

신입 학부 연구생을 위한 개발 환경 셋업 가이드 (Windows ver.)

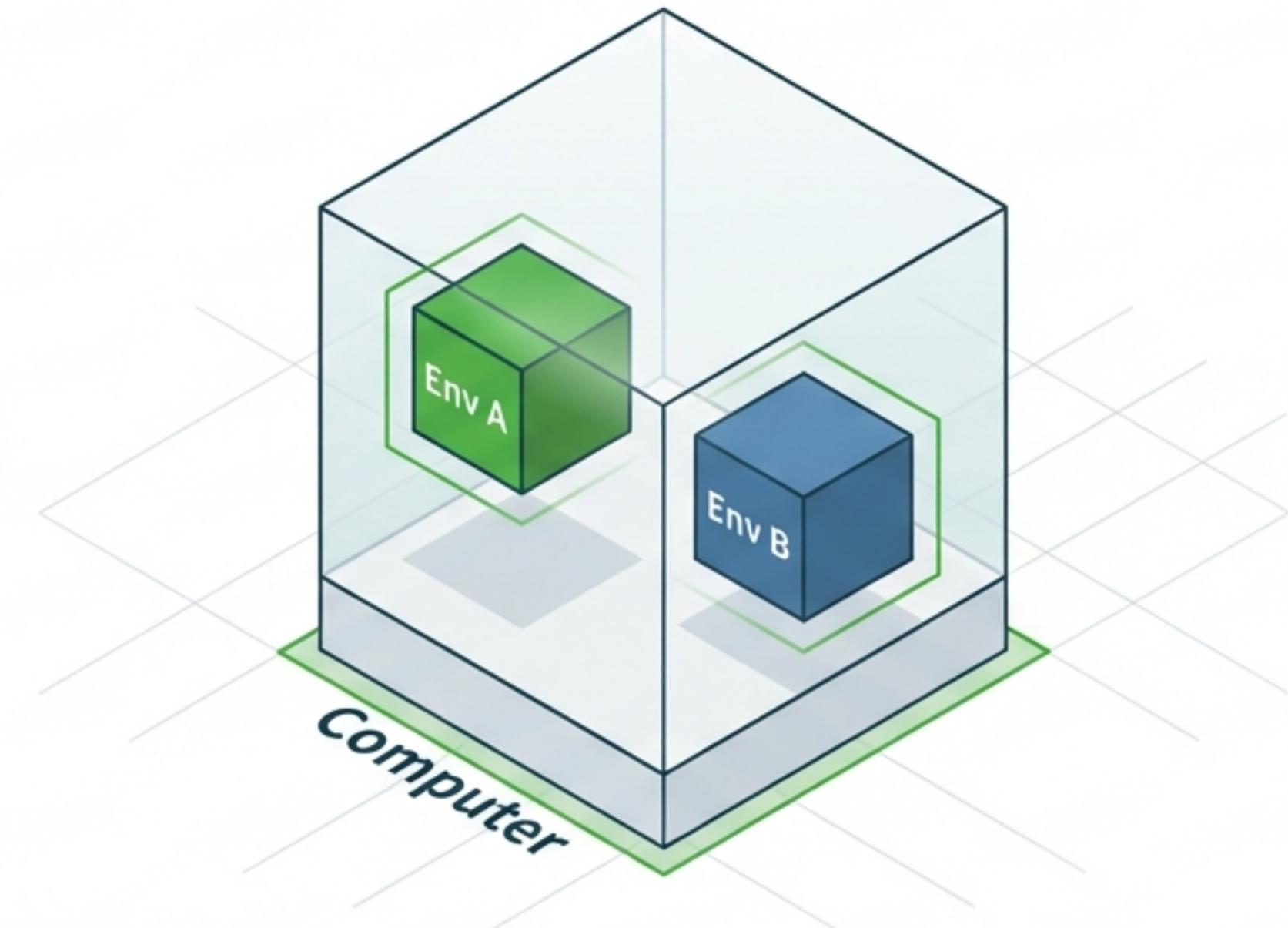
Anaconda, IDE, Git을 활용한 효율적인 연구 워크플로우 구축하기



본 가이드는 단순한 설치 매뉴얼이 아닙니다.
연구 데이터를 보호하고, 협업 효율을 극대화하며,
재현 가능한 연구를 수행하기 위한 ‘프로페셔널 툴킷’을 구축하는 과정입니다.

기반 공사: 왜 Anaconda인가?

- **가상환경 (Virtual Environment)**
연구 프로젝트마다 필요한
연구 프로젝트마다 필요한 라이브러리와
버전이 다릅니다. A 프로젝트는
Python 3.7, B 프로젝트는 Python 3.9가
필요할 수 있습니다.
Anaconda는 이들이 서로 충돌하지
않도록 독립된 “방(가상환경)”을
만들어줍니다.



Included Batteries



NumPy: 행렬/배열 연산



pandas: 데이터 분석 (Dataframe)



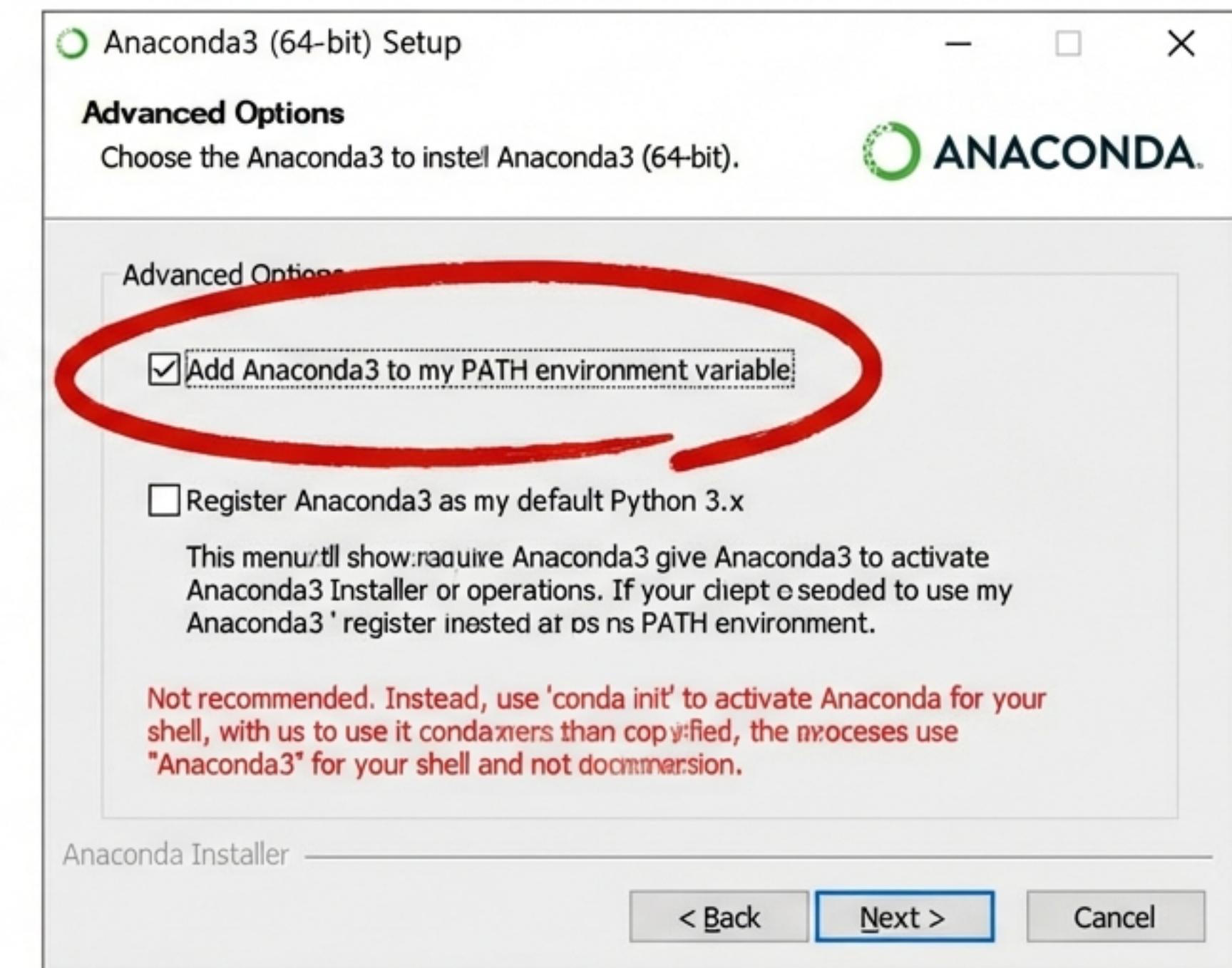
matplotlib: 데이터 시각화



Scikit-learn: 머신러닝 학습
NotebookLM

Anaconda 설치: 이 한 번의 클릭이 가장 중요합니다

1. 공식 홈페이지(anaconda.com)에서 Windows 버전 다운로드 및 설치 시작.
2. [중요] 설치 옵션 중 ‘Add Anaconda3 to my PATH environment variable’ 체크 필수.
3. 이유: 이 옵션을 켜야 윈도우 터미널(CMD, PowerShell) 어디서든 파이썬과 Conda 명령어를 실행할 수 있습니다.



나만의 연구 공간(가상환경) 만들기

Anaconda Prompt 혹은 터미널에서 다음 명령어를 사용합니다.

Create

```
C:\Users\User> conda create --name research_lab python=3.8
Collecting package metadata (current_repodata.json): done
Solving environment: done
# ...
Proceed ([y]/n)? y
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
C:\Users\User>
```

특정 파이썬 버전(3.8)을 지정하여
'research_lab'이라는 이름의 격리된
공간을 생성합니다.

Activate

```
(base) C:\Users\User> conda activate research_lab
(research_lab) C:\Users\User>
```

생성한 방으로 들어갑니다. 터미널
앞부분이 (환경이름)으로 바뀝니다.

List Check

```
(research_lab) C:\Users\User> conda env list
# conda environments:
#
# base                  * C:\Users\User\anaconda3
research_lab            C:\Users\User\anaconda3\envs\research_lab
C:\Users\User>
```

현재 내 컴퓨터에 존재하는 모든
가상환경 목록을 확인합니다.

작업 공간 구축: IDE(통합 개발 환경)가 필요한 이유

“ 메모장으로 코딩하는 것은 맨손으로 집을 짓는 것과 같습니다. IDE는 전기톱, 드릴, 수평계가 갖춰진 전문 작업대입니다.”



자동 완성 (Autocomplete)
오타 방지 및 속도 향상



디버깅 (Debugging)
실행 과정을 멈춰보며 오류 탐색



프로젝트 관리
다수의 파일을 체계적으로 관리

The Old Way vs The Pro Way

PyCharm

파이썬 전용, 강력한 기능.
데이터 분석 및 대규모 프로젝트에 적합.

VS Code

가볍고 빠름, 다양한 확장 프로그램.
범용성 및 웹 개발 병행 시 적합.

핵심 설정: IDE와 Anaconda 가상환경 연결하기

IDE가 방금 설치한 Anaconda 가상환경(도구함)을 인식하도록 '인터프리터(Interpreter)'를 설정해야 합니다.



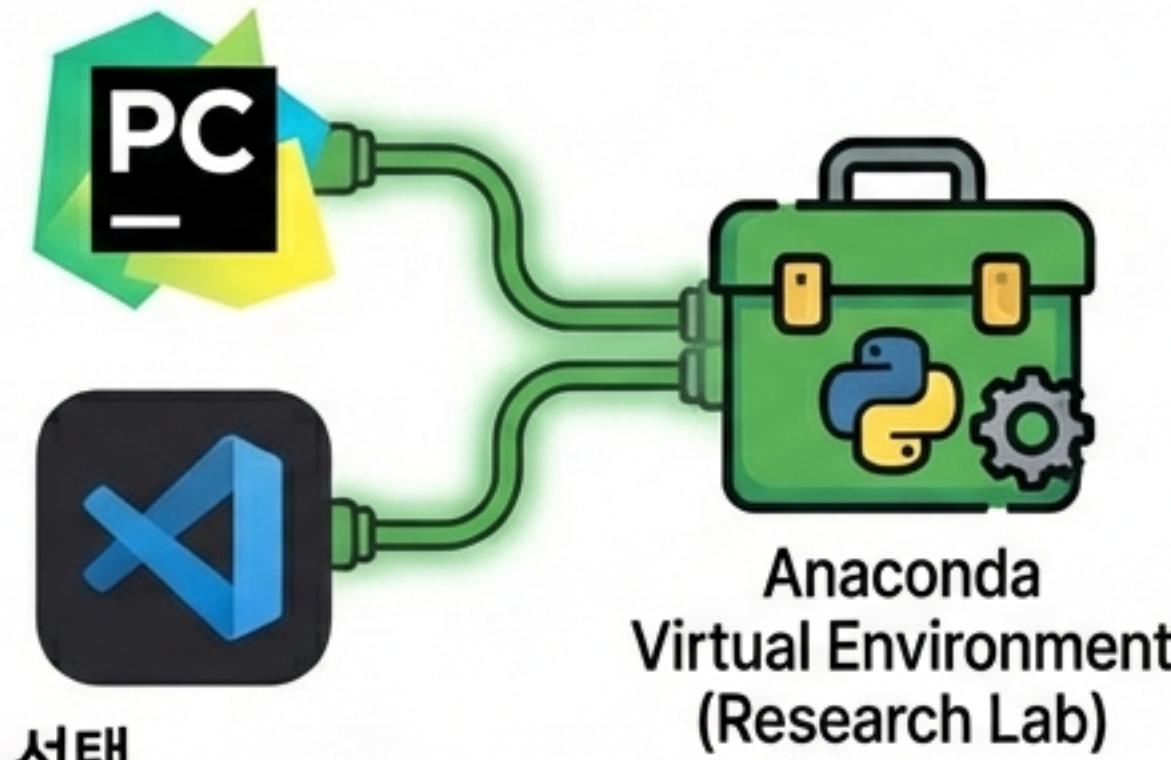
PyCharm Setup

- 1 File > Settings > Project > Python Interpreter

- 2 Add Interpreter
> Conda Environment
> Existing environment

- 3 Anaconda 설치 경로의 **python.exe** 선택

C:\Users\User\anaconda3\envs\research_lab\python.exe



VS Code Setup

- 1 확장(Extensions) 탭에서 Python 설치



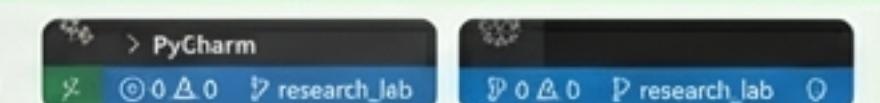
- 2 Ctrl + Shift + P > Python: Select Interpreter
입력

- 3 목록에서 ('환경이름': conda)
가 적힌 항목 선택



Check Point

Verification: 하단 상태 표시줄에 선택한 가상환경 이름이 뜨는지 반드시 확인하세요.



'print()' 도배는 그만! 디버깅(Debugging) 기초

```
8 > def calculate(data):
9     index = 0
10    result = []
11    for item in data:
12        # Debugging point
13        if item > 10:
14            result.append(item * 2)
15            index += 1
16    return result
```

1. Break Point:
줄 번호 옆을 클릭해
빨간 점 찍기.

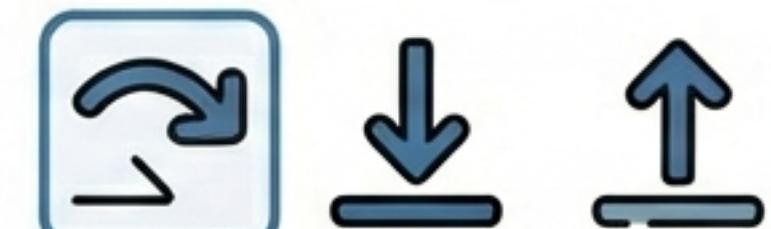


2. Debug Run:
벌레 모양 아이
이콘(Debug) 클릭.

Debug Variables

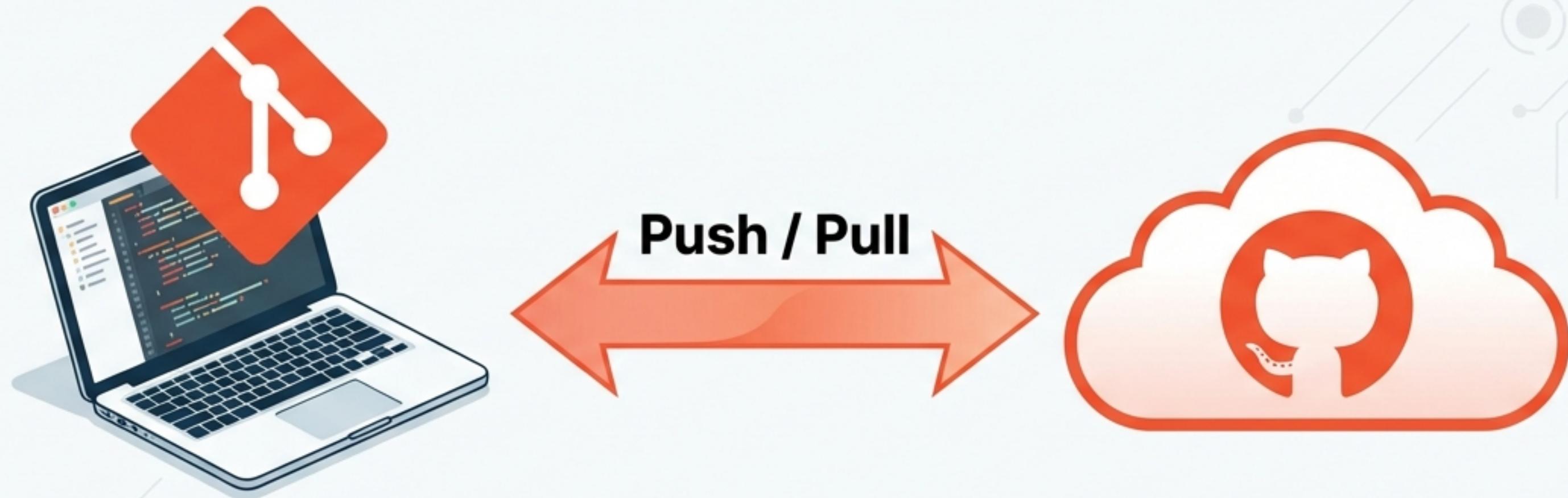
+	> data = [5, 12, 8, 20, 3]
-	index = 1
▼	> result = [24]
	item = 12

3. Variables:
멈춘 시점의
변수 값 실시간 확인.



4. Step Over:
한 줄씩 실행하며
로직 흐름 추적.

협업과 기록: Git & GitHub 개념 잡기



Git (타임머신)

내 컴퓨터(Local)에서 소스 코드의 **변경 이력**을 저장.
언제든 과거 시점으로 되돌릴 수 있음.

GitHub (클라우드 저장소)

저장한 기록을 온라인(Remote)에 업로드.
팀원과 공유 및 안전한 백업.

Git 설치 및 저장소(Repository) 만들기

Step 1: Install



Git for Windows
검색 및 설치
(기본 설정 유지).

Step 2: Create Repo



New Repository

Repository name: [프로젝트 이름]

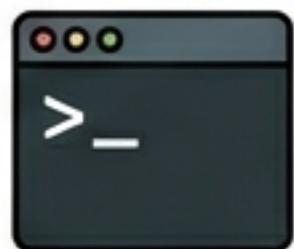


Public / Private 선택



[Tip] .gitignore 설정: 언어 설정에서 'Python'을 선택하면 불필요한 파일 업로드를 방지합니다.

Step 3: Clone

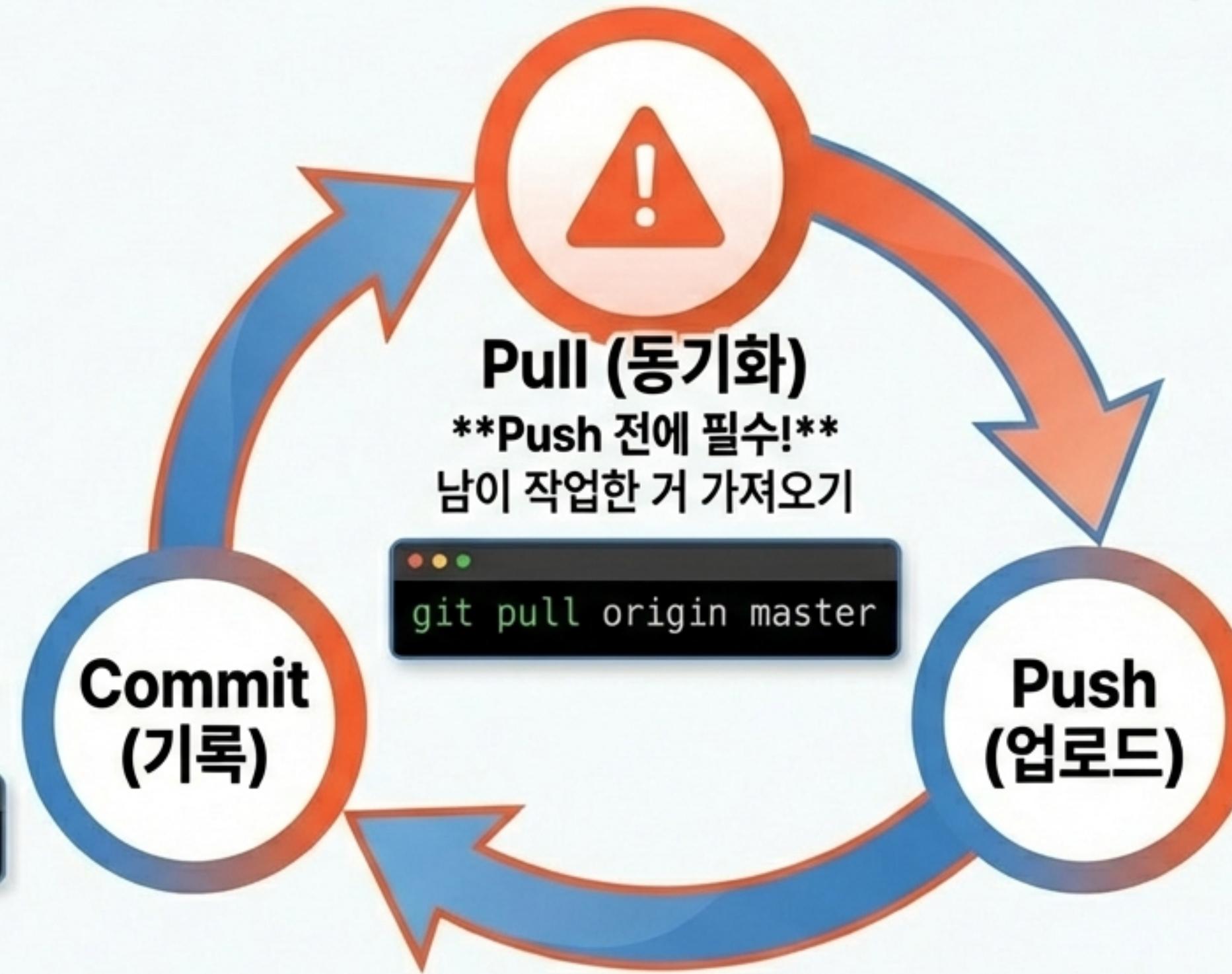


생성된 URL을 복사하여
내 컴퓨터로 가져올 준비.



```
$ git clone [URL]
```

충돌을 막는 골든 사이클: Commit, Pull, Push



당황하지 마세요: 충돌(Conflict) 해결 가이드

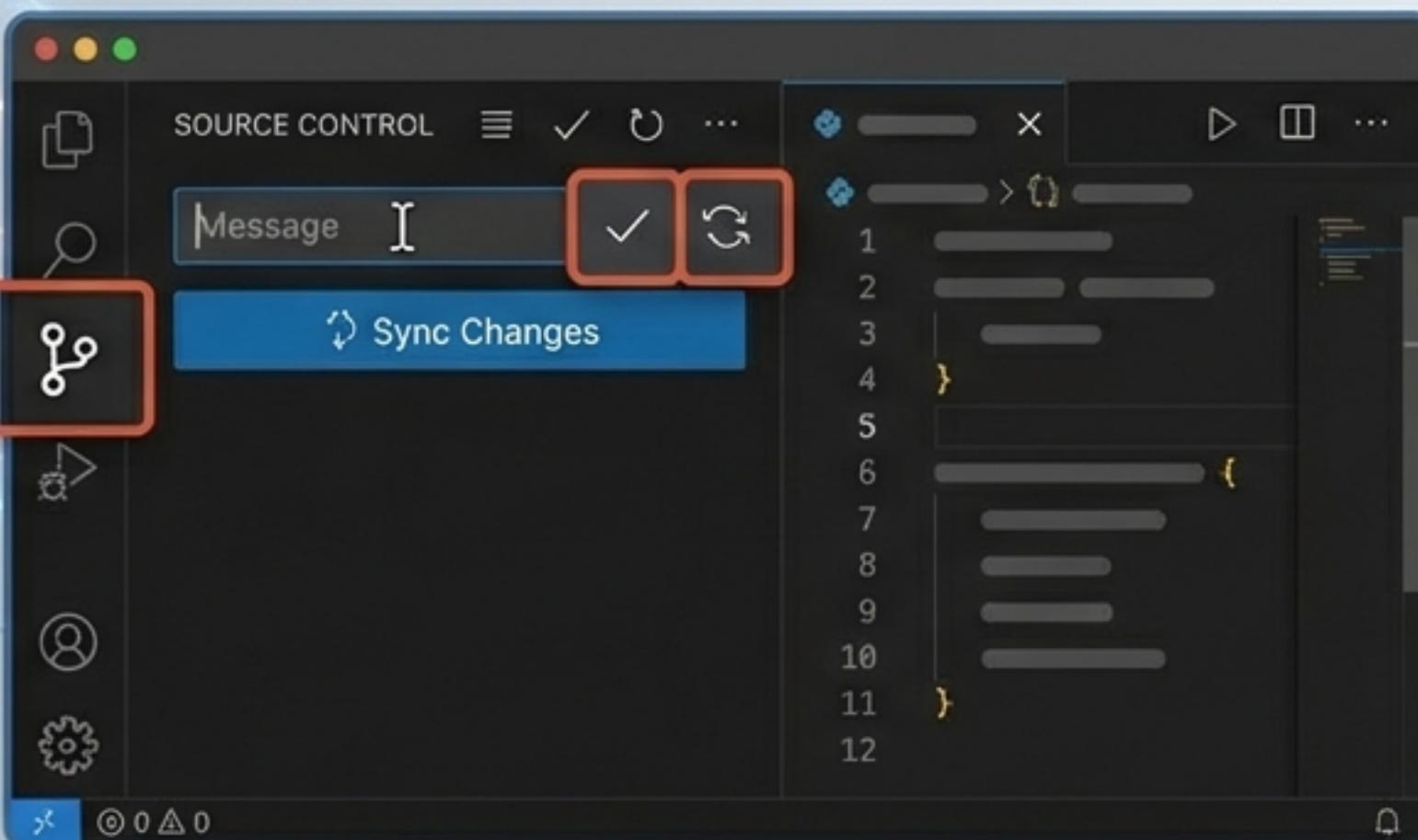
같은 파일의 같은 줄을 동시에 수정했을 때 발생합니다.

```
<<<<<< HEAD
print("나의 수정 코드") My Code
=====
print("동료의 수정 코드") Incoming Code
>>>>> branch_name
```

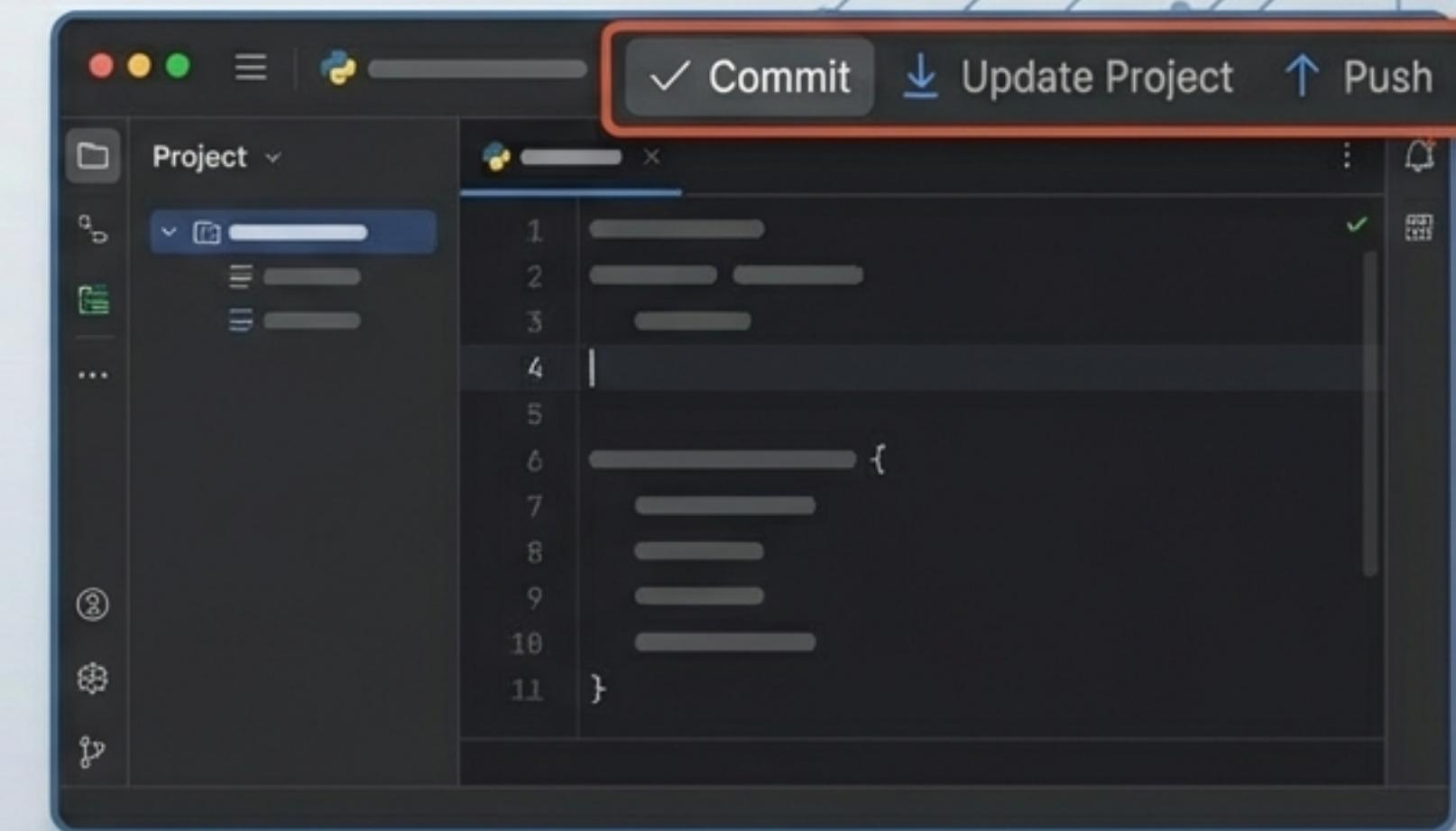
1. IDE에서 위와 같은 기호 확인.
2. 선택의 시간.. 둘 중 하나를 선택하거나, 코드를 수정하여 병합.
3. 특수 기호('<<<', '====', '>>>') 삭제 후 저장.
4. 다시 Commit -> Push 진행.

더 쉽게: IDE에서 Git 활용하기 (GUI)

터미널 명령어가 익숙하지 않다면 IDE의 내장 기능을 활용하세요.



VS Code Source Control



PyCharm Git Menu

Tip: 처음에는 흐름 파악을 위해 명령어를 익히고, 이후 편리한 GUI 사용을 권장합니다.

한 장으로 보는 연구자 셋업 체크리스트

1. Anaconda (기반)

- 설치 시 'Add to PATH' 체크했는가?
- 'conda create'로 프로젝트 전용 가상환경을 만들었는가?

2. IDE (도구)

- PyCharm/VS Code 인터프리터를 Conda 가상환경으로 연결했는가?
- 실행(Run) 시 'Hello World'가 정상 출력되는가?

3. Git (기록)

- GitHub 저장소를 생성하고 Clone 했는가?
- 'Commit' -> 'Pull' -> 'Push' 순서를 지키고 있는가?

자주 묻는 질문과 문제 해결 (Troubleshooting)



Q: `conda` 또는 `git` 명령어를 찾을 수 없대요.



A: 환경변수(PATH) 설정 누락입니다.
`Anaconda Prompt`를 사용하거나 환경변수를 수동 추가하세요.



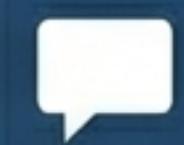
Q: 'No module named 'pandas'' 에러가 떠요.



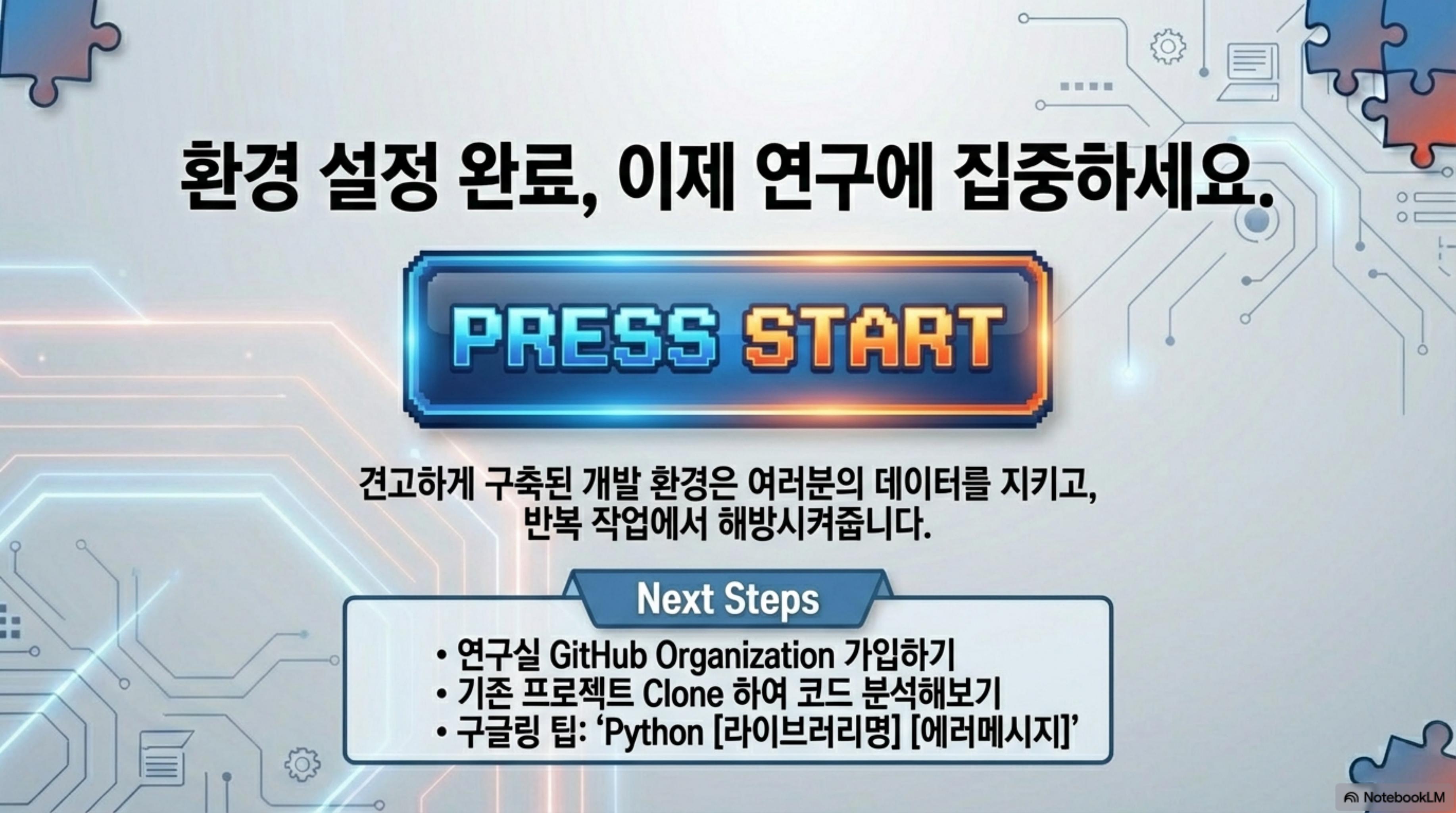
A: IDE의 인터프리터가 올바른 가상환경인지 확인하세요. 해당 환경에 `pip install` 했는지 확인하세요.



Q: 한글 주석이나 출력이 깨져서 나와요.



A: 소스 코드 상단에 '# -*- coding: utf-8 -*' 추가 혹은 IDE 인코딩 설정을 'UTF-8'로 변경하세요.



환경 설정 완료, 이제 연구에 집중하세요.

PRESS START

견고하게 구축된 개발 환경은 여러분의 데이터를 지키고,
반복 작업에서 해방시켜줍니다.

Next Steps

- 연구실 GitHub Organization 가입하기
- 기존 프로젝트 Clone 하여 코드 분석해보기
- 구글링 팁: ‘Python [라이브러리명] [에러메시지]’