とくしまビジネス
リスキリング
スクール2024

基礎から始めるAIエンジニア育成講座

2. Pythonプログラミングと画像処理

徳島大学デザイン型AI教育研究センター
徳島大学理工学部　併任
鳥井 浩平

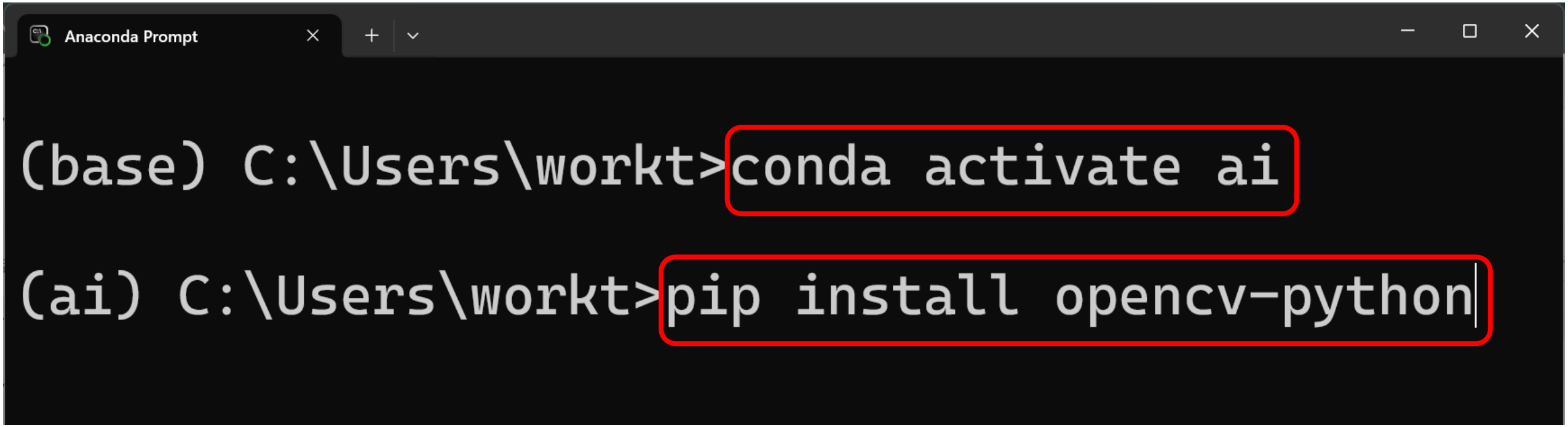
目次

- プログラミングの準備
- Pythonプログラミングの基本
- 空間フィルタリング（畳み込み演算）の実装
- 畳み込みニューラルネットワークの実装

資料の公開

https://researchmap.jp/wt501/published_works

追加の準備



プログラミングの準備

プログラミングとは

- コンピュータへの命令をまとめたもの（プログラム）を作成すること
  - ソースコード：プログラムの元となる文章
  - コーディング：プログラミング言語を用いて文章を記述すること

- プログラミング言語によって文章の記述ルールが異なる
  - Python，C，C++，Javaなど，200種類以上存在
  - 日本語・英語の違いと同様に単語や文法が異なるというイメージ

Pythonの特徴

インタプリタ型の言語

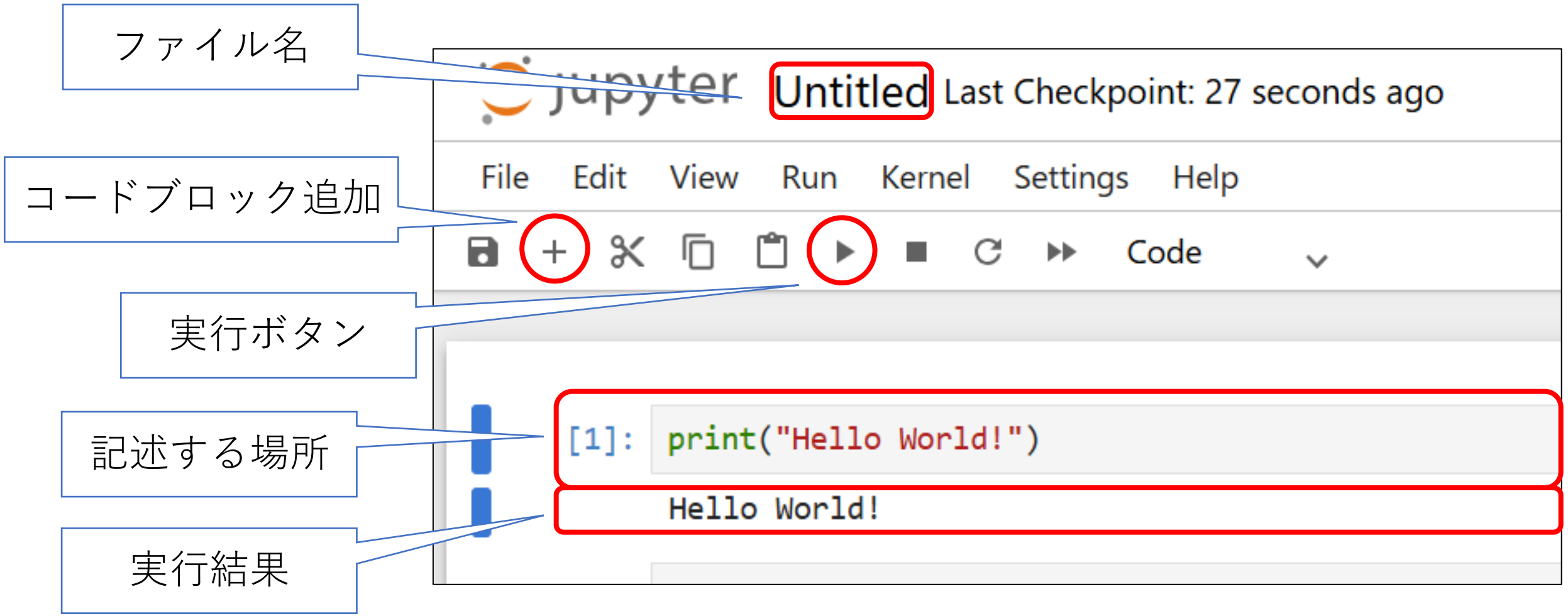
- ソースコードを機械語に翻訳する工程（コンパイル）が不要
- ソースコードをそのまま実行できる（ように見える）

機械学習関係の既成プログラム（ライブラリ）が潤沢

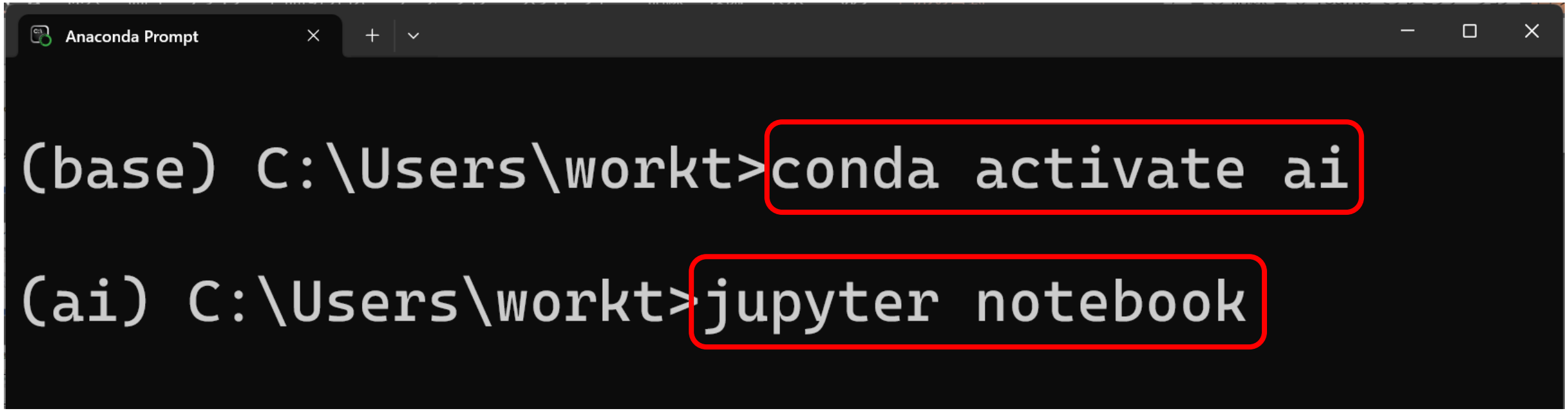
- 自分で一から機械学習のプログラムを作る必要がない
- 複雑な機械学習も簡単に動かすことができる

Jupyter Notebook

※ファイル名をクリックするとファイル名を変更できる

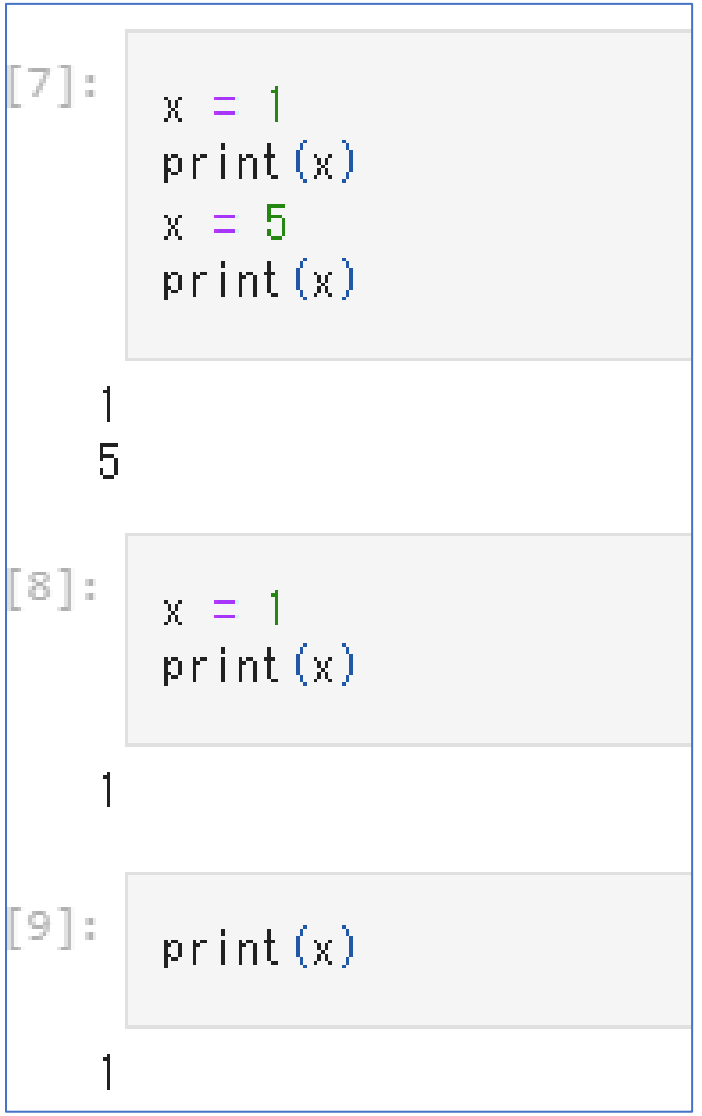


Jupyter Notebookの起動



コードブロック

- コードをブロックに分けて記述することができる
- ブロックごとに実行結果が得られる
- 下のブロックは上のブロックの実行結果を引き継ぐ
- 基本的には上のブロックから順に実行する
  - プログラムは上から順に流れるものです
- うまく使えば見やすいプログラムになる
  - 面倒くさい人は1つのブロックに全部書いてもいい



Pythonプログラミングの基本

ここからのスライド構成

- 黒枠内の太字文章が実際にプログラムとして書く内容です
  - ただし、# から始まる文章は書かなくても良い（詳細は後述）
- 黒枠の下にプログラムの解説を記載します

数値や文字列の代入

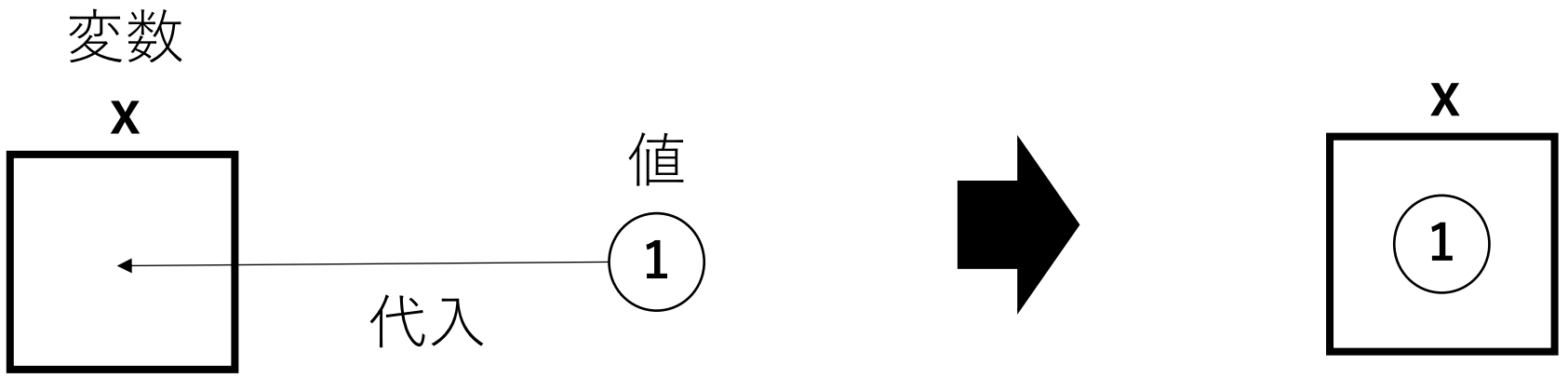
```
x = 1                    # xに1を代入する
y = "apple"              # yに"apple"を代入する
txt = "私は学生です"      # txtに"私は学生です"を代入する
```

- 等式の左辺は**変数**，右辺は**値**という
- 変数は自由に命名できる
  - 空白や特殊記号は基本的に使用できないがアンダーバー（_）は使える
- 右辺には変数に**代入**したい数値や文字列を記述する
- **ダブルクォーテーション(")**で囲んだ文章は**文字列**として扱われる
- #が先頭にある文章を**コメント**といい，コードとして扱われない

変数と値のイメージ

- 変数は「**名前付きの箱**」、値は「**箱の中身**」
- 代入は箱に中身を入れる作業

例）**x**という変数に**1**を代入するとき（x ＝ 1）



数値や文字列の表示

```
x = 1                    # xに1を代入する
print(x)                 # xの値を表示する
print("banana")          # bananaを表示する
```

- **print**は値を表示する**関数**
- 関数は**括弧()**に入力を与えることで結果を得られる
  - 複数の入力が必要なときは**カンマ(,)**で入力を区切る
- printは計算結果の確認などで頻繁に用いる

$$y = f(x)$$
$$z = g(a, b, c)$$

演算

```
x = 31
r = (x + 5) / 2          # r = (x+5)/2
print(r)

r = r % 4                # rを4で割った余り
print(r)

y = 23 ** 4              # 23^4
print(y)
```

演算	演算子
加算	+
減算	-
乗算	*
除算	/
べき乗	**
除算（余り）	%
切り捨て除算	//

リスト

```
s = [12, 32, 53]                # 12, 32, 53のリストをsに代入
print(s)
num = 55
fruits = ["apple", num]         # “apple”とnumのリストをfruitsに代入
print(fruits)
print(fruits[0])                # fruitsの0番目の要素を取り出して表示
```

- 複数の値を一つにまとめたいときは**リスト**を用いる
- **大括弧([])**と**カンマ(,)**を使う
- リストの中身を取り出すには**リスト[要素の番号]**のように記述する
  - 要素の番号はリストの左から順に0, 1, 2, ...のように割り当てられる

if文とfor文

if文

- 「もし～なら～を実行する」を表現できる文
- 実行結果によって後の処理を変更したいときに用いる

for文

- 「～を～回繰り返す」を表現できる文
- 同じ処理を何度も繰り返したいときに用いる

if文

```
x = 1
if x < 2:              # 条件を満たしたとき
  print("OK")          # OKを表示する
else:                  # 条件を満たさないとき
  print("NG")          # NGを表示する
```

比較演算	演算子
大なり	>
小なり	<
イコール	==
大なりイコール	>=
小なりイコール	<=
notイコール	!=

- **if**と**コロン(:)**の間に**条件式**を書く
- 文章を**字下げ**することで条件に対する処理を追加できる
- 字下げは**TABキー**を使う

字下げ（インデント）の注意点

```
x = 3
if x < 2:
  print("OK")
  x = x + 1
print(x)
```

```
x = 3
if x < 2:
  print("OK")
x = x + 1
print(x)
```

- 左のコードと右のコードは実行結果が異なる
- 左はif文の対象内に x = x + 1 が記述されている
- 右はif文の対象外に x = x + 1 が記述されている
- 字下げによってif文の対象となる処理の範囲が変わることに注意
  - for文でも同様

複数の条件分岐

```
x = 1.5
if x < 1:              # xが1未満なら
  print(“a”)           # aを表示する
elif x < 2:            # xが2未満なら
  print(“b”)           # bを表示する
else:                  # それ以外は
  print(“c”)           # cを表示する
```

- **elif**を使って条件分岐を増やすこともできる

for文

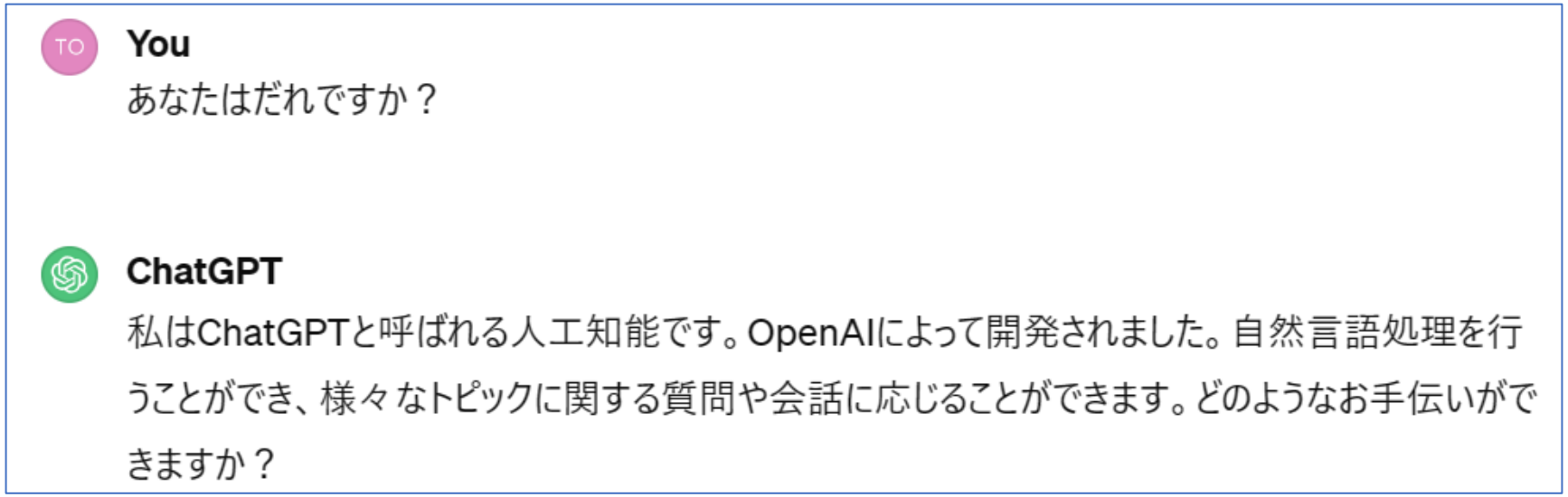
```
numbers = [31, 59, 43]
for n in numbers:       # numbersから順に値を取り出しnに代入する
  print(n)              # nを表示する
  print("---")          # ---を表示する

for n in range(10):     # 10個の正の整数（0~9）を順にnに代入する
  print(n)
```

- **for 変数 in リスト:** でリストから順に値を取り出し変数に代入する
- リストから値を取り出すたびに対象の処理を実行する
- **range(回数)**は一定回数処理を繰り返したいときに便利な関数

ChatGPTの活用

ChatGPTとは

- 人間と対話しているかのように自然な回答文を返すチャットサービス
- 質問すると大体それらしい回答をしてくれる超有能AI
- 特にプログラミングサポートとして非常に有用
- アカウント登録不要かつ無料で利用可能



ChatGPTの活用例①

You

Pythonで単回帰を行う簡単なコードを教えてください

ChatGPT

以下は、Pythonで単回帰を行う簡単なコードの例です。この例では、NumPyとscikit-learnライブラリを使用しています。

python Copy code

```python
import numpy as np
from sklearn.linear_model import LinearRegression

# サンプルデータの準備
X = np.array([[1], [2], [3], [4], [5]])  # 説明変数
y = np.array([2, 3.5, 2.8, 4.6, 5.3])     # 目的変数
```

ChatGPTの活用例②

You

このコードについて、より詳しく解説してください

ChatGPT

このコードは、単回帰分析を実行するための基本的なPythonコードです。以下、各部分の詳細な解説です。

1. ライブラリのインポート:

python Copy code

```
import numpy as np
from sklearn.linear_model import LinearRegression
```

- `numpy`は数値計算を行うためのライブラリであり、配列操作や線形代数の機能を提供します。
- `LinearRegression`はscikit-learnライブラリに含まれる線形回帰モデルのクラスです。

ChatGPTの活用例③

You

model = LinearRegression()
これは何をしているコードですか

ChatGPT

このコードは、scikit-learnライブラリの`**LinearRegression**`クラスのインスタンスを作成しています。`**LinearRegression**`クラスは、線形回帰モデルを表すためのクラスです。このモデルは、説明変数と目的変数の間の線形関係をモデル化します。`**model = LinearRegression()**`の行は、単に線形回帰モデルのインスタンスを作成し、それを`**model**`という変数に割り当てています。これにより、そのインスタンスを使用してモデルのフィットや予測を行うことができます。

ChatGPTの活用例④

You

x == 1
print(x)
このコードでエラーが出ます。修正してください。

ChatGPT

エラーが出る理由は、`x == 1`を式として使用しようとしているためです。これは比較演算子であり、`True`または`False`を返しますが、これを変数に割り当てることはできません。正しいコードは次のようになります。

python Copy code

```
x = 1
print(x)
```

課題①

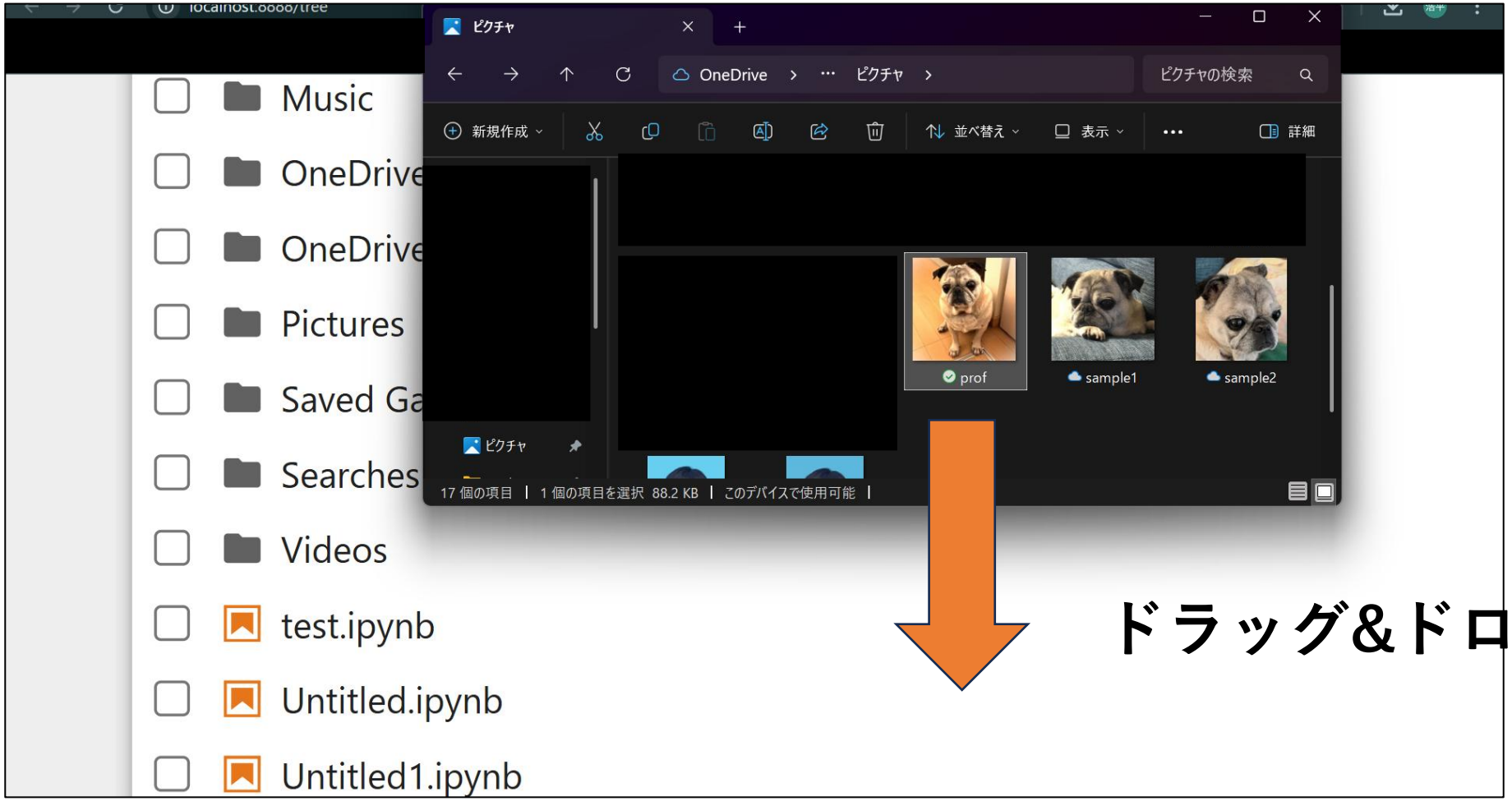
二次方程式$2x^2 + 19x + 13 = 0$を満たす実数解を計算する

- 解の公式を用いる
- a, b, cは変数で定義する
- $\pm$は$+$と$-$の式に分ける
- $\sqrt{x}$は$x^{0.5}$として考える
- 実数解がない場合はエラー文を出す

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

空間フィルタリング

画像の準備



おまじない

```
import cv2
import numpy as np
from matplotlib import pyplot as plt
```

- いろんな機能を使うためのおまじない
- **import パッケージ名**　で機能を読み込み

画像の読み込み

```
img = cv2.imread("prof.JPEG")[..., ::-1]
img = cv2.resize(img, (256, 256))
plt.imshow(img)
plt.axis("off")
plt.show()
plt.clf()
```

- 画像を読み込み、256×256の解像度にリサイズ
- **赤字**の部分は自分が準備した画像のファイル名が入る
- **plt**を使って画像を表示

画像のグレースケール変換

```
gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
plt.imshow(gray, cmap="gray")
plt.axis("off")
plt.show()
plt.clf()
```

- **cv2.cvtColor**で画像の色変換を行う

空間フィルタリング

```
kernel = np.array([
    -1, -2, -1,
    0, 0, 0,
    1, 2, 1
])
dst = cv2.filter2D(gray, -1, kernel)
plt.imshow(dst, cmap="gray")
plt.axis("off")
plt.show()
plt.clf()
```

- **kernel**の値を変更すると出力画像も変化する

演習②

kernelの値を書き換えて実行してみよう

畳み込みニューラルネットワークの実装

データセットとプログラム

https://genai-book.github.io/materials/