

# PyTorch講座1

shun sato

# 本講座について

- おしらせ
  - 「実験3・演習3」です
  - 最後に課題が出るので毎週出席してください
  - Pythonの基本的な文法を抑えているという前提で進めます
- 目標
  - PyTorchを使ってCNNのトレーニングができるようになる
- 内容
  - 研究に必要なツールの使い方(Gitなど)
  - Pythonでのプロジェクト管理
  - PyTorchの基本的な使い方

# 環境構築・ツール導入

# 環境構築:コード管理

- **Git**

- <https://git-scm.com/downloads>
- バージョン管理ツール(詳細は後ほど)
- 全部デフォルト設定でインストールすればok

```
PS C:\Users\shun> git --version
git version 2.42.0.windows.2
PS C:\Users\shun> |
```

Gitインストール確認

- **GitHub**

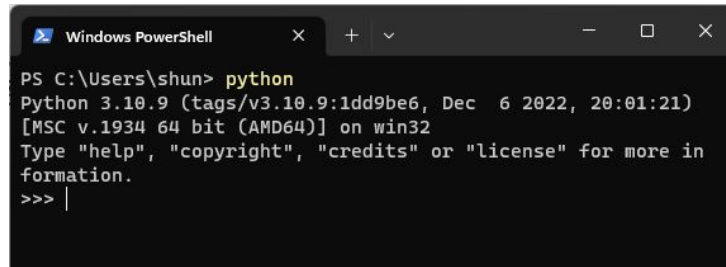
- <https://github.com/>
- リモートリポジトリサービス
- コードの保存をオンラインでできる

- **VSCode**

- <https://azure.microsoft.com/ja-jp/products/visual-studio-code>
- コードエディター
- 他に使いたいのが無ければこれを使う

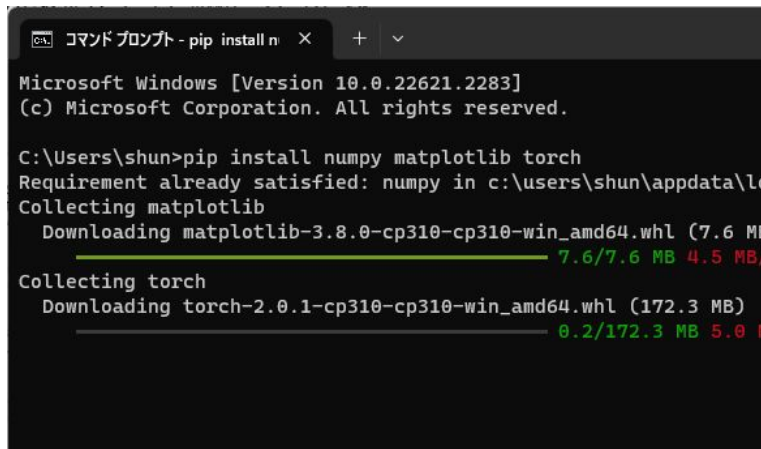
# 環境構築: Python

- **Python**: 適当なバージョンをインストール
  - <https://www.python.org/downloads/>
- **パッケージ**: 使いそうなものをインストール
  - numpy
  - matplotlib
  - torch



```
PS C:\Users\shun> python
Python 3.10.9 (tags/v3.10.9:1dd9be6, Dec 6 2022, 20:01:21)
[MSC v.1934 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more in
formation.
>>> |
```

Pythonインストール確認



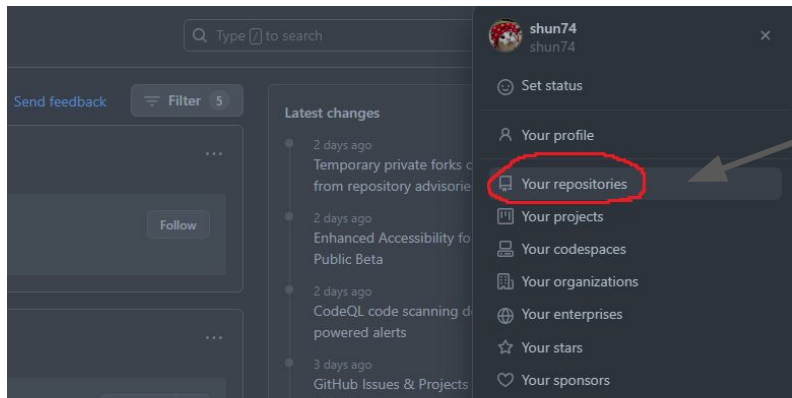
```
コマンド プロンプト - pip install n × + v
Microsoft Windows [Version 10.0.22621.2283]
(c) Microsoft Corporation. All rights reserved.

C:\Users\shun>pip install numpy matplotlib torch
Requirement already satisfied: numpy in c:\users\shun\appdata\lo
Collecting matplotlib
  Downloading matplotlib-3.8.0-cp310-cp310-win_amd64.whl (7.6 MB)
 7.6/7.6 MB 4.5 MB/s
Collecting torch
  Downloading torch-2.0.1-cp310-cp310-win_amd64.whl (172.3 MB)
 0.2/172.3 MB 5.0 MB/s
```

パッケージのインストール

# GitHubリポジトリを作る

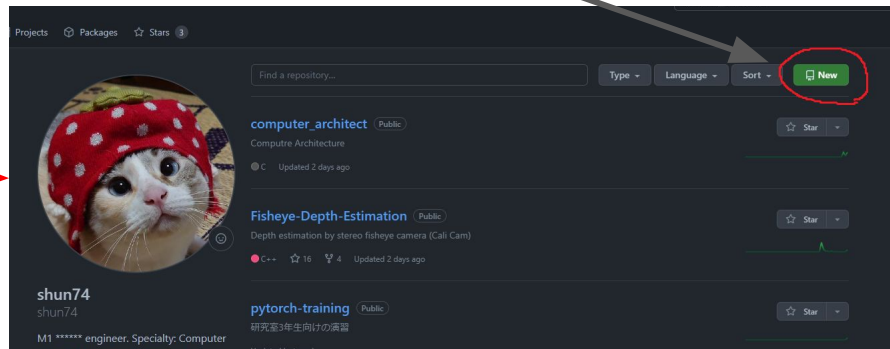
- 今回の演習でコードを保存するリポジトリを作る



ページ右上の自分のアイコンを押してタブを開く

“Your repositories”を押す

新しいリポジトリの作成



## Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Required fields are marked with an asterisk (\*).

Owner \*

shun74

Repository name \*

pytorch-training

⚠ The repository pytorch-training already exists on this account.

Great repository names are short and memorable. Need inspiration? How about [symmetrical-giggle](#) ?

Description (optional)

実験3演習3のコード保存用リポジトリ

☒



Public

Anyone on the internet can see this repository. You choose who can commit.

☐



Private

You choose who can see and commit to this repository.

Initialize this repository with:

☒

Add a README file

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore

.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)

Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

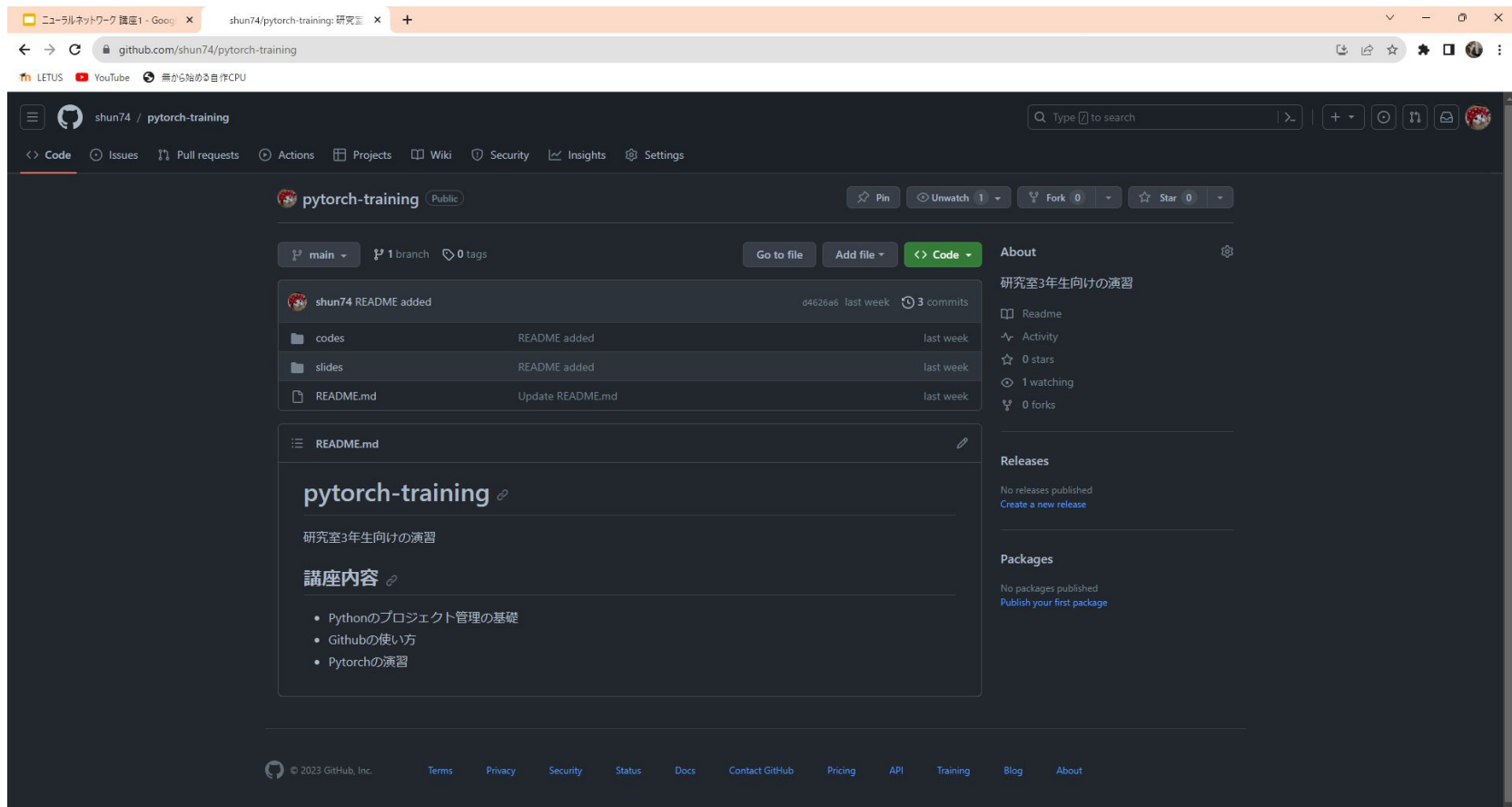
This will set `main` as the default branch. Change the default name in your [settings](#).

ⓘ You are creating a public repository in your personal account.

Create repository

- リポジトリ名は”pytorch-training”
- Descriptionは適当に書く
- Publicを選択
- Add a README fileを選択

全部できたら”Create repository”を押す

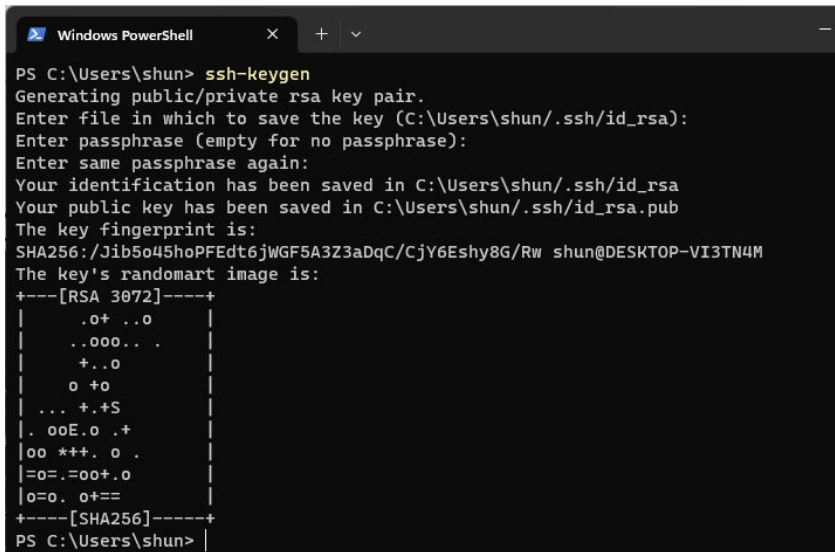


こんな感じになったら OK



# SSHキーの登録

- SSHを使ってリモートとデータをやり取りする
- まずはローカルでキーのペアを発行する

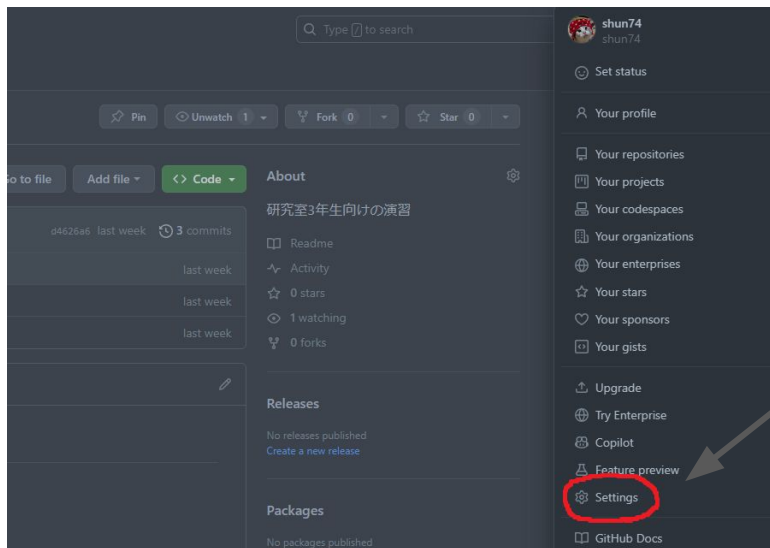


```
Windows PowerShell
PS C:\Users\shun> ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (C:\Users\shun/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in C:\Users\shun/.ssh/id_rsa
Your public key has been saved in C:\Users\shun/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:/Jib5o45hoPFEdt6jWGF5A3Z3aDqC/CjY6Eshy8G/Rw shun@DESKTOP-VI3TN4M
The key's randomart image is:
+---[RSA 3072]-----+
|      .o+  ..o      |
|     ..ooo.. .      |
|      +..o          |
|      o +o          |
|     ...+.+S        |
|.. ooE.o .+         |
|oo **+. o .         |
|=o=..=oo+.o         |
|o=o. o+=+          |
+-----[SHA256]-----+
PS C:\Users\shun> |
```

- “ssh-keygen”コマンドでキーを発行できる
- 何も指定しないとSHA256で発行される
- ファイル名やパスフレーズの指定が無ければEnterを何回か押すだけで○
- デフォルトで”~/.ssh”に”id\_rsa”(秘密鍵)と”id\_rsa.pub”(公開鍵)が生成される

```
Windows PowerShell
PS C:\Users\shun> cat ~/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCwV+SI5XCrrreE4df+vh08si30cgs1HFydDnUTzYe3MjMD7yT
BY9+LIAC4dOguyumDRtUKGvSP6sqIHala8q0B4L7JID06SHMB5eFqcLFeblouP22dU5+YH5vzUwLnxZ4ixr2T0
+tak/LElgr9sncY48Y2XfqKpkMrTx7X4H1PFQen+FQ76avCPbtFQisBBQ0VXwMtAk+Nnh4IvC5k+TdfGsc/KZDZ
rbJKambVNsD6A3SE2shtk3HYKpY0mHmtwTrbDjdflgC+qnTtOzWdpokWIhLUTxy7ibDK2pnNqGpvL09z27oAbOB
sYksfqGV9hgKr1vRX+138bxBQJWfyEQd6h49LeZ6ZtfIAwiBFyv47f0IY8qiSzZTvPLYCD1xy8HuFPA/9aKHLp
ORZDkHKjqwRV/8m8z3J/FNGjGDQcc00lm1AyFo0AdWHsr4+8kZFG4+gF7T0PMq2ZFKR0wdcwE6Hc2dy9Y2GIV+V
RRgdvyjp4LvkOLupYnNKqQFcygMV= shun@DESKTOP-VI3TN4M
PS C:\Users\shun> |
```

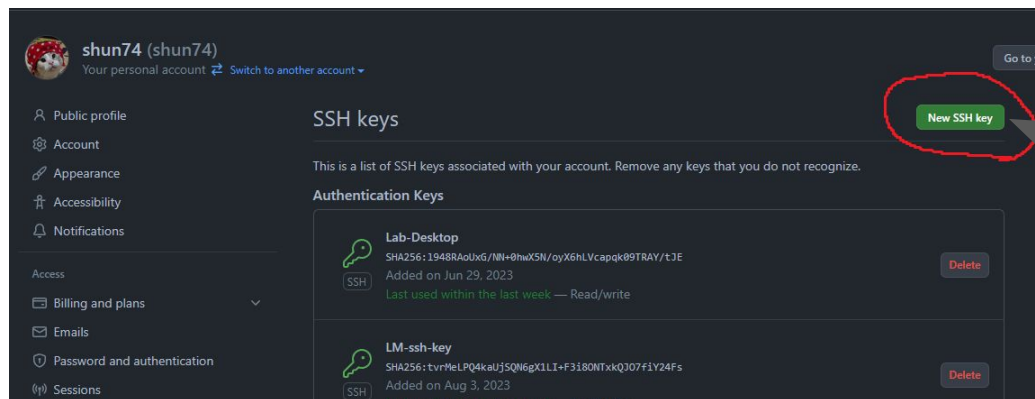
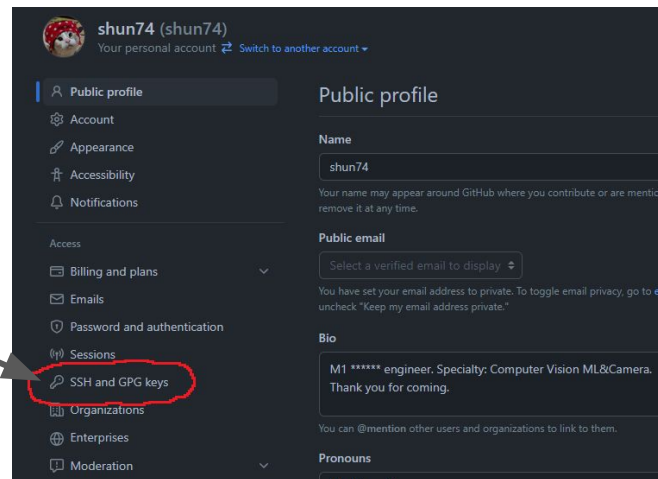
- 公開鍵の中身を確認
- これをコピーする



ブラウザのGithubで自分のアイコンをクリック

“Settings”を押す

“SSH and GPG keys”を押す



“new SSH key”を押す

shun74 (shun74)  
Your personal account [Switch to another account](#) [Go to y](#)

- Public profile
- Account
- Appearance
- Accessibility
- Notifications

Access

- Billing and plans
- Emails
- Password and authentication
- Sessions
- SSH and GPG keys**
- Organizations
- Enterprises
- Moderation

Code, planning, and automation

## Add new SSH Key

**Title**  
Laptop-key

**Key type**  
Authentication Key

**Key**

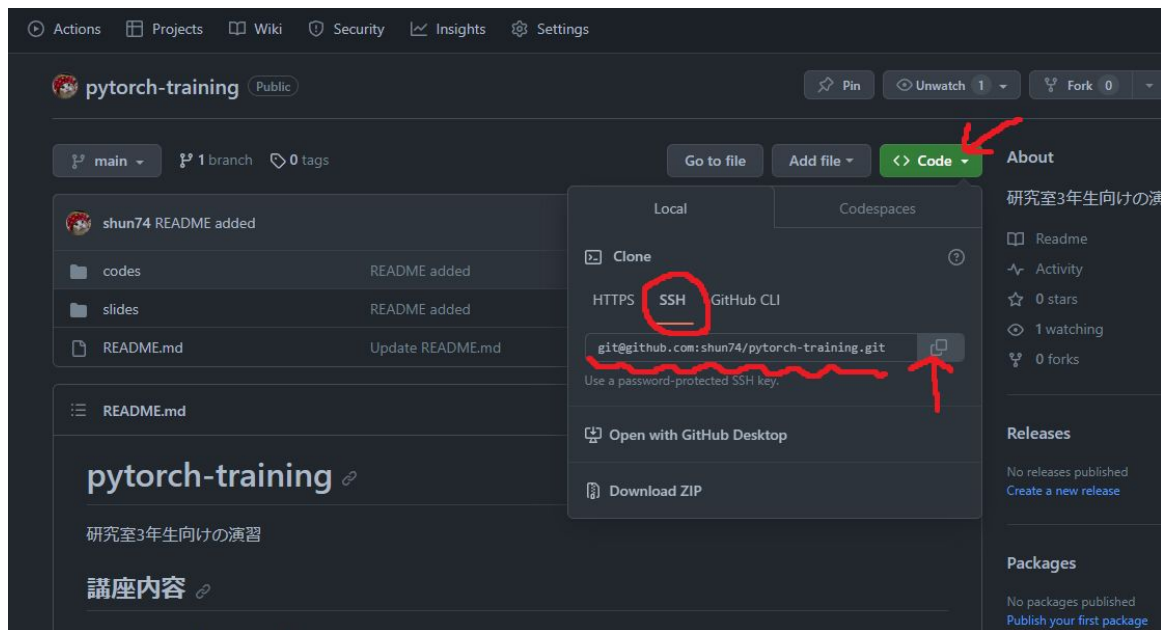
```
ssh-rsa
AAAAAB3NzaC1yc2EAAAADAQABAAQGCwV+SISXCrrreE4df+vhO8si3Ocs1HFydDnUTzYe3MImD7yTBY9+IIAC4dOguy
umDRTUKGvSP6sqlHala8q0B4L7JID06SHMB5eFQcLFeblouP22dU5+YH5vzUWLnXZ4ixr2T0+taK/LElgr9sncY48Y2XfqKpk
MrTx7X4H1PFQen+FQ76avCPbtQisBBQ0VXwMtAk+Nnh4lvC5K+TDfGsc/KZDZrbJKambVNsD6A3SE2shtk3HYKpYOMHM
twTrbDjdflgC+qnTtOzWdpokWihLUTxy7ibDK2pnNqGpvlO9z27oAbOBsYksfqGV9hgKr1vRX+138bx8QJWfyEQd6h49IEzE
6ZtflAwiBFy47fOIY8qiSzZTvPIYCD1xy8HufPA/9aKHLPORZDkHKjqwRV/8m8z3J/FNGjGDQcc00Im1AyFoOAdWHsr4+8kZF
G4+gF7TOPMq2ZFKROwdcwE6Hc2dy9Y2GIV+VRRgdvyjp4LvKOLupYnNKqQOfcygMV= shun@DESKTOP-VI3TN4M
```

Add SSH key

- “Title”は適当な名前をつける
- “Key”に公開鍵をコピーする
- 最後に”Add SSH key”を押す

# リポジトリのクローン

- リモートのリポジトリを取得する

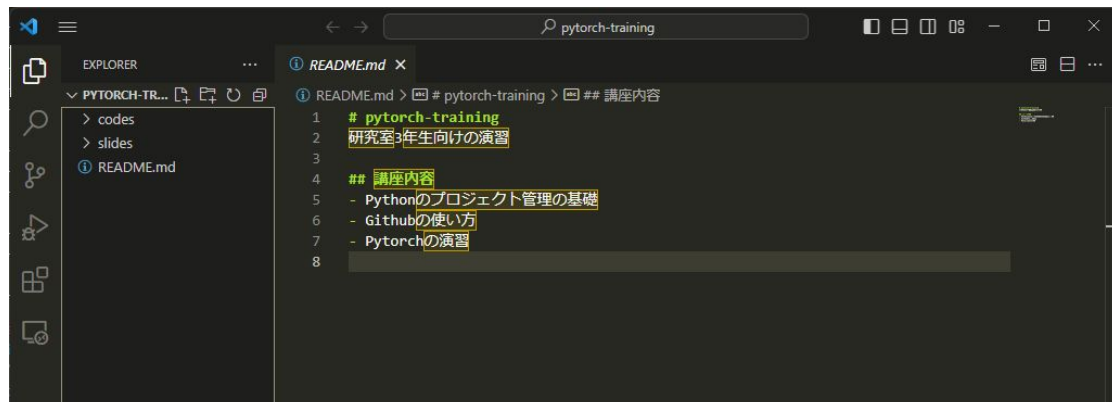


- “Code”を押してタブを開く
- “SSH”を押す
- URLをコピーする

```
Windows PowerShell
PS C:\Users\shun> cd ~
PS C:\Users\shun> git clone git@github.com:shun74/pytorch-training.git
Cloning into 'pytorch-training'...
The authenticity of host 'github.com (20.27.177.113)' can't be established.
ED25519 key fingerprint is SHA256:+DiY3wvvV6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'github.com' (ED25519) to the list of known hosts.
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (6/6), done.
Receiving objects: 100% (12/12), done.
Resolving deltas: 100% (1/1), done.
remote: Total 12 (delta 1), reused 6 (delta 0), pack-reused 0
```

- “cd ~”でホームディレクトリに移動
- “git clone コピーしたURL”でリポジトリをクローンする

- VSCodeで開いて中身を確認する
- これで準備完了



# Gitの概要



- バージョン管理＝コードの**変更履歴**を記録する
  - リモート:コードを保存しているサーバーのこと
  - ローカル:コードを編集している手元のPCのこと
- **ブランチ機能**による作業の分岐
  - Githubでは最初に**mainブランチ**が作られる
  - ブランチを複数作成することで異なる機能を同時に開発できる
  - 機能が完成したタイミングで**マージ**を行ってブランチ間のコードを統合する
- **メリット**
  - 特定の機能の実装前までコードを戻すことができる
  - 複数人で独立した機能開発を行うのに便利

# Gitの機能を使う

- 以下のディレクトリ構造でファイルを作成する

```
/pytorch-training  
└─ /01  
    └─ test.py
```

- test.pyは適当でいいので何か書き込んでおく. print("Hello")とか



## 今回使うコマンド

- “git status”:  
現在のリポジトリの状態を確認
- “git add *path*”:  
*path*のファイルを追跡に追加  
(今回はワイルドカード)
- “git commit -m “*message*””:  
*message*を付けて変更をコミット  
コミット: 変更を記録する操作
- “git push *dst branch*”  
*dst*の*branch*にローカルのコミットをプッシュ  
プッシュ: 手元の変更をサーバーに送信する

”origin”はGitHub上のリモートサーバー

プッシュしたらブラウザ上で GitHubのリポジトリに変更が追加されているかを確認する

```
選択Windows PowerShell
PS C:\Users\shun\pytorch-training> git status
On branch main
Your branch is up to date with 'origin/main'.

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    01/

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\shun\pytorch-training> git add *
PS C:\Users\shun\pytorch-training> git status
On branch main
Your branch is up to date with 'origin/main'.

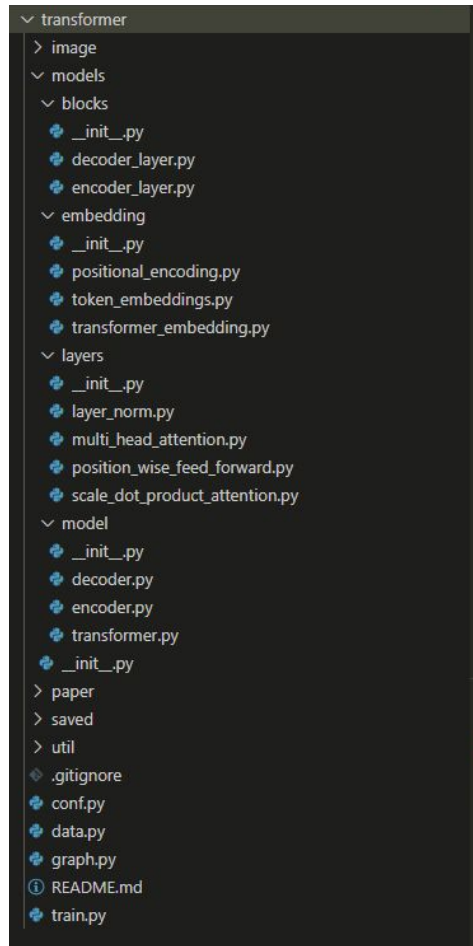
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    new file:   01/test.py

PS C:\Users\shun\pytorch-training> git commit -m "test commit"
[main d183d19] test commit
1 file changed, 1 insertion(+)
create mode 100644 01/test.py
PS C:\Users\shun\pytorch-training> git push origin main
```

# Pythonの基礎

# Pythonでのプロジェクト管理

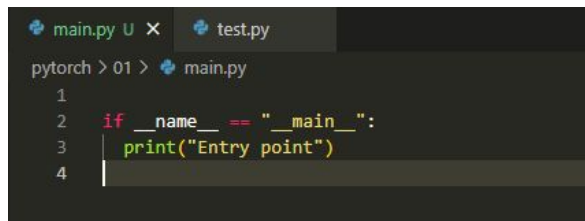
- ファイルを分割して実装する
  - 実行ファイル
  - 機能実装ファイル
- 機能をファイルごとにまとめる
  - 関数の作成
  - クラスの作成
- なぜファイル分割するのか
  - **大規模なプロジェクト**を扱うため
  - 機械学習のプロジェクトは大規模なものが多い



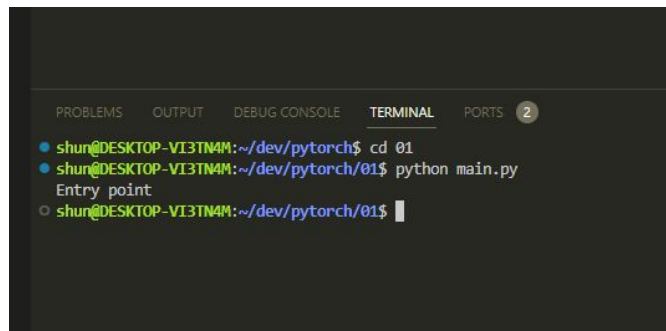
transformerという言語モデルのプロジェクト  
<https://github.com/hyunwoongko/transformer>

# Pythonの基礎

- Pythonのプログラムファイル
  - ○○.py
- ファイルの実行方法
  - **VSCode右上の「▶」ボタンは絶対に使わない**
  - “python *file\_path*”



```
main.py U x test.py
pytorch > 01 > main.py
1
2 if __name__ == "__main__":
3     print("Entry point")
4
```

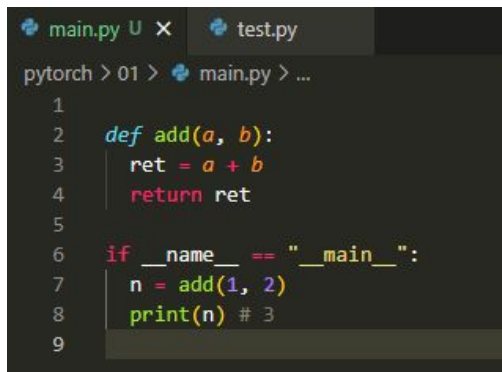


```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS 2
shun@DESKTOP-VI3TN4M:~/dev/pytorch$ cd 01
shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$ python main.py
Entry point
shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$
```

main.pyを実行する例

# 関数の定義

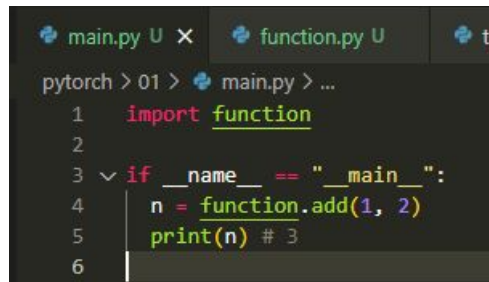
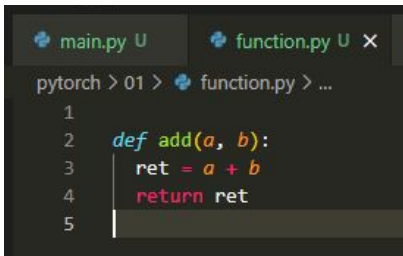
- `def func_name(arg1, arg2, ...):`
- 関数を使って機能を追加していく



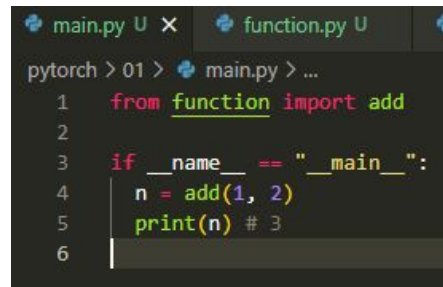
```
main.py U X test.py
pytorch > 01 > main.py > ...
1
2 def add(a, b):
3     ret = a + b
4     return ret
5
6 if __name__ == "__main__":
7     n = add(1, 2)
8     print(n) # 3
9
```

# Pythonのimport

- function.pyを作成



1. ファイルの名前をimport



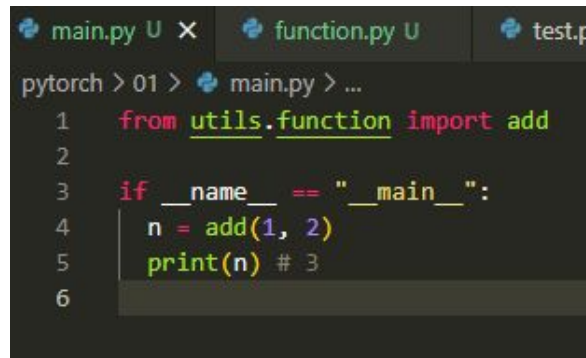
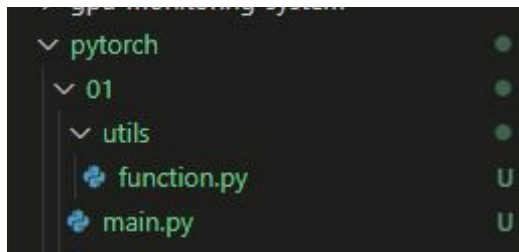
2. 関数を直接import

※ここで紹介しているのは「絶対import」で、他にも「相対import」が存在する。

# Pythonのimport

- 子フォルダからのimport方法

/01  
├── *utils*  
│ └── *function.py*  
└── *main.py*



名前を繋げてimport可能

# エントリーポイント

- エントリーポイント: プログラムの実行が開始される場所

```
main.py U  function.py U ×
pytorch > 01 > utils > function.py > ...
1
2 print("function.py is imported")
3
4 def add(a, b):
5     ret = a + b
6     return ret
7
```



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS 3
● shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$ python main.py
function.py is imported
3
○ shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$
```

- `if __name__ == "__main__":`:
  - 実行ファイル以外だと実行されない

```
main.py U  function.py U ×
pytorch > 01 > utils > function.py > ...
1
2 def add(a, b):
3     ret = a + b
4     return ret
5
6 if __name__ == "__main__":
7     print("function.py is imported")
8
```



```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS 3
● shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$ python main.py
3
○ shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$
```



演習

# 演習

- `count_word`を実装する
  - 文字列に含まれているアルファベットの数を数える関数



utils/count.pyを作成  
count\_wordをcount.pyに実装する

```
main.py U ×  count.py U  function.py U
pytorch > 01 > main.py > ...
1  from utils.count import count_word
2
3  if __name__ == "__main__":
4      s = "hello world"
5      print("e:", count_word(s, 'e')) # 1
6      print("o:", count_word(s, 'o')) # 2
7      print("l:", count_word(s, 'l')) # 3
8
```

main.pyの内容

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  3
● shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$ python main.py
e: 1
o: 2
l: 3
○ shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$
```

# 演習：追加問題

1. function.pyに`add`関数を実装
  - a. count.pyに`add`関数をimportして`count_word`関数で使う
2. `word_count`関数の入力を`assert`句でチェックする
  - a. 第一引数が`str`型かチェックする
  - b. 第二引数が`str`型かつ長さが1かチェックする

※`assert`の使い方: <https://note.nkmk.me/python-assert-usage/>  
`isinstance`の使い方: <https://note.nkmk.me/python-type-isinstance/>

```
main.py U × count.py U function.py U
pytorch > 01 > main.py > ...
1  from utils.count import count_word
2
3  if __name__ == "__main__":
4      s = "hello world"
5      print("e:", count_word(s, 'e')) # 1
6      print("o:", count_word(s, 'o')) # 2
7      print("l:", count_word(s, 'l')) # 3
8      count_word(s, "aaa")
9
```

```
shun@DESKTOP-VI3TN4M:~/dev/pytorch/01$ python main.py
e: 1
o: 2
l: 3
Traceback (most recent call last):
  File "/home/shun/dev/pytorch/01/main.py", line 8, in <module>
    count_word(s, "aaa")
  File "/home/shun/dev/pytorch/01/utils/count.py", line 5, in count_word
    [AssertionError]
AssertionError
```

assertでエラーが出ている例