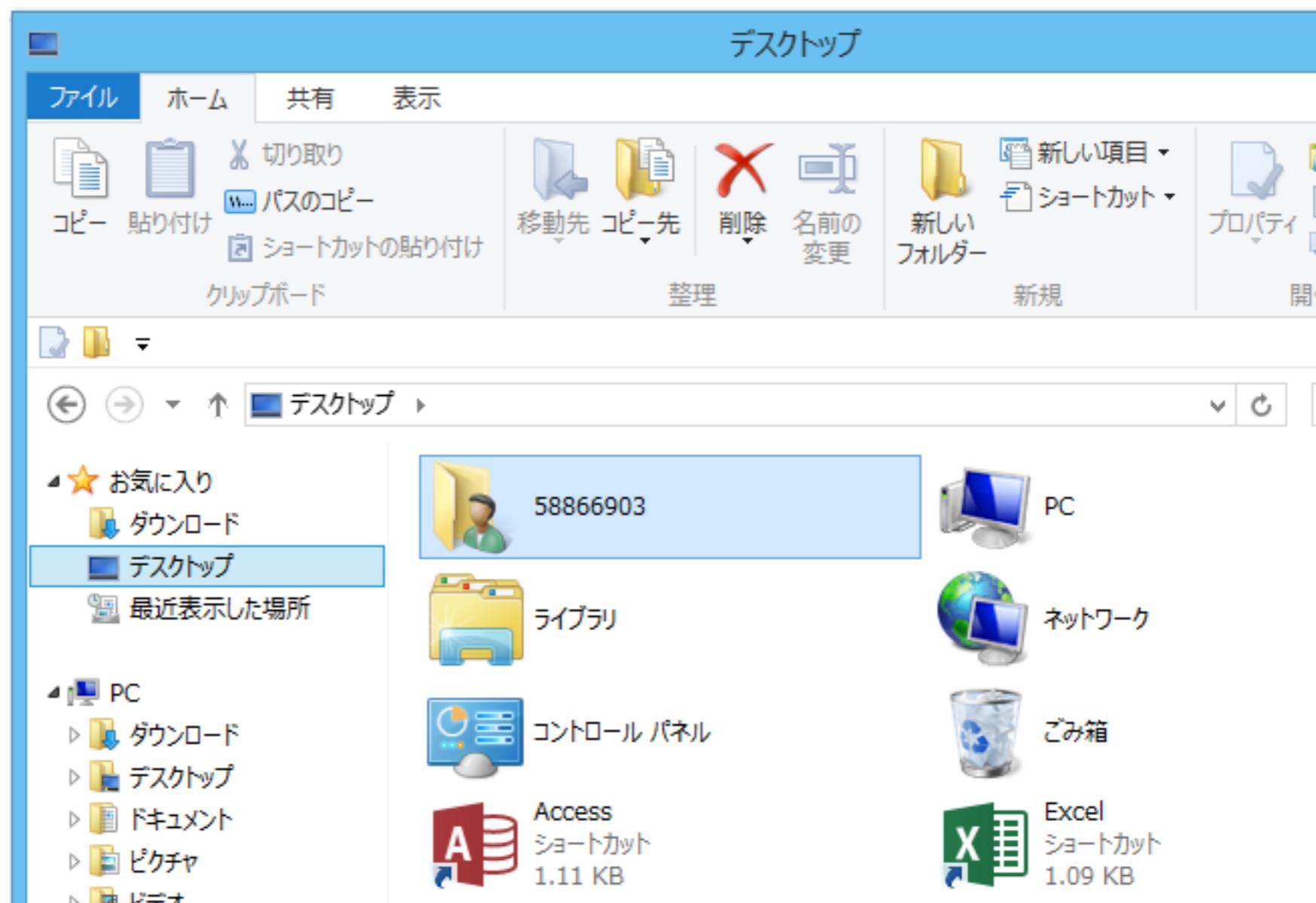
**Pythonが起動しているフォルダにコピーし、  
クリックして開く！**



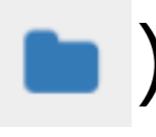
# 「Pythonが起動しているフォルダ」

Anacondaを入れたフォルダのScriptsのjupyter-notebookを実行して起動した場合 → そのScriptsフォルダ

## 「Jupyter\_Notebook.lnk」を設定し起動した場合



# 「Pythonが起動しているフォルダ」

ファイルブラウザの画面でホーム(  )をクリックしたところに作ったPython3ファイル(.ipynb)で

## Windowsの場合

コードセルで 「!cd」 を実行して表示されるフォルダ

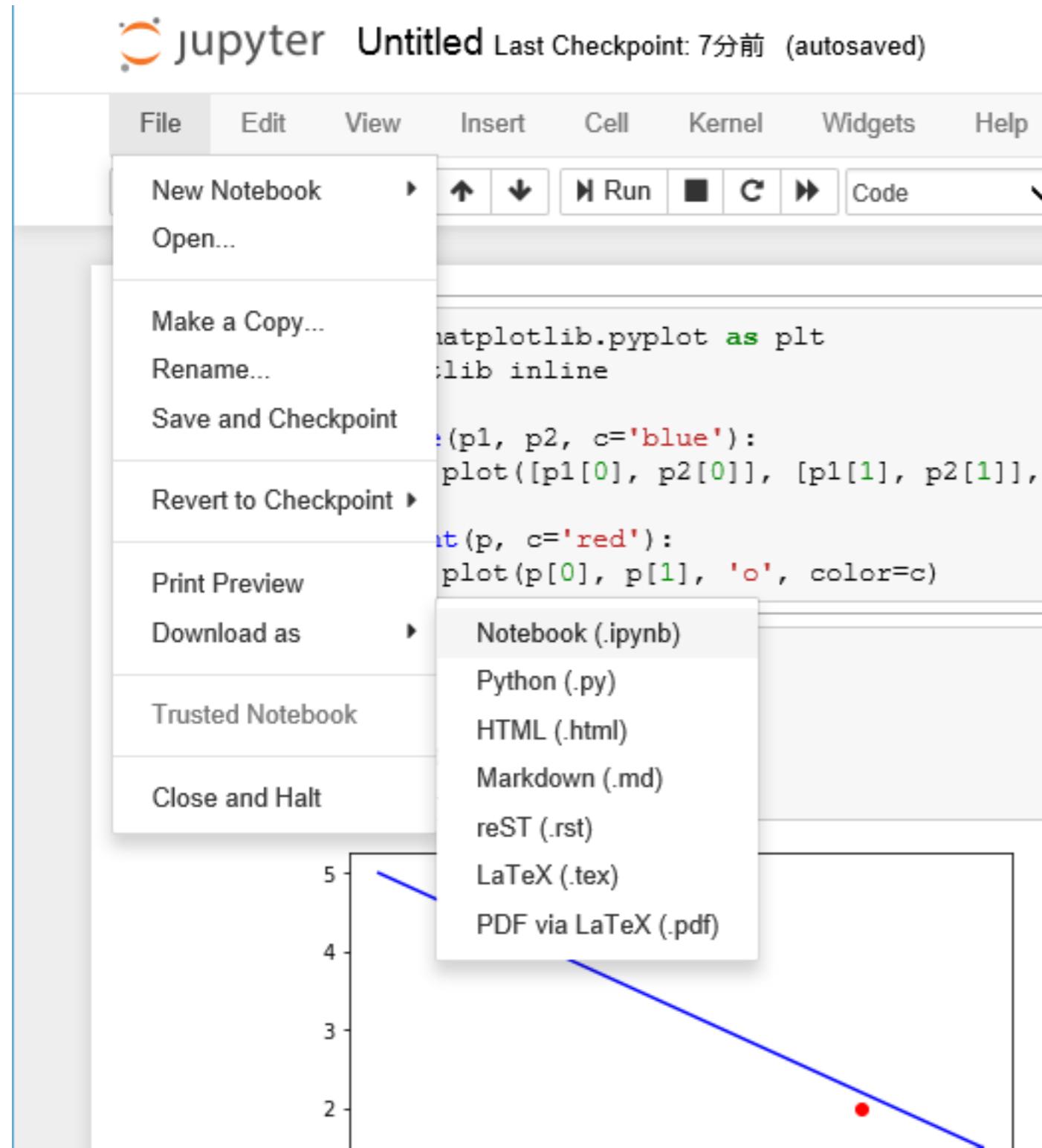
F:¥python¥subfolder ←これはF:ドライブのpython  
フォルダの中のsubfolderフォルダ

## Mac/Linuxの場合

コードセルで 「!pwd」 を実行して表示されるフォルダ

注意：「¥」記号は英語では「\」に対応するので  
同じことを指す。適宜読み替えてください

# 実行できたらそのJupyter Notebookを保存してみよう！



Jupyter Notebookが起動し動いて入れば、この環境でかなり高度なこともできます。またネットに転がっているNotebookを実行して見ることもできます！

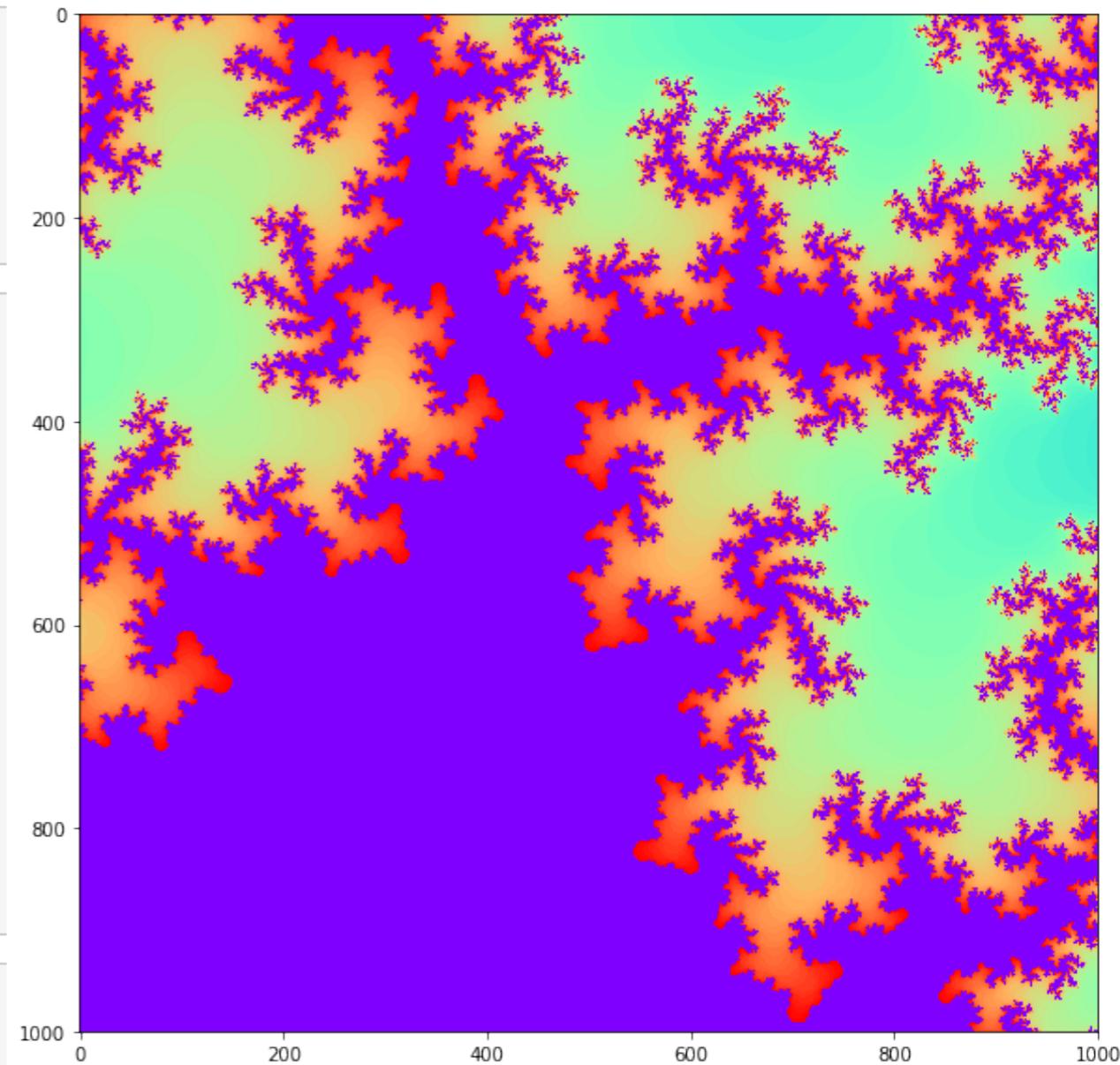
```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline

mpl.rcParams['figure.figsize'] = 10, 10
```

```
def mandelbrot(c, maxiter):
    z = c
    for n in range(maxiter):
        if abs(z) > 2:
            return n
        z = z*z + c
    return 0

def mandelbrot_set(xmin, xmax, ymin, ymax, w, h, maxiter):
    a = np.linspace(xmin, xmax, w)
    b = np.linspace(ymin, ymax, h)
    s = np.empty((w, h))
    for i in range(w):
        for j in range(h):
            s[i, j] = mandelbrot(a[i] + 1j*b[j], maxiter)
    return s
```

```
m = mandelbrot_set(-0.56, -0.55, -0.56, -0.55, 1000, 1000, 80)
plt.imshow(m, cmap='rainbow')
```



# Pythonの学習について

- Python公式チュートリアル

<https://docs.python.jp/3/tutorial/>

- LearnPython.org interactive Python tutorial (英語)

<https://www.learnpython.org>

- CodeAcademy (英語)

<https://www.codecademy.com/learn/learn-python>

- Intro to Python for Data Science (英語)

<https://www.datacamp.com/courses/intro-to-python-for-data-science/>

- Python tutorials for beginners (英語)

<http://thepythonguru.com>

- Progate (プロゲート)

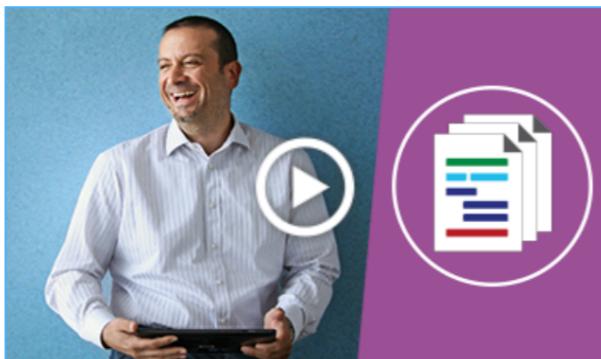
<http://prog-8.com/languages/python>

- ドットインストール

[https://dotinstall.com/lessons/basic\\_python\\_v3](https://dotinstall.com/lessons/basic_python_v3)

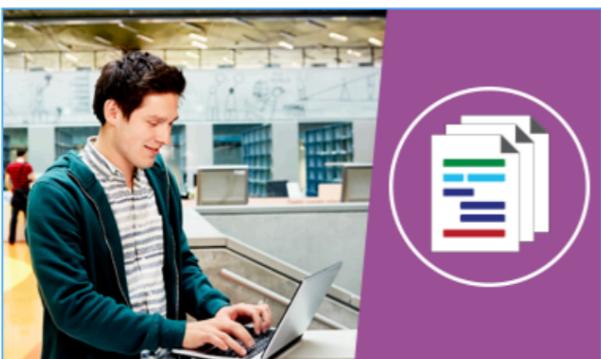
# オンラインで無料の良い講義も多数！

<https://wiki.python.org/moin/BeginnersGuide/NonProgrammers>



## Introduction to Python: Absolute Beginner

In this course that's perfect for true beginners, learn Python basics and start coding right away.



## Introduction to Python: Fundamentals

Build on what you learned in the "Introduction to Python: Absolute Beginner" course, and dig into data structure basics.



FREE COURSE

## Introduction to Python

Starting Out in Python 3

START FREE COURSE

- LearnPython.org interactive Python tutorial (英語)

## Indentation

Python uses indentation for blocks, instead of curly braces. Both tabs and spaces are supported, but the standard indentation requires standard Python code to use four spaces. For example:

The screenshot shows a web-based Python environment. At the top, there are two tabs: "script.py" and "IPython Shell". The "script.py" tab is active, displaying the following Python code:

```
1 x = 1
2 if x == 1:
3     # indented four spaces
4     print("x is 1.")
```

Below the code editor is a large, empty white area representing the IPython Shell. At the bottom left of this area is a yellow "Run" button.

Powered by DataCamp 

## Exercise

Use the "print" command to print the line "Hello, World!".

The screenshot shows a web-based Python environment. At the top, there are two tabs: "script.py" and "IPython Shell". The "script.py" tab is active, displaying the following Python code:

```
1 print("Goodbye, World!")
```

Below the code editor is a large, empty white area representing the IPython Shell.

# •CodeAcademy (英語)

The screenshot shows a Python tutorial on Codecademy. The main area features a code editor with a dark theme, a terminal window, and a sidebar.

**Code Editor:** The file is named "script.py". It contains the following code:

```
1 print "hoge"
```

**Terminal:** The output "hoge" is displayed.

**Sidebar:**

- PYTHON SYNTAX**
- Hello World!**
- If programming is the act of teaching a computer to have a conversation with a user, it would be most useful to first teach the computer how to speak. In Python, this is accomplished with the `print` statement.
- ```
print "Hello, world!"  
print "Water—there is not a drop  
of water there! Were Niagara but a  
cataract of sand, would you travel  
your thousand miles to see it?"
```
- A `print` statement is the easiest way to get your Python program to communicate
- Instructions
- [Community Forums](#)
- [Report a Bug](#)

**Bottom Navigation:**

- 1. Hello World!
- Back
- 1/14
- Next
- Get Help

- Intro to Python for Data Science (英語)

The screenshot shows a DataCamp video player interface. At the top left is the DataCamp logo. To its right is a navigation bar with 'Course Outline' and a right-pointing arrow. Further right are icons for notifications, a document, and help. The main title 'Hello Python!' is displayed prominently in large dark font. To its right is a small circular badge with '50XP'. Below the title is a blue header bar with the DataCamp logo and a Python icon. The main content area features a large white hexagonal graphic containing the text 'INTRO TO PYTHON FOR DATA SCIENCE'. Overlaid on this graphic is the text 'Hello Python!' in large bold letters. To the right of the graphic, a man named Filip Schouwenaars is speaking. He is wearing a black polo shirt with the DataCamp logo on the chest. A subtitle at the bottom left identifies him as 'Filip Schouwenaars' and shows a timestamp of '0:04'. The video player has a dark blue footer bar with controls: a play button, a volume icon, a progress bar, a download icon, a 1x speed icon, a CC icon, and a full-screen icon. In the bottom right corner of the video frame is a yellow button with the text 'Got it!'. The overall background of the player is light gray.

The screenshot shows a Progate Python tutorial page. At the top, there's a navigation bar with links for 'レッスン一覧' (Lesson List), 'スライド検索' (Slide Search), 'ランキング' (Ranking), and 'ヘルプ' (Help). On the right, there's a user profile for 'Ichigaku Takigawa (Lv.2)' with icons for person, group, and notifications.

The main content area has a title '文字列' (String) with a star icon. Below it is a section titled '文字列とは?' (What is a string?). It contains two tabs: '・ 文字列' (String) and '・ クオーテーション' (Quotation). The 'String' tab is selected.

The text in this section explains that strings are represented by single quotes ('') or double quotes ("") and provides examples:

シングルクオーテーション(')  
または、ダブルクオーテーション("")で囲む

Two code snippets in 'script.py' are shown:

```
print('Hello Python')
print("Hello Python")
```

Below this, another 'script.py' snippet shows an error:

```
# エラー発生
print(Hello Python)
```

To the right, under '出力結果' (Output Result), there are two sections labeled '\_ コンソール' (Console). The first shows successful outputs:

>\_ コンソール  
Hello Python ← 出力結果は同じ  
Hello Python

The second section shows an error:

>\_ コンソール  
SyntaxError: invalid syntax  
X エラー！！：必ず文字列はクオーテーションで囲む！

Green arrows point from the successful outputs to the explanatory text '出力結果は同じ' (The output results are the same).

At the bottom right, there are navigation arrows for the slide.

[レッスン一覧](#)[スライド検索](#)[ランキング](#)[ヘルプ](#)

Ichigaku Takigawa (Lv.2) ▾

[Pythonの基礎を学...](#)

3 / 17 ▾

script.py

```
1 # 数値の7を出力してください  
2  
3  
4 # 9に3足した値を出力してください  
5  
6  
7 # 「9 + 3」を文字列として出力してください  
8
```



&gt;\_ コンソール



## 数値

数値を出力してみましょう。

[script.py](#)

数値の7を出力してください。

[② スライドで確認](#)[script.py](#)

「+」を用いて、9に3を足した値を出力してください。

[② スライドで確認](#)[リセット](#)[答えを見る](#)[確認して次へ](#)[④ スライドを見る](#)

11%

[ご意見箱](#)

script.py

```
1 # 数値の7を出力してください
2 print(7)
3
4 # 9に3足した値を出力してください
5 print(9 + 3)
6
7 # 「9 + 3」を文字列として出力してください
8 print('9 + 3')
```



>\_ コンソール

```
7
12
9 + 3
□
```

>\_ 見本

🏆 PYTHON Lv.3  (+ 100 EXP)



# Congratulations!

正解です！次のレッスンは  
「計算してみよう」です！

解答を見る

次に進む



トップ

有料会員プラン

タグから検索

スキルチェック

paizaラーニングトップ > Python > Python3入門編レッスン一覧



## Python3入門編 (全6レッスン)

paizaラーニングの学び方

完全無料

Python

ツイート

Point.1  
環境構築不要  
5秒で開始!

Point.2  
3分動画で  
毎日続けられる

Point.3  
不明点は  
エンジニアに質問

Point.4  
就職直結の  
カリキュラム

paizaラーニングは、[プログラミング初心者～中級者向けのスキルアップ学習コンテンツ](#)です。

[1チャプター3分動画](#) なので、忙しくても毎日続けられます。

不明点は [エンジニアに質問が出来る](#) ので、つまづく事がありません。



### 本レッスンで身につけられるスキル

Pythonでプログラミングの初步を学ぶレッスンです。[\(Python3を対象として学んでいきます\)](#)

標準出力や変数、配列、演算子、条件分岐、ループ といったプログラミングの基本を手順を追って学びます。

D

E

対象者



# 今日のお題：Pythonに慣れよう

- 前回の復習とふりかえり
- 演習室を使うための**毎回の準備作業**について
- Jupyter Notebookの動かしかた
- Notebookをダウンロードして実行してみる
- Python言語の基本を学ぶ  
print文、変数、数字の計算
- Markdownで文書を飾ってみよう
- ミニレポートタイム