



# AI総合演習

## 第2回: 再現可能性とプログラミング基礎

瓜生真也 (デザイン型AI教育研究センター/理工学部・助教)

# 【課題】 Pythonプログラムの作成・実行

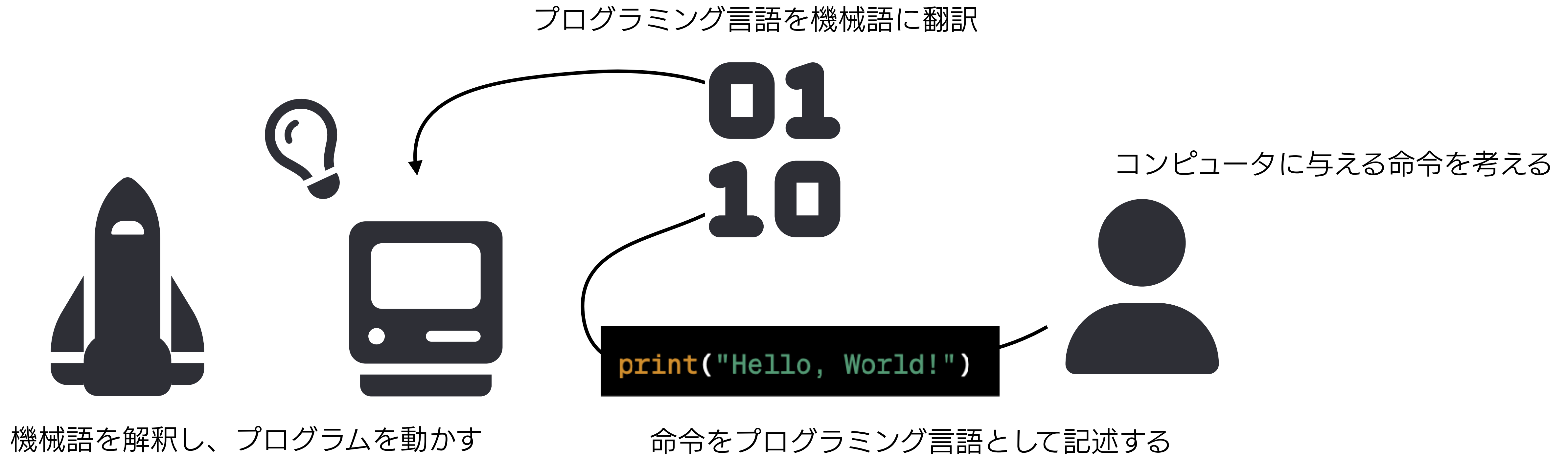
提出期限: 来週の講義開始前まで

manabaのレポートとして提出してください

# プログラミング

→コピーから月面着陸までを可能にする

# プログラミングの流れ



# 言葉の整理

プログラム(program) コンピュータが特定の作業を行うための一連の命令（の集まり）

命令（プログラム）をコンピュータに与えることで機能する

一つ一つの命令は非常に単純な処理を実行する

多くの場合、複数の命令を組み合わせて、より複雑で実用的なプログラムを作成する→アプリケーション、ソフトウェア

## プログラミング

プログラムを作成・編集する過程や活動

デバッグ（エラーの検出と修正）を含む

## プログラミング言語

プログラムを実装するために使用される言語

多様な言語… C、Python、R、Swift

選択する言語は目的の要件によって異なる

## プログラマー

プログラムを開発・編集・デバッグする人々

プログラムの内容に応じて職種がわかる… アプリケーションエンジニア、ゲーム開発者、データサイエンティストなど

---

## スクリプト(script)ないしコード(code)

コードはプログラムを構成するプログラミング言語で記述された一連の命令や文を指す

スクリプトはコードの断片

# プログラミングの流れ（再掲）

プログラマー

プログラミング

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

プログラム

プログラミング言語

# 退屈なことはプログラムに任せよう

例えば…ウェブページのURLをもとにQRコードを作りたい  
そういうウェブサービス・ツールはいくつか存在する

対象のURLが1つならいいかも 対象のURLをコピーして貼り付け、生成 😊

URLが1000種類あったら？ 1000回もやってられない 😞



## プログラミングでできることの例

データ分析、機械学習・深層学習、ウェブスクレイピング、アプリケーション開発

プログラミング言語ごとに得意・不得意、向き不向きがある

## プログラムとAIの力で時間短縮

戦略1: プログラミングを学ぶ

まずはここからはじめよう

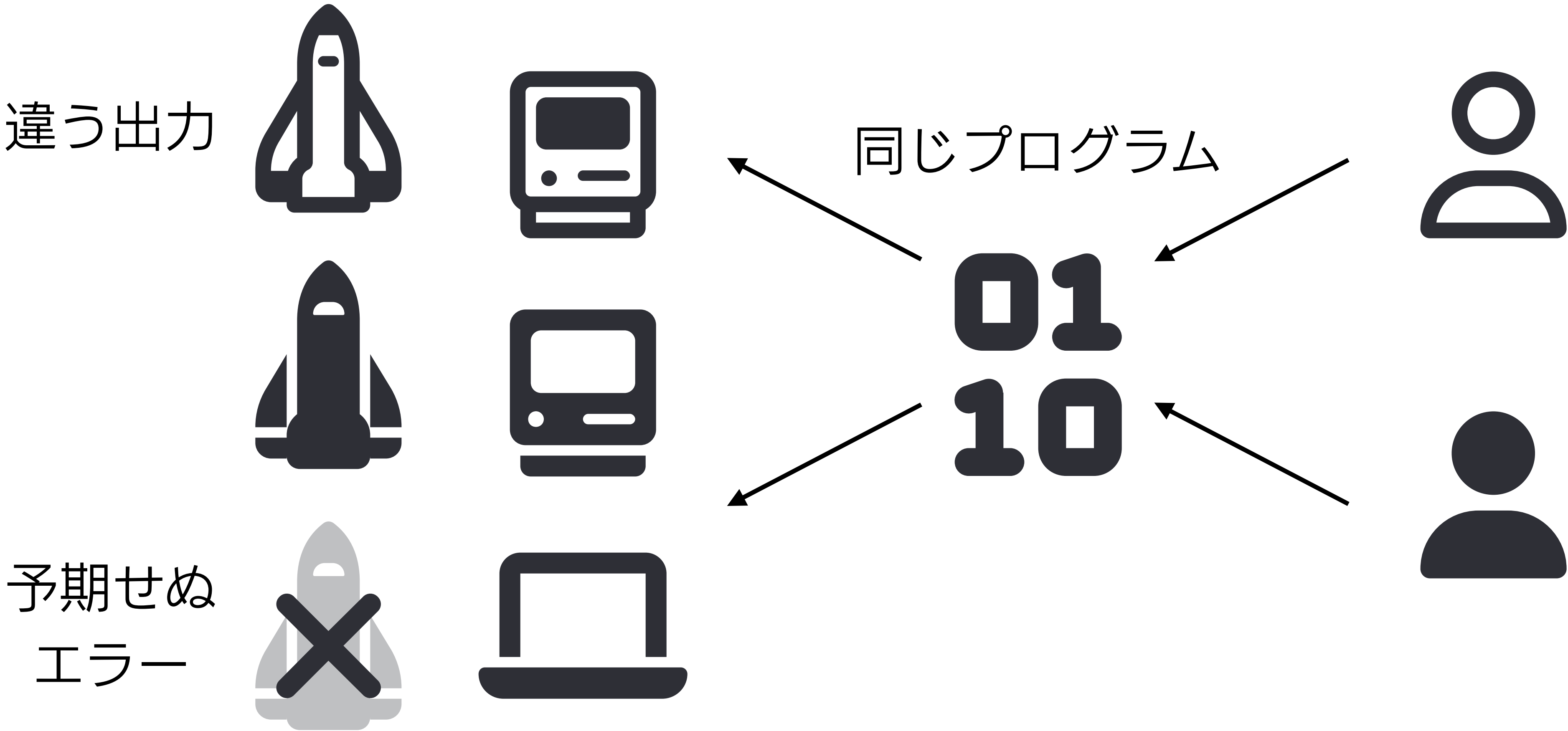
戦略2: AIの力を借りる

後半で解説

# 実行環境の違いに注意

プログラム実行時に利用されるコンピュータのハードウェア・ソフトウェアの環境  
人も機械もさまざま

異なる結果





# バージョンと依存性地獄

# プログラムのバージョンには意味がある

メジャー

## APIの変更に

## 互換性のない場合など

# マイナー

## 後方互換性があり

## 機能性を追加した場合

パッチ

## 後方互換性を伴う

## バグ修正



# 3.10.12

※開発者・コミュニティによってバージョンの付け方が異なる

# 大事なこと

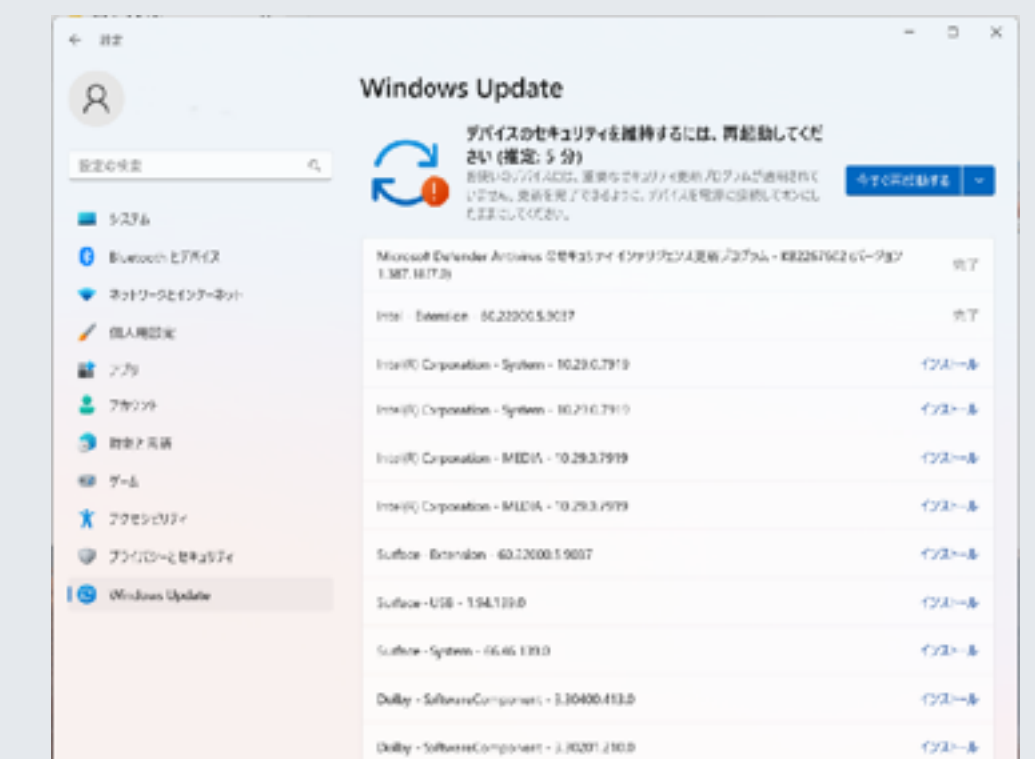
うかつに変えない… 波及効果、これまでのコードが機能しなくなるリスク  
更新内容をまとめたニュース等を見る。特にマイナー以上の変更  
最新版はバグがあることが当たり前。安定したバージョンを使う

OSではセキュリティ  
関係で重大な更新があ  
るのでさらに注意。そ  
の場合は素早く更新す  
る。

## iPhoneの例 (iOS)



## Windowsの例



# Jupyter

# Jupyter

# Project Jupyter

<https://jupyter.org/>

科学的研究における再現性とインタラクティブ性を高めるためのプロジェクト

## Jupyter Notebook

コードと実行結果、またそれらを補足する文章（画像等を含む）からなるファイルを作成するためのツール  
ノートブックファイルはJSON形式でメタデータ等を記録した .ipynb として扱われる→テキストファイル

Python, R, Juliaを始め、40以上の言語に対応

ウェブアプリケーションとしてウェブブラウザ上で動作する

Google Colaboratory (Colab)

Binder

Kaggle Notebooks

Cocalc

## JupyterLab

Jupyter Notebookのための優れた実行環境

タブ機能による複数ファイルの同時編集やファイル操作が柔軟に可能

→Jupyter Notebook単体で使うよりも操作性に優れる



# Google Colaboratory (Colab)

 <https://colab.research.google.com>

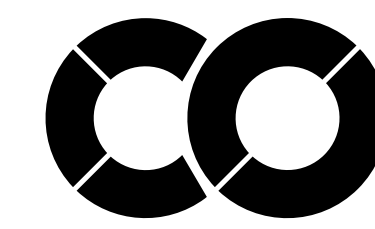
Googleが提供するJupyter Notebookの実行環境

Googleアカウントでログイン (メールアドレスの登録が必要)

Googleのサーバー上で動くため、実行環境の整備や違いを気にしなくて良い  
データサイエンス、機械学習で用いられる主要な環境が構築済み  
ノートブック、データ等のファイルをGoogleドライブと紐付けられる  
有償契約で高性能なGPU、TPUの利用が可能



# Google Colaboratory (Colab)



<https://colab.research.google.com>



右上の「ログイン」をクリック

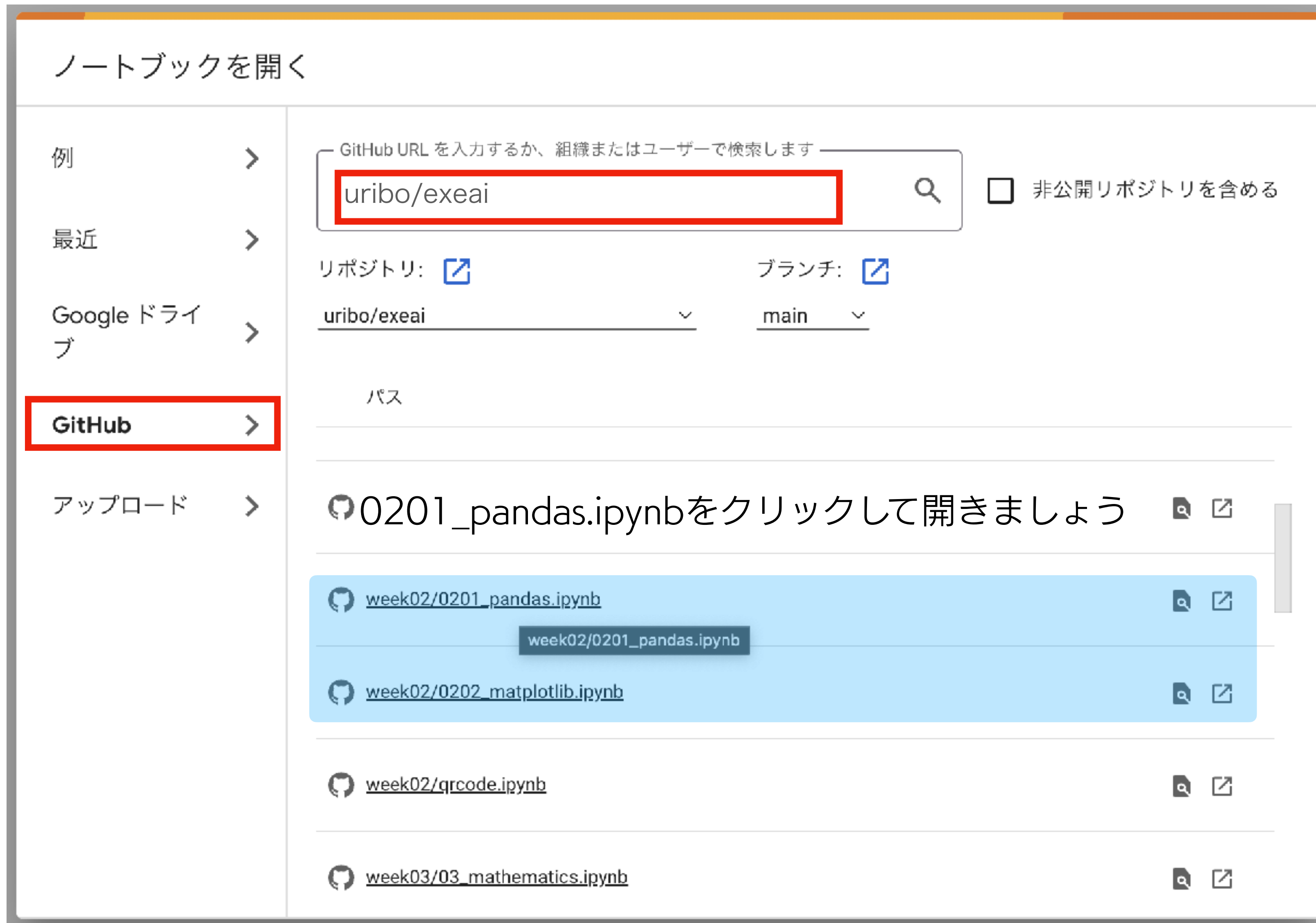


Googleアカウントでログイン

なければCアカウント等で  
アカウントを作成してください



# GitHub上のDipynbファイルを開く



ファイルを開くとこのような表示がでますが「このまま実行」を押してください

ここまでのまとめ



ここまでのまとめ

# これまでのまとめ

コンピュータ上で「何か」を行うためにプログラムを用いる

プログラムの実行には機械語が求められるが、「プログラミング言語」は人間が判読可能な形で記述し、翻訳によってコンピュータに命令を送る形式をとる

プログラミング言語に指示を記録しておくことで、繰り返しの処理や条件が変わった時の対処がしやすくなる

Jupyter Notebookの利用により、コードの内容と実行結果、さらに補足文章を加えることができる

ノートブックで実行するコードの記述は「コードセル」で行う→Python, R, Juliaを始め、40以上の言語に対応

Jupyter NotebookはColabやJupyterLabなどでも利用される

# Google Colaboratotyでの ipynbファイルの操作

# Google Colaboratotyでの ipynbファイルの操作

# コードセルとテキストセル

## コードセル

プログラムを記述するセル

実行後は [1]、[2] のような番号が  
セルの左端に表示される

(ipynb ファイル中での  
プログラムの実行順)

 実行     停止

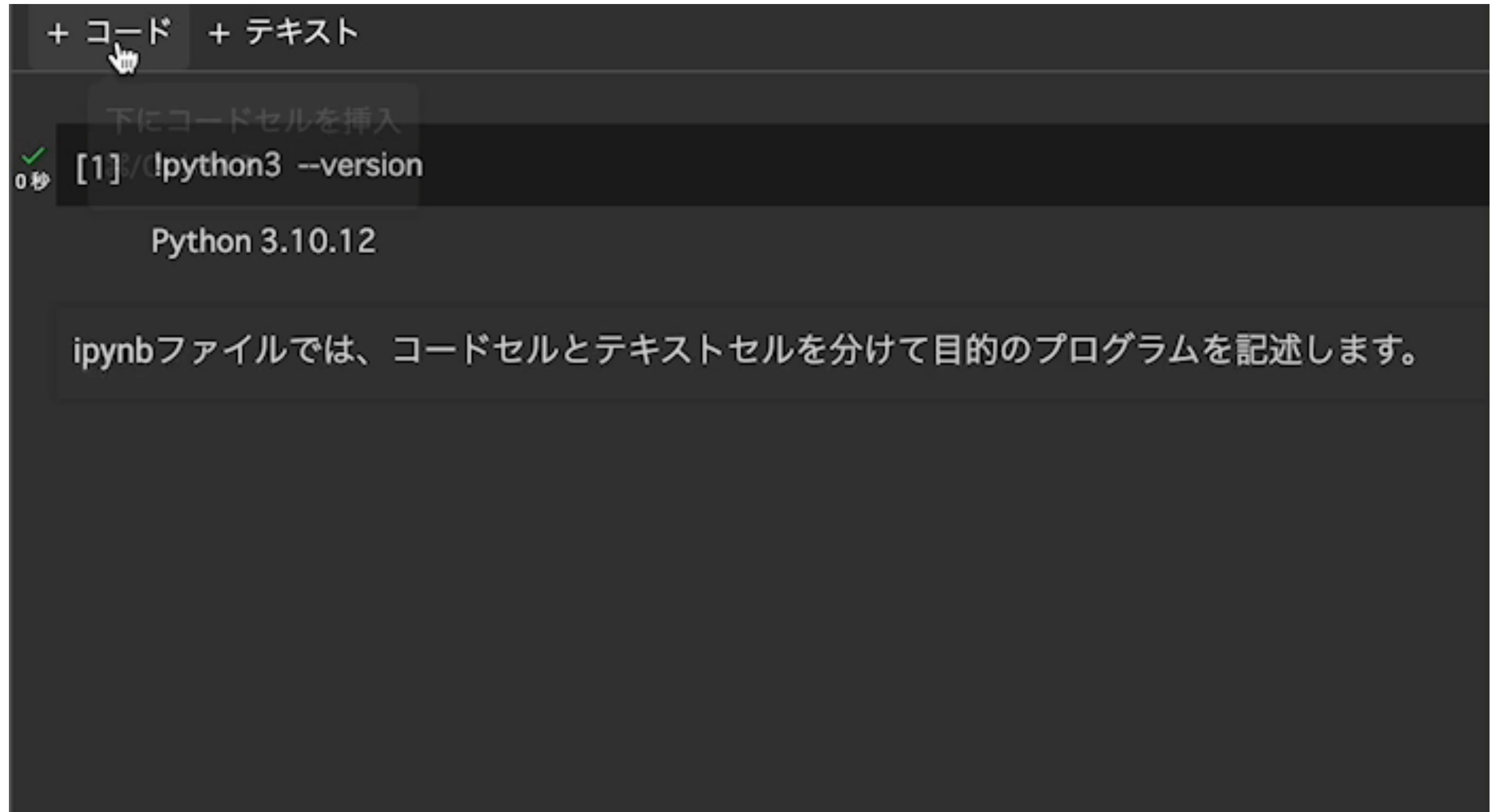
## テキストセル

プログラム以外のコメントや

図表の出力をおこなうセル

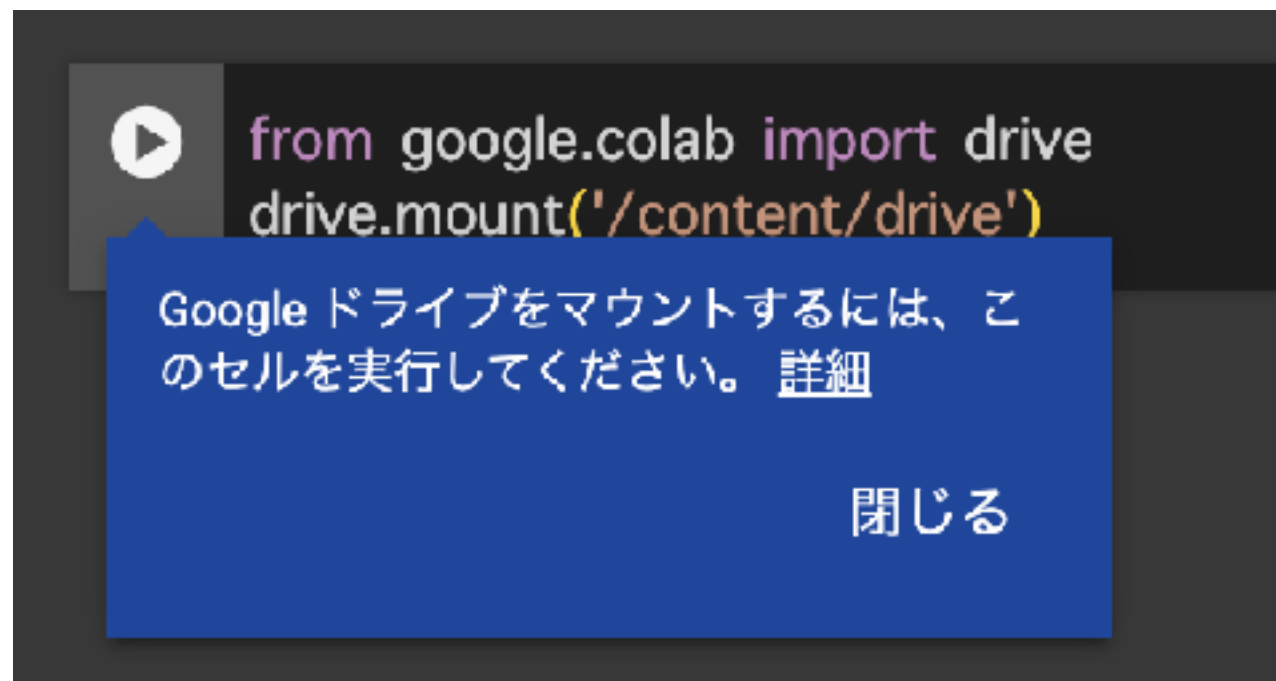
マークダウン記法による文字装飾

(見出し、太字、箇条書き等) が  
可能

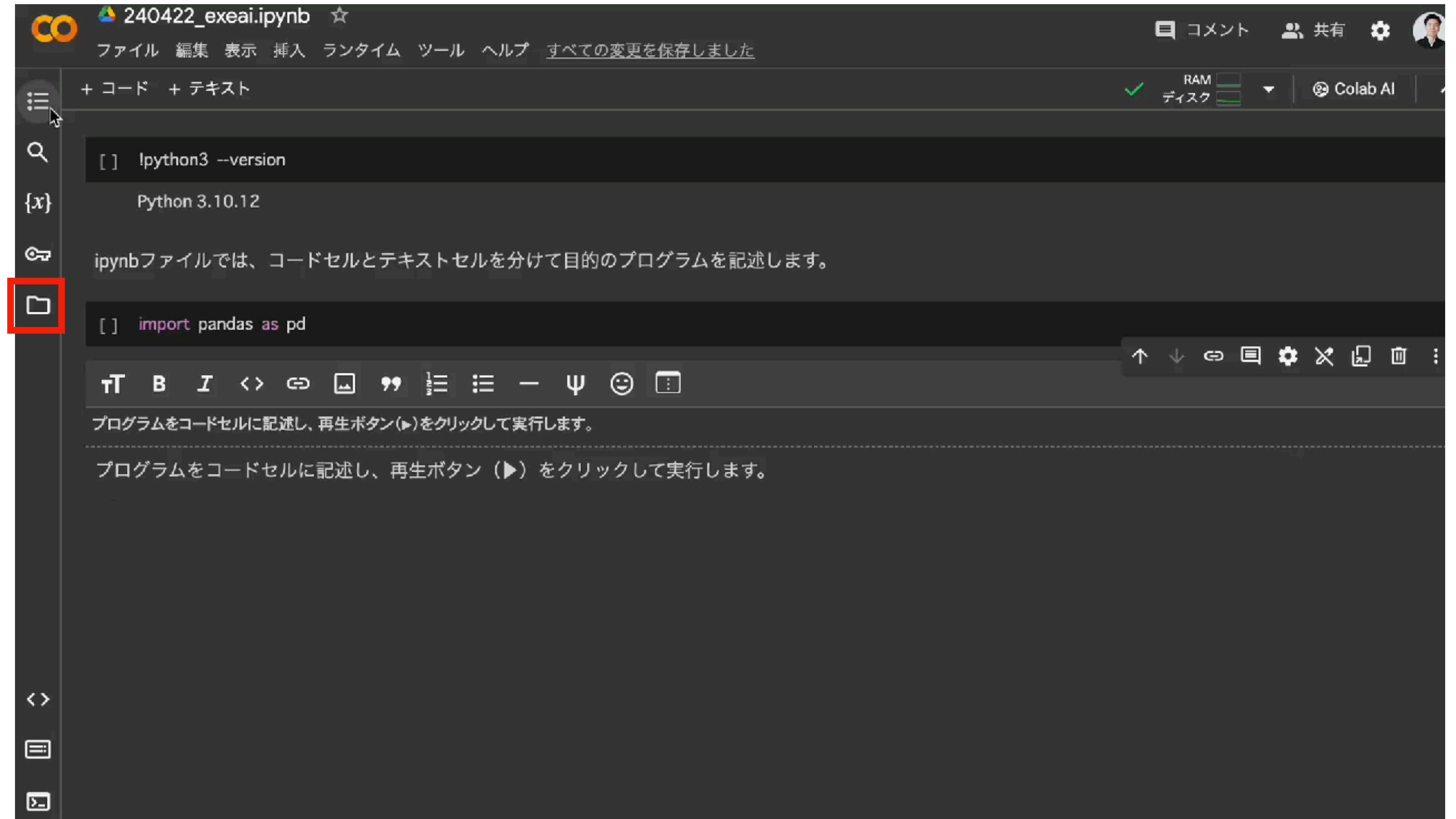


# Googleドライブのマウント（紐付け）

左側のアイコン群から  
フォルダのアイコンをクリック、  
左から3つ目の  
「ドライブをマウント」の  
アイコンをクリック



プログラムからも実行可能



# Python



汎用プログラミング言語の一つ

クロスプラットフォーム（Windows、macOS、Linuxなど）で動作

便利な関数をまとめた「ライブラリ」や

特定の目的に適した「フレームワーク」が豊富

機械学習・深層学習で使われる

ライブラリ（パッケージ）… NumPy, Pandas, Matplotlib, Scikit-Learnなど

フレームワーク… TensorFlow, PyTorch, Chainer, Hugging Face Transformersなど

ツール… Jupyter Notebook

TIOBE Index（プログラミング言語の人気ランキングの一種）で上位の常連

InstagramやYouTubeでも使われる

この講義ではバージョン3.10を利用



# Pythonを実行する場所（ソフトウェア環境）

テキストエディタ等でコードを記述して実行する

ターミナル、いわゆる?黒い画面

hello.py

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

```
~/Documents/teaching/23_exeai/sr  
> ../.venv/bin/python3 hello.py  
Hello, World!
```

## Jupyter Notebook ウェブブラウザ内で作動



## Visual Studio Code ファイル操作性や拡張機能が便利

### ハードウェアとソフトウェア

#### プログラムの実行に必要な要素

ハードウェア… コンピュータシステムの物理的な部分。実体をもつ。ソフトウェアが実行できるような資源やプラットフォームを提供する。

例) キーボード、ストレージ（HDDやSSD）、グラフィックカード、中央処理装置など



ハードウェアの例… キーボード

ソフトウェア… 具体的に何らかの処理を行うプログラムを指す。物理的には存在しない。ハードウェアを介してプログラムを実行する。

例) オペレーティングシステム（OS）、プリンタなどのドライバ、アプリケーションなど

# Pythonチュートリアル

.ipynbファイルをダウンロードしてColabへアップロード

1. 「ドキュメント等」から

「Jupyter hub 簡易操作説明」と

「上記チュートリアルを纏めたもの」を

クリックしてダウンロード

## Jupyter hub documentation (2022/09/27更新)

### • Jupyter hub 簡易操作説明

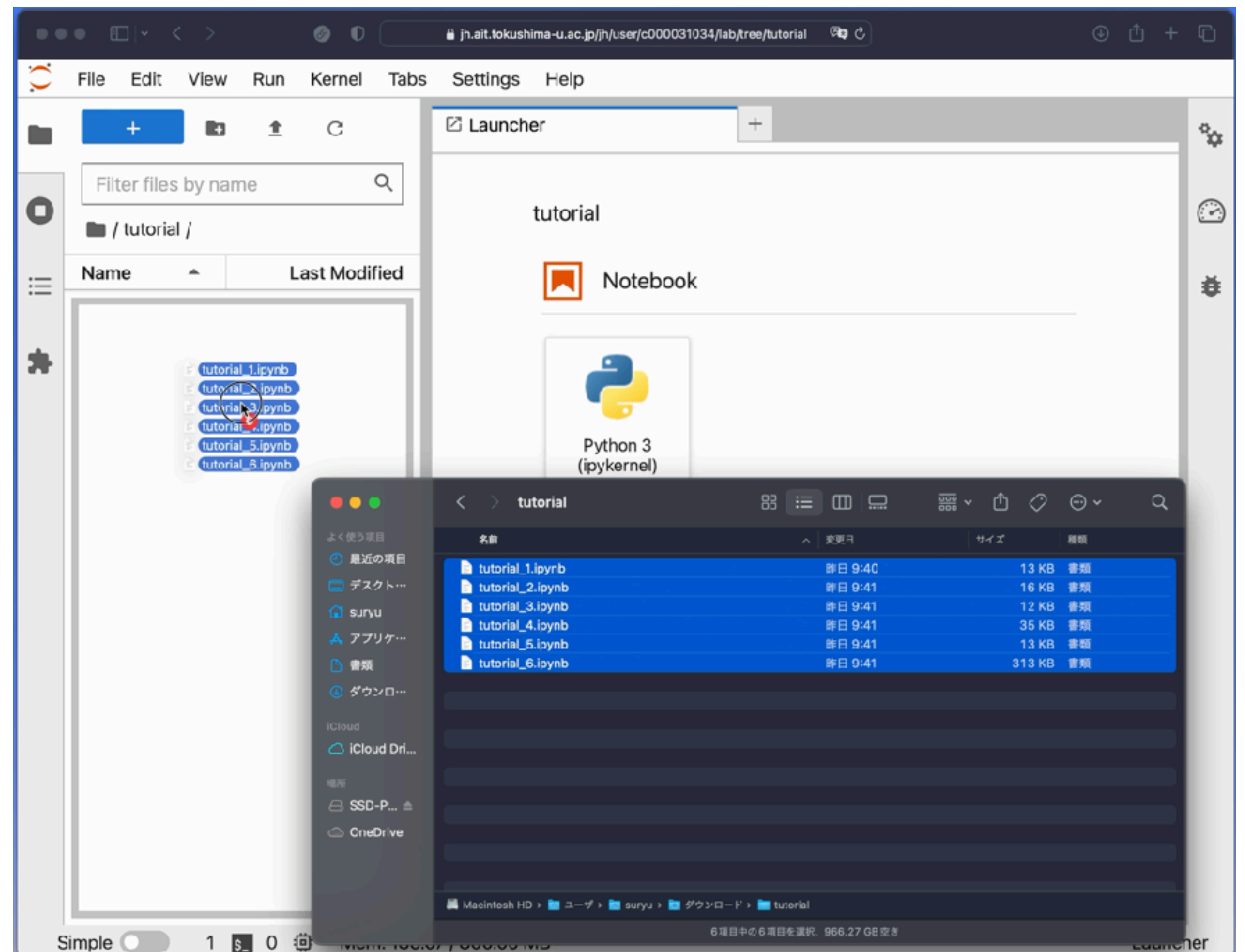
以下はPython学習向けのノートブックです。ダウンロードして、Jupyterhubにアップロードして利用してください。

右ボタンクリックで「名前をつけてリンクを保存」としてダウンロードしてください。

- [Python チュートリアル1：演算と代入](#)
- [Python チュートリアル2：数値や変数の型 \(2022/9/14更新\)](#)
- [Python チュートリアル3：比較とif 構文 \(2022/9/14更新\)](#)
- [Python チュートリアル4：繰り返し処理 \(2022/9/14更新\)](#)
- [Python チュートリアル5：関数 \(2022/9/14更新\)](#)
- [Python チュートリアル6：入出力 \(2022/9/14更新\)](#)
- [上記チュートリアルを纏めたもの \(2022/9/14更新\)](#)

2. ファイルを「ドラッグ&ドロップ」または

「矢印のアイコン↑をクリック」でアップロード



# 日本語（平仮名、片仮名、漢字）の扱い

文字列やコードとして処理されない「コメント」では日本語の使用も問題ない

```
# これはコメントです  
# Pythonでは、#から始まる行はコメントとして扱われます  
# コメントの行は実行されません  
  
# 行中に#がある場合は、#以降はコメントとして扱われます  
print("Hello, World!") # これもコメントです
```

変数名としても使えるが、原則として利用しない

ファイル名に日本語を使うのはOK。空白はハイフン (-) やアンダースコア (\_) を使う  
例) 241017第2回課題\_瓜生真也.ipynb

ファイル名はファイルの内容が何を示しているのかわかると良い（中身を開いて確認する手間を省く）



# パッケージによって必要な機能を追加する

Python Package Index (PyPI)から第三者が開発したパッケージをインストール可能  
pipと呼ばれるパッケージ管理のためのシステムがPythonに標準的に備わっている

例) plotlyの追加

<https://pypi.org/project/plotly/>

```
[1]: import plotly
```

-----  
-----  
ModuleNotFoundError  
(most recent call last)  
Cell In [1], line 1  
----> 1 import plotly  
  
ModuleNotFoundError: No module named 'plotly'

対象のモジュールが存在しない  
(インストールされていない) ためのエラー  
ModuleNotFoundError

## インストール

```
pip install plotly
```

## アンインストール


```
pip uninstall plotly
```

## 更新

```
pip install plotly --upgrade
```

AIによるプログラミング支援



# Gemini (Googleの対話型生成AI)

 240422\_exeai.ipynb ☆

ファイル 編集 表示 挿入 ランタイム ツール ヘルプ [すべての変更を保存しました](#)

コメント 共有 設定 ユーザー

+ コード + テキスト

RAM  ディスク  Colab AI

ノートブック Colab AI ×

Uryu Shinya

pandasでペンギンデータの体重に対する回帰分析を行いたいです。

Colab AI

```
import pandas as pd
import statsmodels.api as sm

# ペンギンデータの読み込み
df = pd.read_csv('penguins.csv')

# 体重と回帰直線の計算
x = df['body_mass_g']
y = df['flipper_length_mm']

x_const = sm.add_constant(x)
model = sm.OLS(y, x_const).fit()

# 回帰直線の傾きと切片
slope = model.params[1]
intercept = model.params[0]
```


ここにメッセージを入力

0 / 400

回答には、Google の考え方を反映していない、不正確または不適切な情報が表示されることがあります。 [詳細](#)

0 秒 完了時間: 1:35

# 分からないことをChatGPTに聞いてみよう



pipのインストールコマンドで`[`で囲まれているのは何を示しますか？



`pip`のインストールコマンドで`[`と`]`で囲まれている部分は、オプションの依存関係（extras\_require）を示しています。これにより、指定された追加機能に必要な追加のパッケージがインストールされます。

たとえば、`qrcode`ライブラリをインストールする際に、以下のコマンドを使用しました。

CSS

Copy code

```
pip install qrcode[pil]
```

このコマンドでは、`qrcode`パッケージに加えて、`pil`オプションに関連する追加の依存関係もインストールします。`pil`オプションは、Python Imaging Library（PIL）の後継であるPillowライブラリとの統合をサポートするためのものです。このオプションを指定することで、QRコードを画像として操作しやすくなります。

`[pil]`を指定しない場合、`qrcode`ライブラリの基本機能のみがインストールされ、Pillowとの統合がサポートされません。ただし、この例では画像としてQRコードを生成するため、`[pil]`オプションを使用してPillowとの統合を有効にすることが推奨されます。

🔄 Regenerate response

←Pythonの～、 pipの使い方を聞いてみた

# 【推奨】 GitHubの活用

## GitHub by Microsoft

プログラミングのソースコード置き場、開発者のためのプラットフォーム  
バージョン管理システムGitを用いたコード共有  
学生であれば有料プランを無料で利用可能（教育プログラム、要申請）

## GitHub Copilot

コメント、関数名等からコードの入力を自動補完するAIツール  
VSCodeの機能拡張としても利用可能  
↑ 上記の教育プログラムに加入で無料



# 【推奨】GitHubの活用

tidyverse.R U ●



tidyverse.R > ...

```
1  # 入力の際に表示されるグレーの文字は
2  # GitHub Copilotによる入力補完です
3  # このファイルはGitHub Copilotによって自動生成されました
4  library(tidyverse)
5  df <- read_csv("https://raw.githubusercontent.com/rstudio/c
6
7  ggplot(df, aes(x = reorder(country, -lifeExp), y = lifeExp))
    geom_col() +
    coord_flip() +
    labs(x = NULL, y = "Life Expectancy (years)", title = "Li
```

main\*+



0 0 1

スペース: 4

UTF-8

LF

R





# 参考資料・URL

- 📖 池内孝啓, 片柳薫子, @driller 『PythonユーザのためのJupyter〈実践〉入門（改訂版）』（2020）技術評論社. ISBN: 978-4-297-11568-5

瓜生居室: あり（電子版）、徳大図書館: なし、市立図書館: なし、県立図書館: なし

- 📖 有賀友紀, 大橋俊介 『RとPythonで学ぶ実践的データサイエンス&機械学習（増補改訂版）』（2021）講談社. ISBN: 978-4-297-12022-1

瓜生居室: あり（電子版）、徳大図書館: あり、市立図書館: あり、県立図書館: なし

- 📖 Joel Grus(著), 菊池彰(訳) 『ゼロからはじめるデータサイエンス：Pythonで学ぶ基本と実践（第二版）』（2020）オライリー・ジャパン. ISBN: 978-4-87311-911-3

瓜生居室: あり（電子版）、徳大図書館: あり、市立図書館: あり、県立図書館: なし

- 📖 Al Sweigart(著), 相川愛三(訳) 『退屈なことはPythonにやらせよう：ノンプログラマーにもできる自動化処理プログラミング（第二版）』（2023）オライリー・ジャパン. ISBN: 978-4-87311-927-4

瓜生居室: あり（電子版）、徳大図書館: あり（初版）、

市立図書館: あり（初版）、県立図書館: あり（初版）

- 📖 八谷大岳 『ゼロからつくるPython機械学習プログラミング入門』（2020）講談社. ISBN: 978-4-06-520612-6

瓜生居室: あり（電子版）、徳大図書館: なし、市立図書館: なし、県立図書館: なし

🔗 Python公式ドキュメント <https://docs.python.org/3/>

🔗 ゼロからのPython入門講座 <https://www.python.jp/train/index.html>

