

Python 리터러시



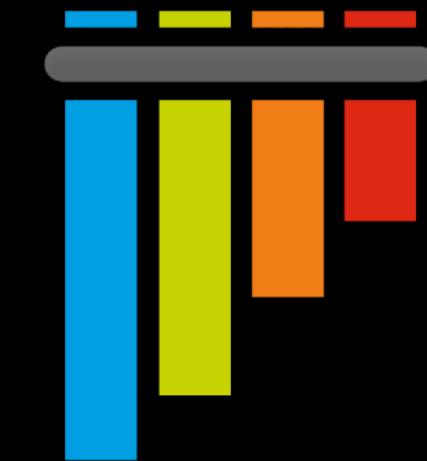
QA 엔지니어에게 Python이 왜 필요할까?

자동화 도구와 라이브러리

Selenium, PyTest, Robot Framework 등 다양한 자동화 도구와 라이브러리 지원



Selenium



PyTest

...

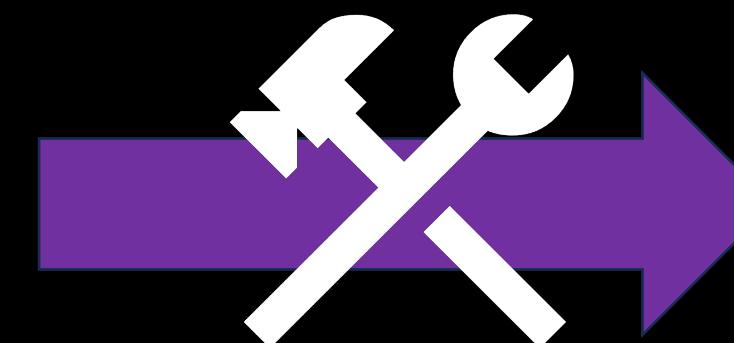


QA 엔지니어에게 Python이 왜 필요할까?

웹 테스트 및 크롤링

Selenium과 같은 라이브러리를 사용하여 웹 애플리케이션의 테스트를 자동화 가능

BeautifulSoup, Scrapy와 같은 도구로 웹 크롤링 및 데이터 수집 가능



Selenium





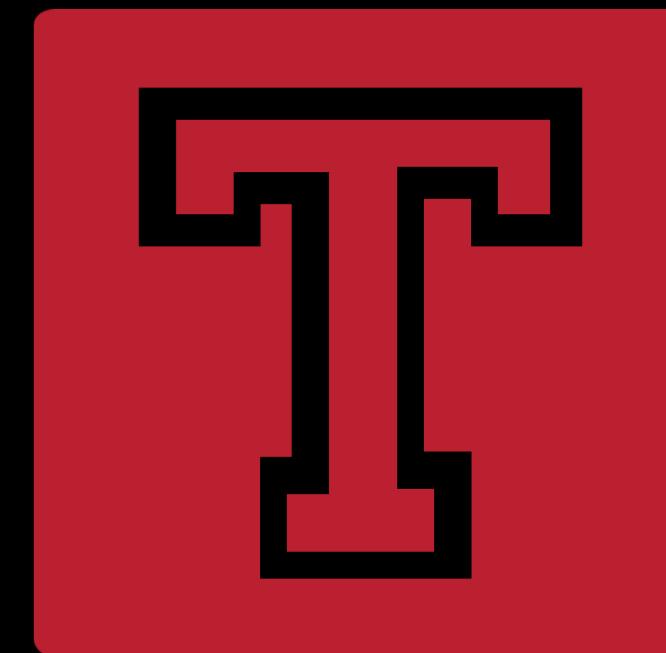
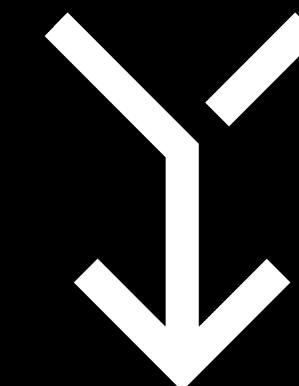
QA 엔지니어에게 Python이 왜 필요할까?

통합 테스트와 CI/CD 지원

Jenkins, Travis CI 등 다양한 Continuous Integration (CI) 도구와의 통합이 쉬움



Jenkins



Travis CI



QA 엔지니어에게 Python이 왜 필요할까?

데이터 처리 및 분석

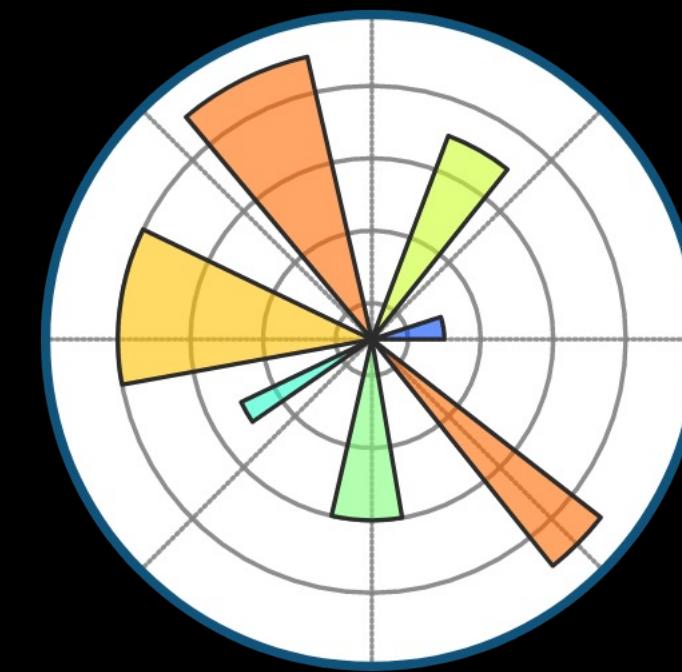
Pandas, NumPy, Matplotlib 등 강력한 데이터 처리 및 분석 도구 제공, 결과 분석에 유리



Pandas



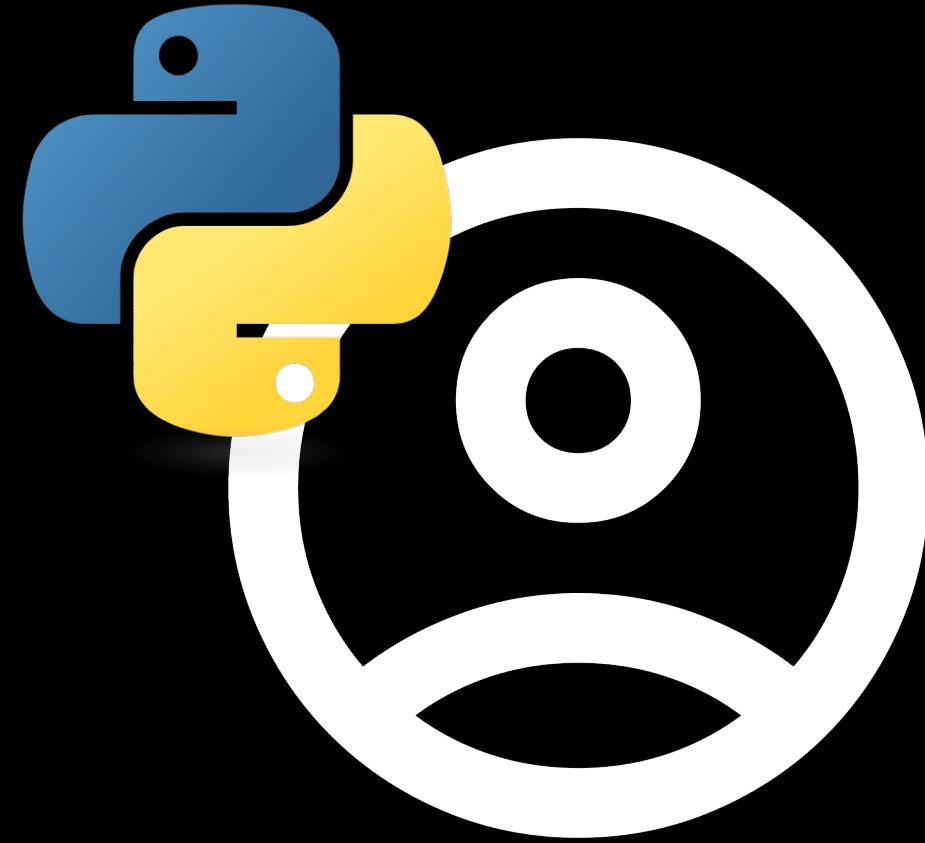
NumPy



Matplotlib



Python 사용 가능 여부에 따른 QA 엔지니어의 차이점

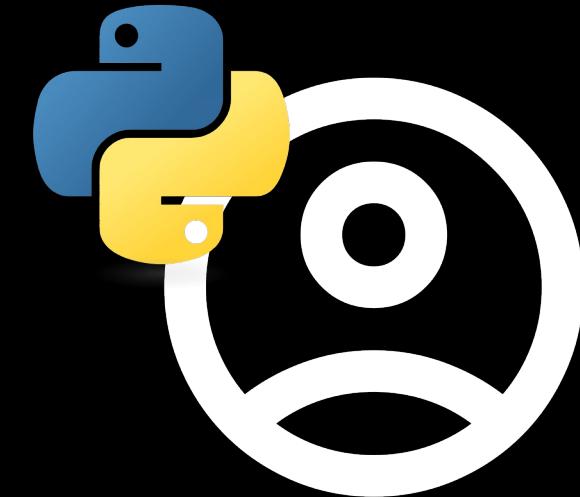


VS



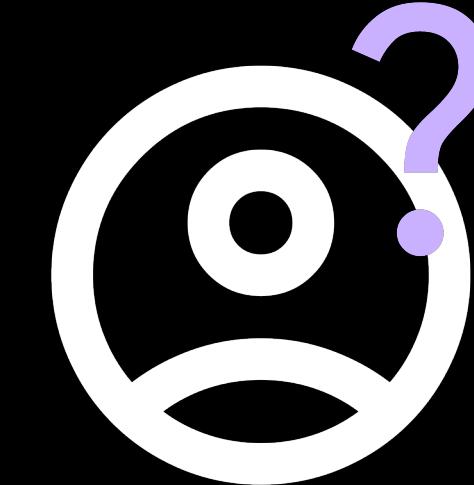


Python 사용 가능 여부에 따른 QA 엔지니어의 차이점



테스트 자동화 스크립트
작성 가능

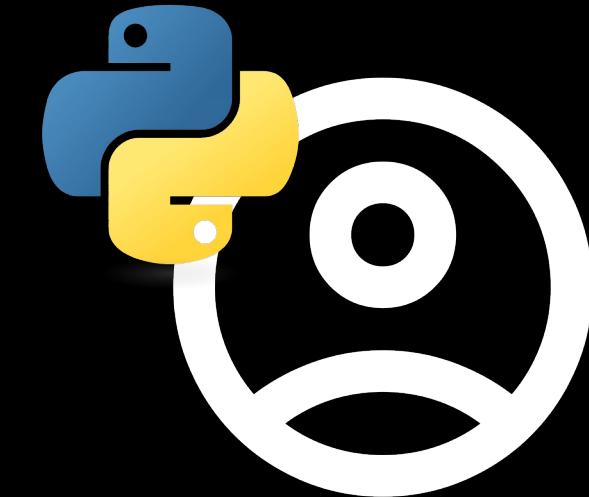
자동화 능력



수동 테스트가 주로 필요

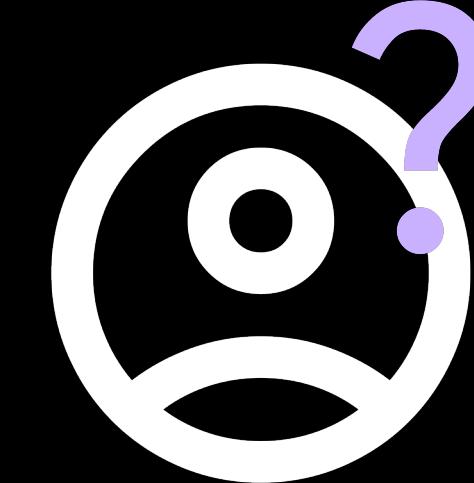


Python 사용 가능 여부에 따른 QA 엔지니어의 차이점



코드로 문제 해결 및
디버깅 가능

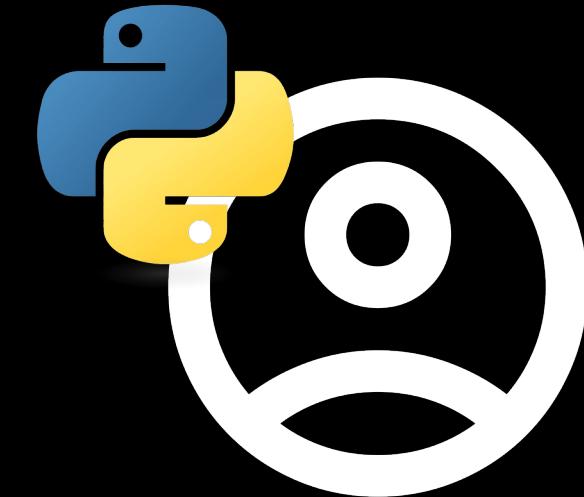
문제 해결 능력



수동으로 문제 추적 및 해결



Python 사용 가능 여부에 따른 QA 엔지니어의 차이점



효율성

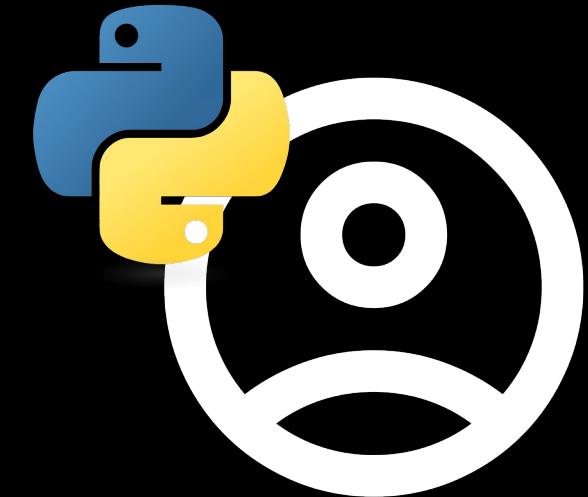


반복 작업 자동화,
빠르고 효율적인 테스트 실행

반복 작업 수동 처리,
시간이 오래 걸림

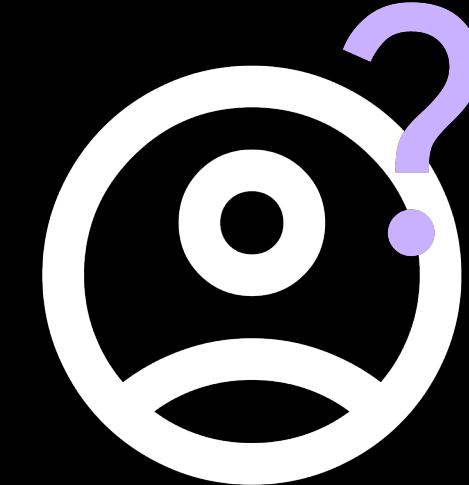


Python 사용 가능 여부에 따른 QA 엔지니어의 차이점



데이터 분석

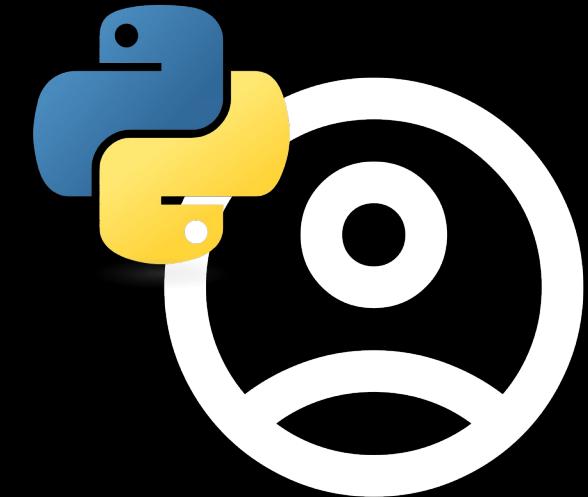
Pandas, Matplotlib으로
테스트 결과 분석 가능



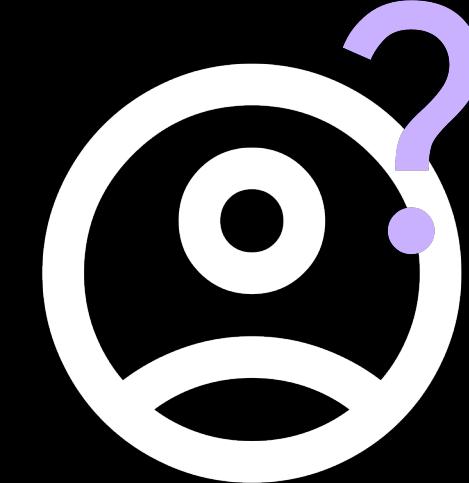
수동으로 데이터 분석 및
리포트 작성



Python 사용 가능 여부에 따른 QA 엔지니어의 차이점



CI/CD 통합

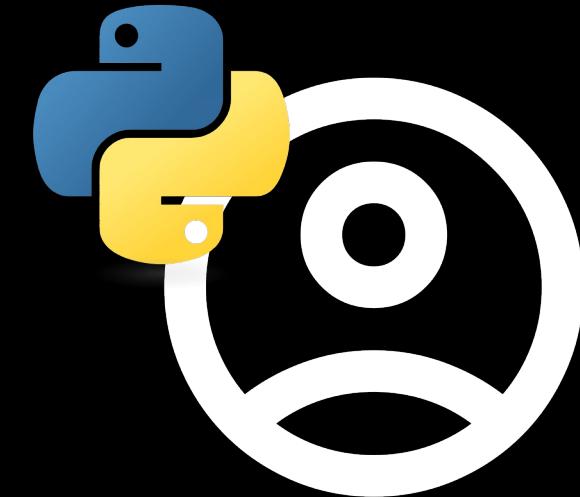


CI/CD 파이프라인에
자동화 테스트 통합 가능

CI/CD 파이프 라인
통합에 어려움

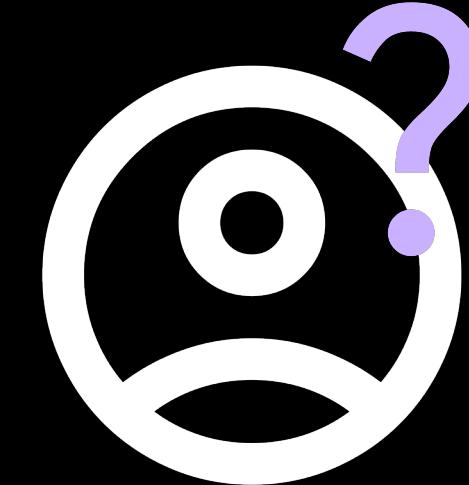


Python 사용 가능 여부에 따른 QA 엔지니어의 차이점



코드 확장 및 유지보수 용이

유지보수



테스트 스크립트
수정 및 확장 어려움



Python 사용 가능 여부에 따른 QA 엔지니어 어의 차이점 정리

	Python을 활용할 수 있는 QA 엔지니어	Python을 모르는 QA 엔지니어
자동화 능력	테스트 자동화 스크립트 작성 가능	수동 테스트가 주로 필요
문제 해결 능력	코드로 문제 해결 및 디버깅 가능	수동으로 문제 추적 및 해결
효율성	반복 작업 자동화, 빠르고 효율적인 테스트 실행	반복 작업 수동 처리, 시간이 오래 걸림
데이터 분석	Pandas, Matplotlib으로 테스트 결과 분석 가능	수동으로 데이터 분석 및 리포트 작성
CI/CD 통합	CI/CD 파이프라인에 자동화 테스트 통합 가능	CI/CD 파이프라인 통합에 어려움
유지보수	코드 확장 및 유지보수 용이	테스트 스크립트 수정 및 확장 어려움



우리의 목표!

파이썬 프로그래밍 언어를 잘 다루는 것



우리의 목표!

파이썬 프로그래머를 잘 다루는 것

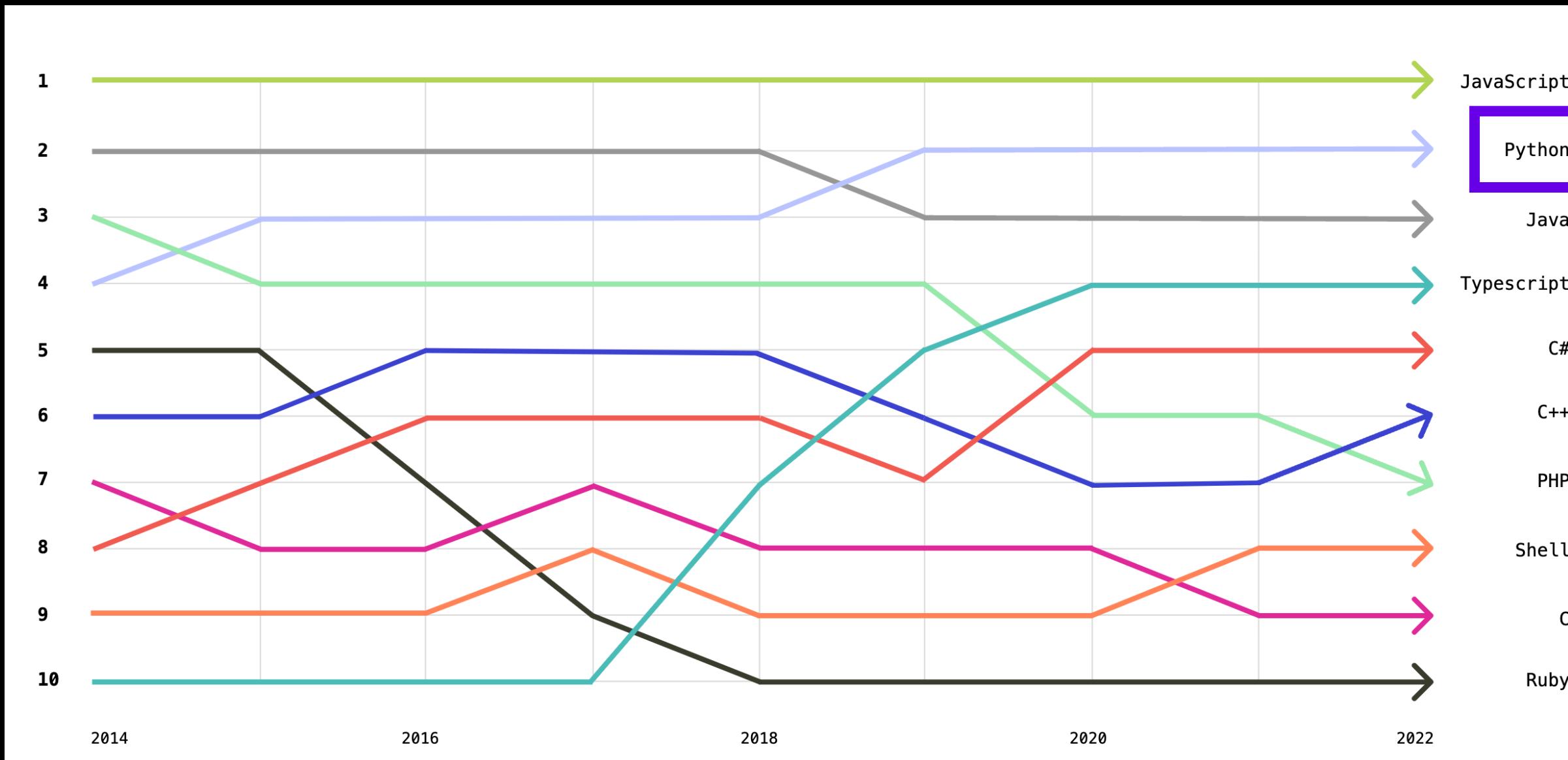
테스트 자동화를 위해 필요한 **최소한의 지식** 습득!



파이썬의 세계로 Dive in!



프로그래밍 언어 (Programming Languages)



프로그래밍

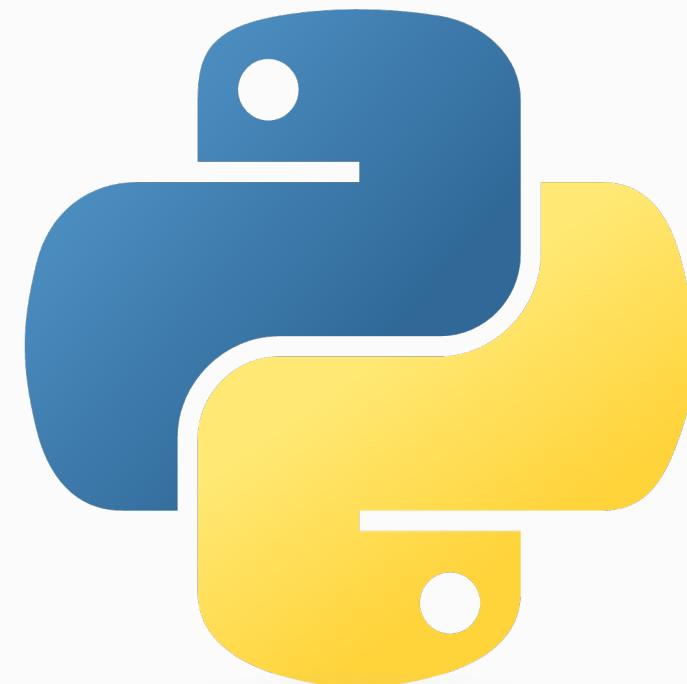
- 프로그램을 만드는 작업
- 컴퓨터에게 일을 시키는 방법

프로그래밍 언어

- 프로그램을 만들기 위해 컴퓨터와 통신하는 언어



Python (파이썬)



Guido Van Rossum에 의해 개발됨

ABC 프로그래밍 언어의 후속으로 개발

1991년 첫 번째 릴리즈

[So who is Guido van Rossum? \(gvanrossum.github.io\)](http://gvanrossum.github.io)



아름다운 것이 추한 것보다 낫습니다.

명시적인 것이 암시적인 것보다 낫습니다.

단순한 것이 복잡한 것보다 낫습니다.

복잡한 것이 난해한 것보다 낫습니다.

평면적인 것이 중첩된 것보다 낫습니다.

희소한 것이 밀집한 것보다 낫습니다.

가독성은 중요합니다.

특별한 경우라도 규칙을 깨기에 충분하지 않습니다.

그러나 실용성이 순수함을 이깁니다.

오류는 절대 침묵해서는 안 됩니다.

명시적으로 침묵시킨 경우를 제외하고요.

모호함을 마주했을 때, 추측의 유혹을 거부하세요.

그것을 하는 방법이 하나 -- 되도록이면 오직 하나 -- 있어야 합니다.

비록 그 방법이 처음에는 명확하지 않을 수 있습니다. 당신이 네덜란드인이 아니라면요.

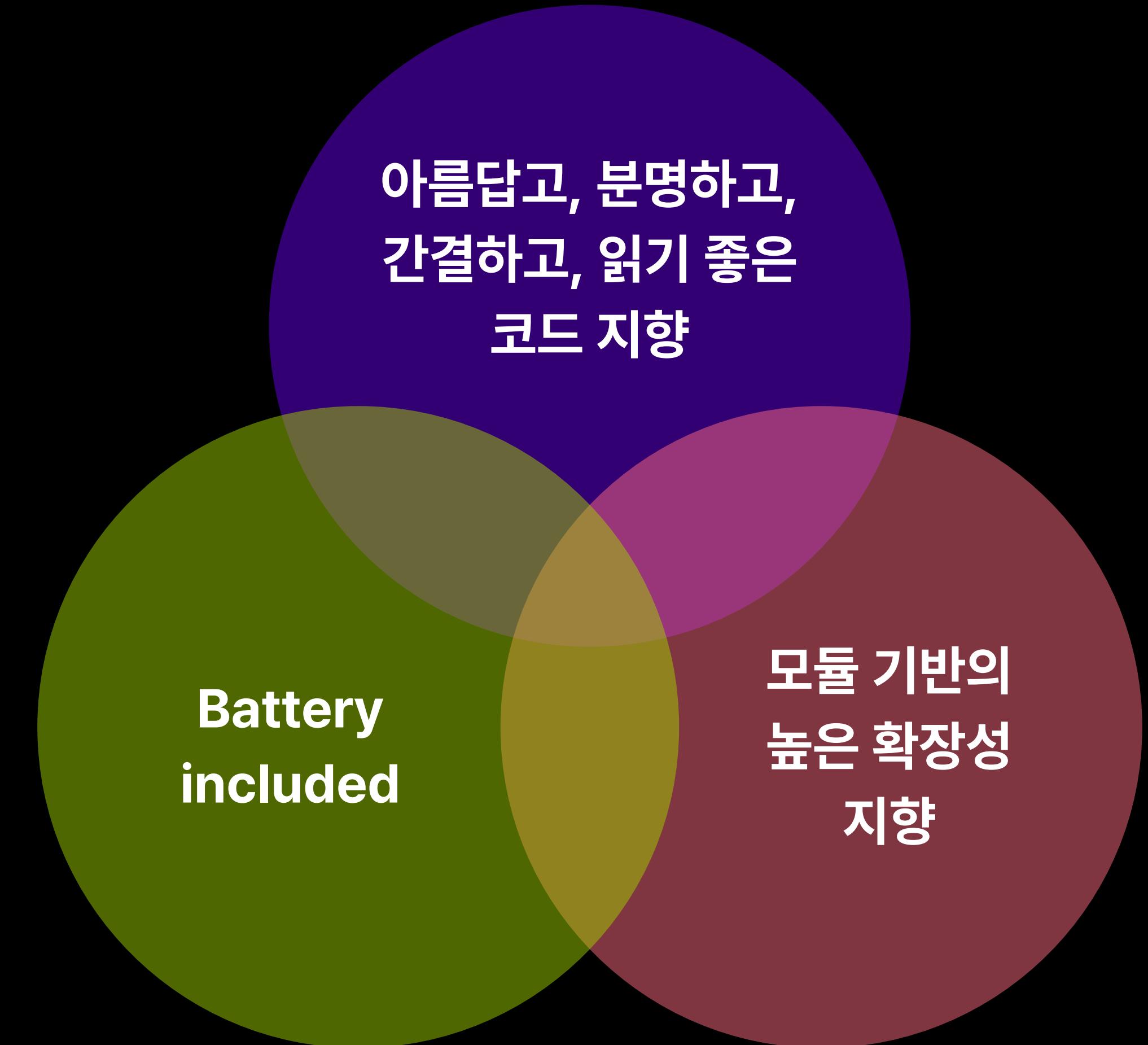
지금 하는 것이 결코 하지 않는 것보다 낫습니다.

그렇지만 때로는 바로 지금하는 것보다는 결코 하지 않는 것이 낫습니다.

구현이 설명하기 어렵다면, 그것은 나쁜 아이디어입니다.

구현이 설명하기 쉽다면, 그것은 좋은 아이디어일 수 있습니다.

네임스페이스는 정말 좋은 아이디어입니다 -- 그것을 더 많이 사용합시다!





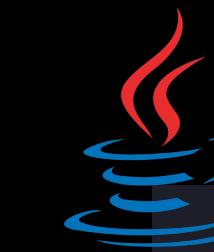
높은 가독성



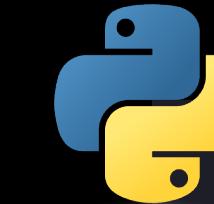
```
#include <stdio.h>

int main()
{
    printf("Hi, Serendippers!");

    return 0;
}
```



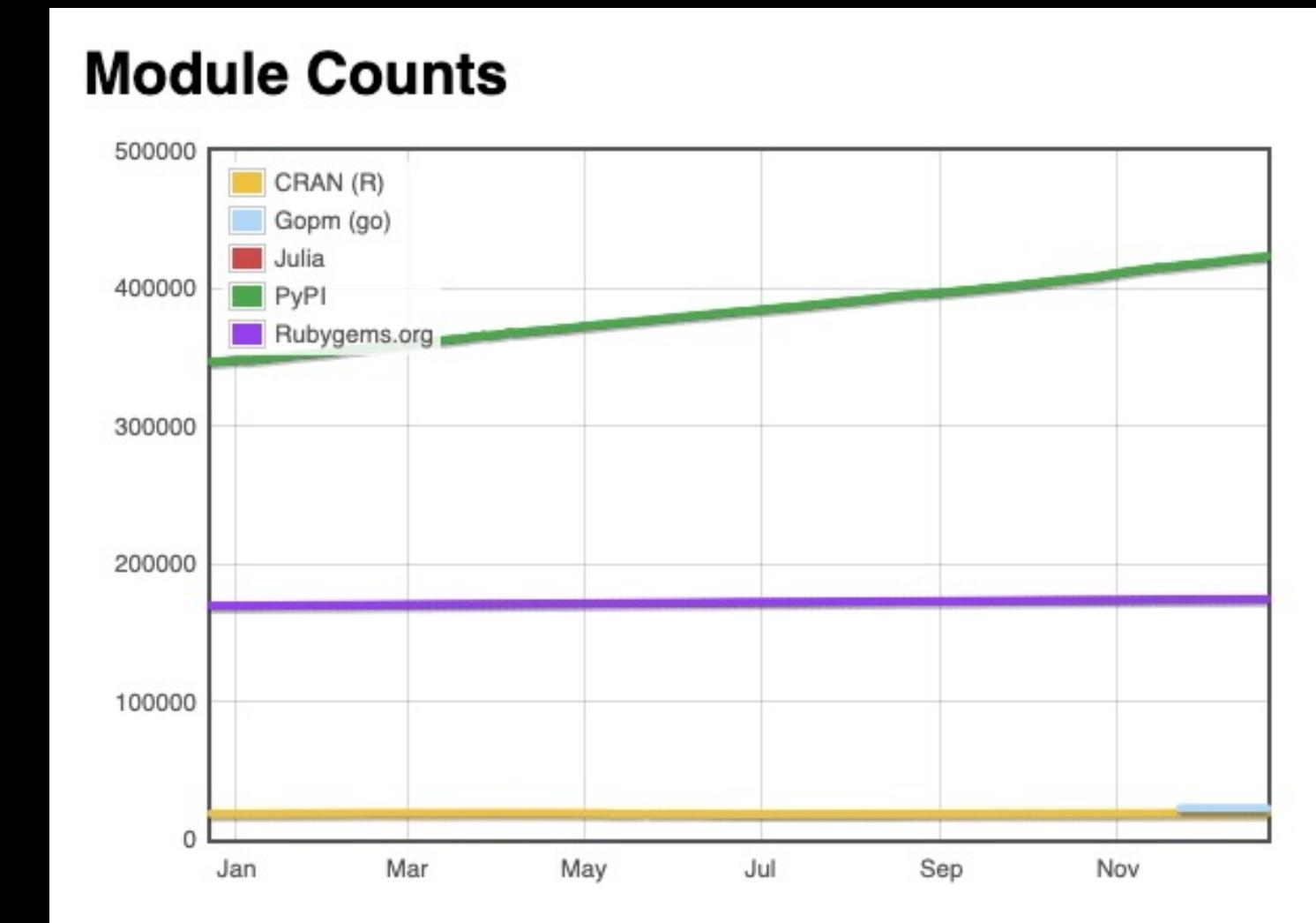
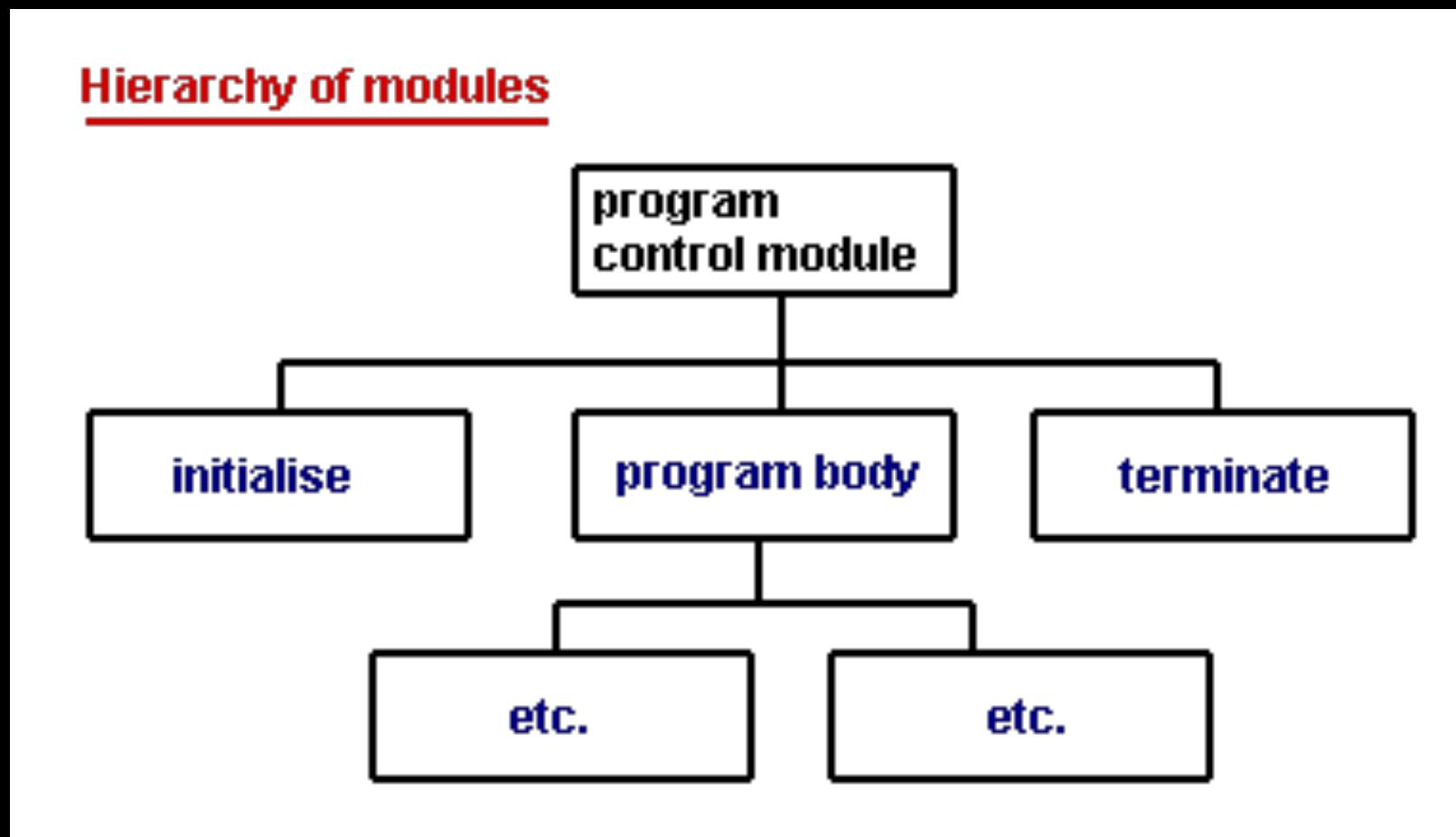
```
public class Main {
    public static void main(String[] args) {
        System.out.println("Hi, Serendippers!");
    }
}
```



```
print("Hi, Serendippers!")
```



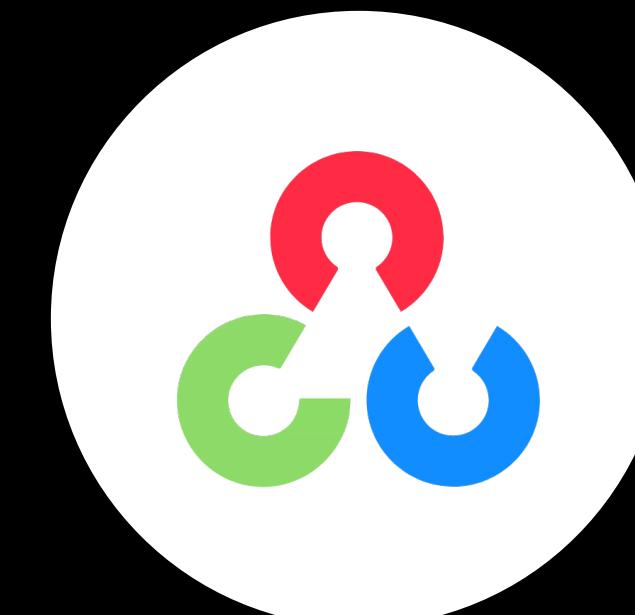
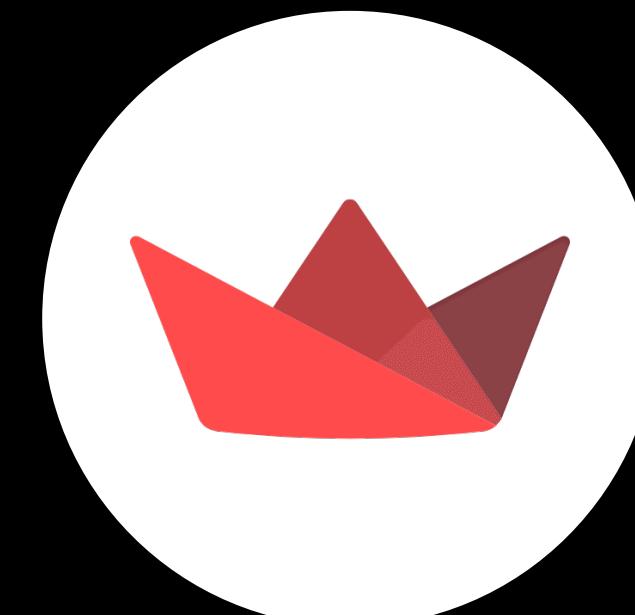
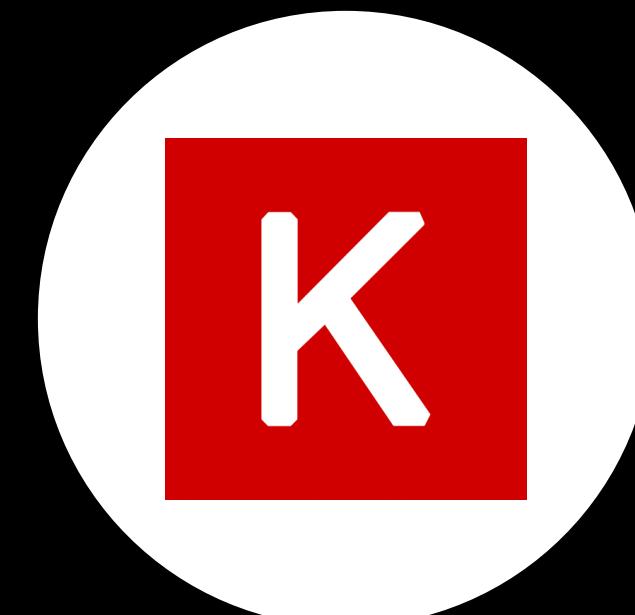
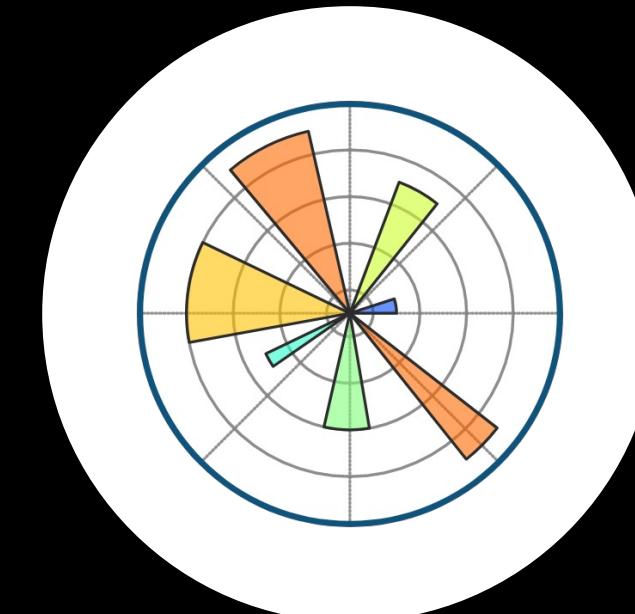
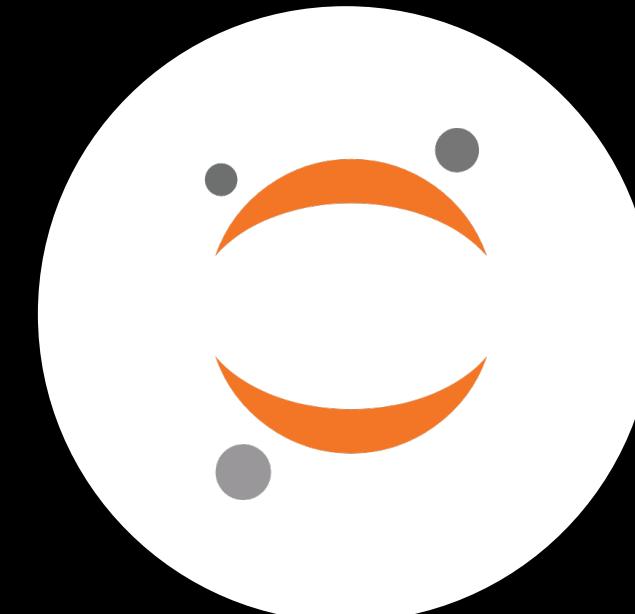
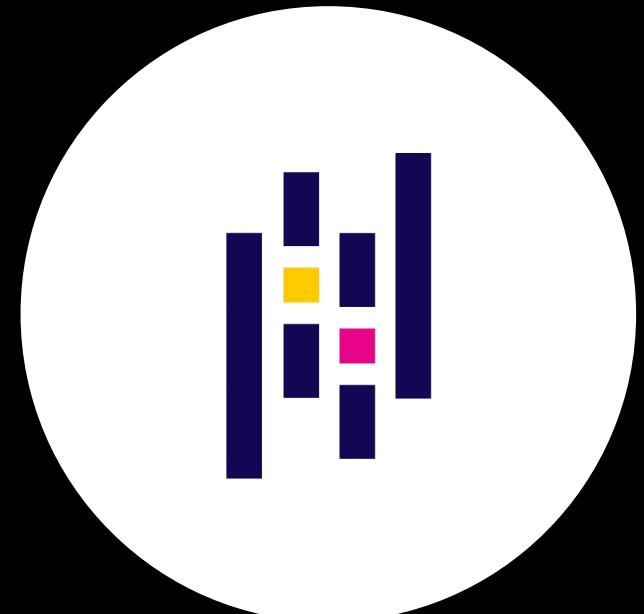
모듈



- **프로그래밍 모듈**
 - 독립적으로 실행가능하며 교체가능한 프로그램
 - 일반적으로 특정한 역할을 수행
- **모듈 인터페이스**
 - 모듈과 메인 프로그램을 연결하는 규약



The Python's Scientific Ecosystem





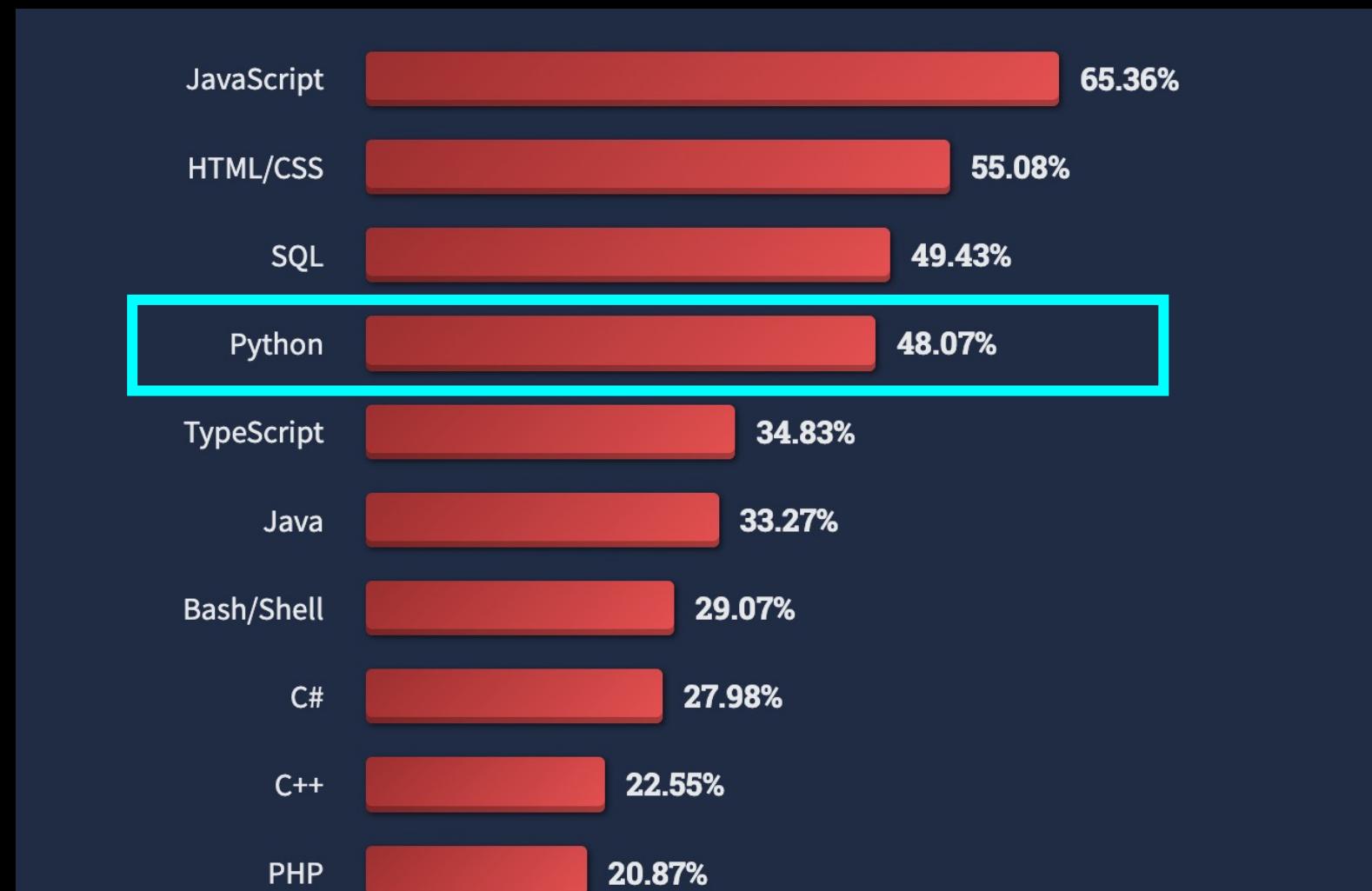
2023 Developer Survey



지난 1년(2023년) 동안 어떤 프로그래밍, 스크립트, 마크업 언어로 광범위한 개발 작업을 수행했으며,
내년에는 어떤 언어로 작업하고 싶나요?

Which programming, scripting, and markup languages have you done extensive development work in over the past year, and which do you want to work in over the next year?

2022





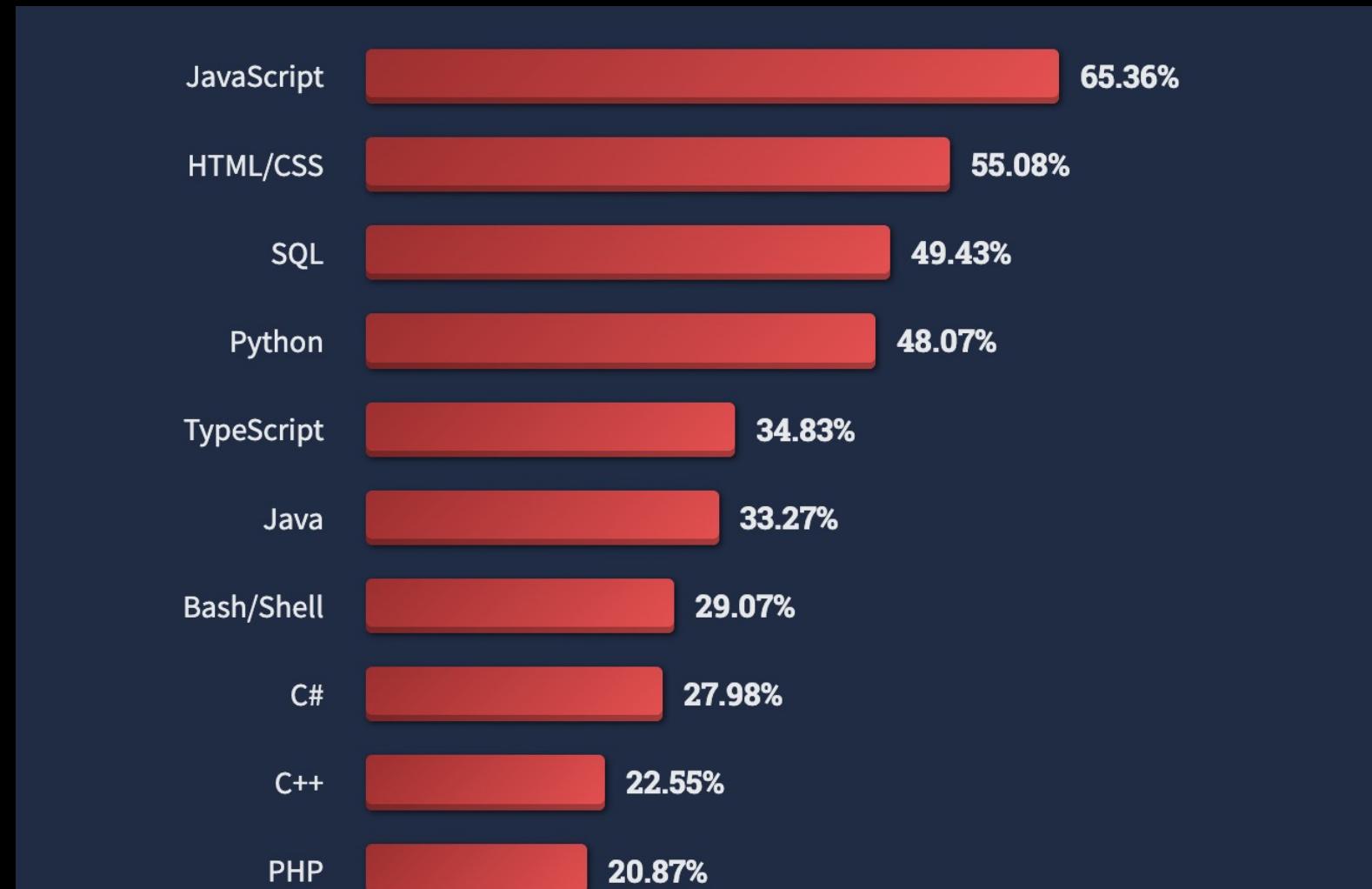
2023 Developer Survey



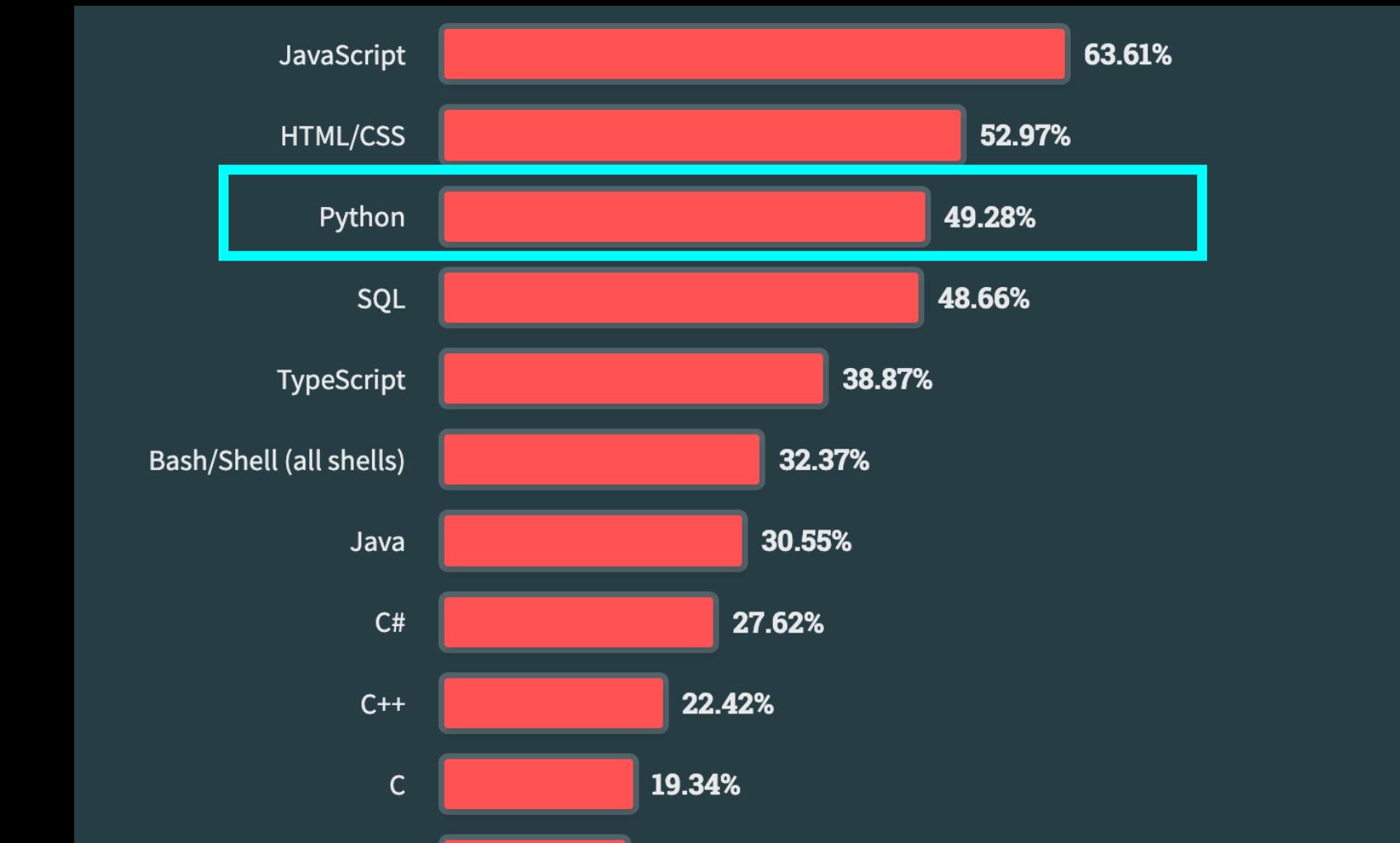
지난 1년(2023년) 동안 어떤 프로그래밍, 스크립트, 마크업 언어로 광범위한 개발 작업을 수행했으며,
내년에는 어떤 언어로 작업하고 싶나요?

Which programming, scripting, and markup languages have you done extensive development work in over the past year, and which do you want to work in over the next year?

2022



2023



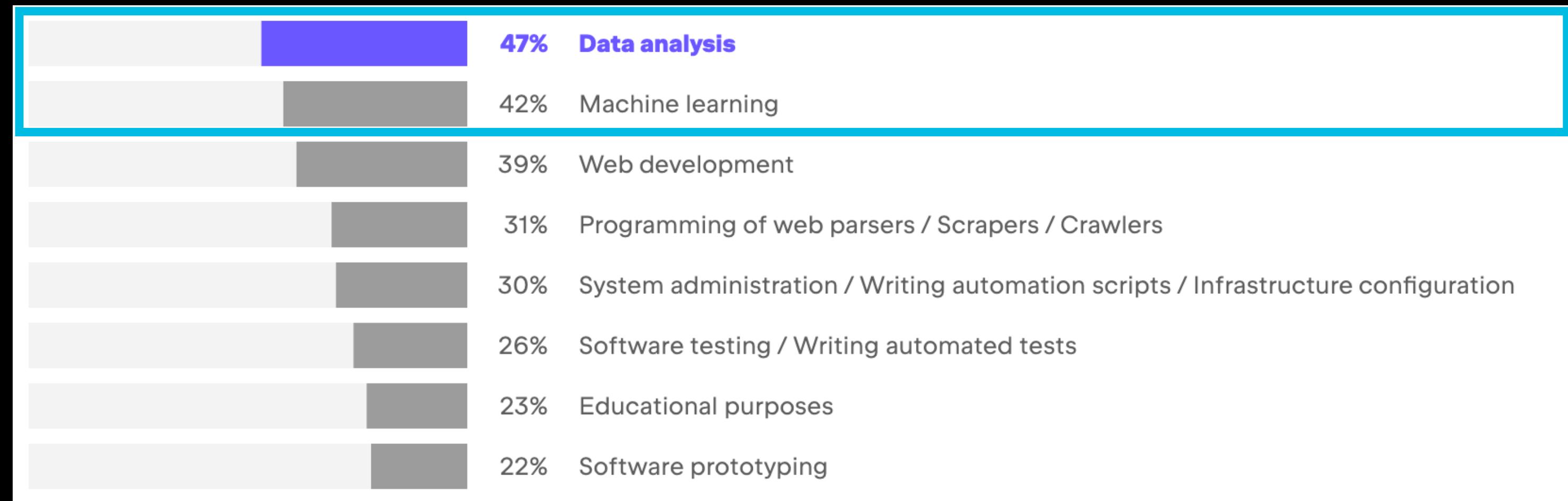


The State of Developer Ecosystem



Python을 어떤 용도로 사용하시나요?

What do you use Python for?



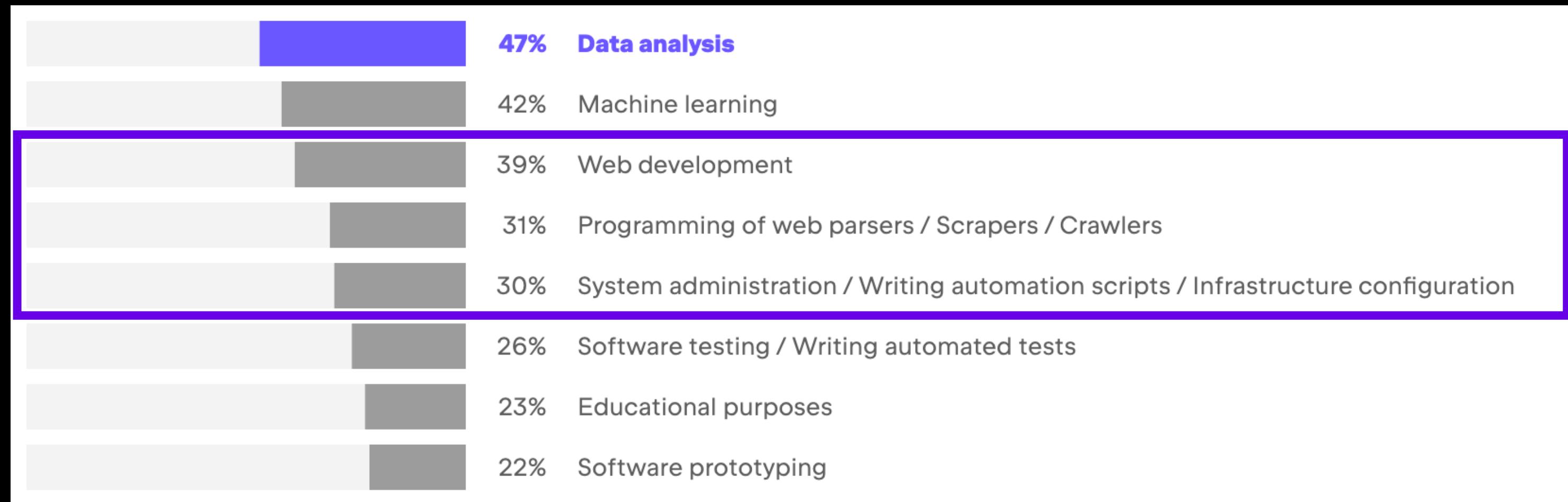


The State of Developer Ecosystem



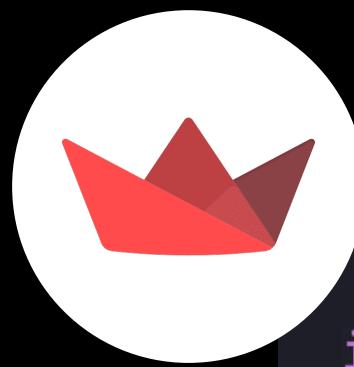
Python을 어떤 용도로 사용하시나요?

What do you use Python for?

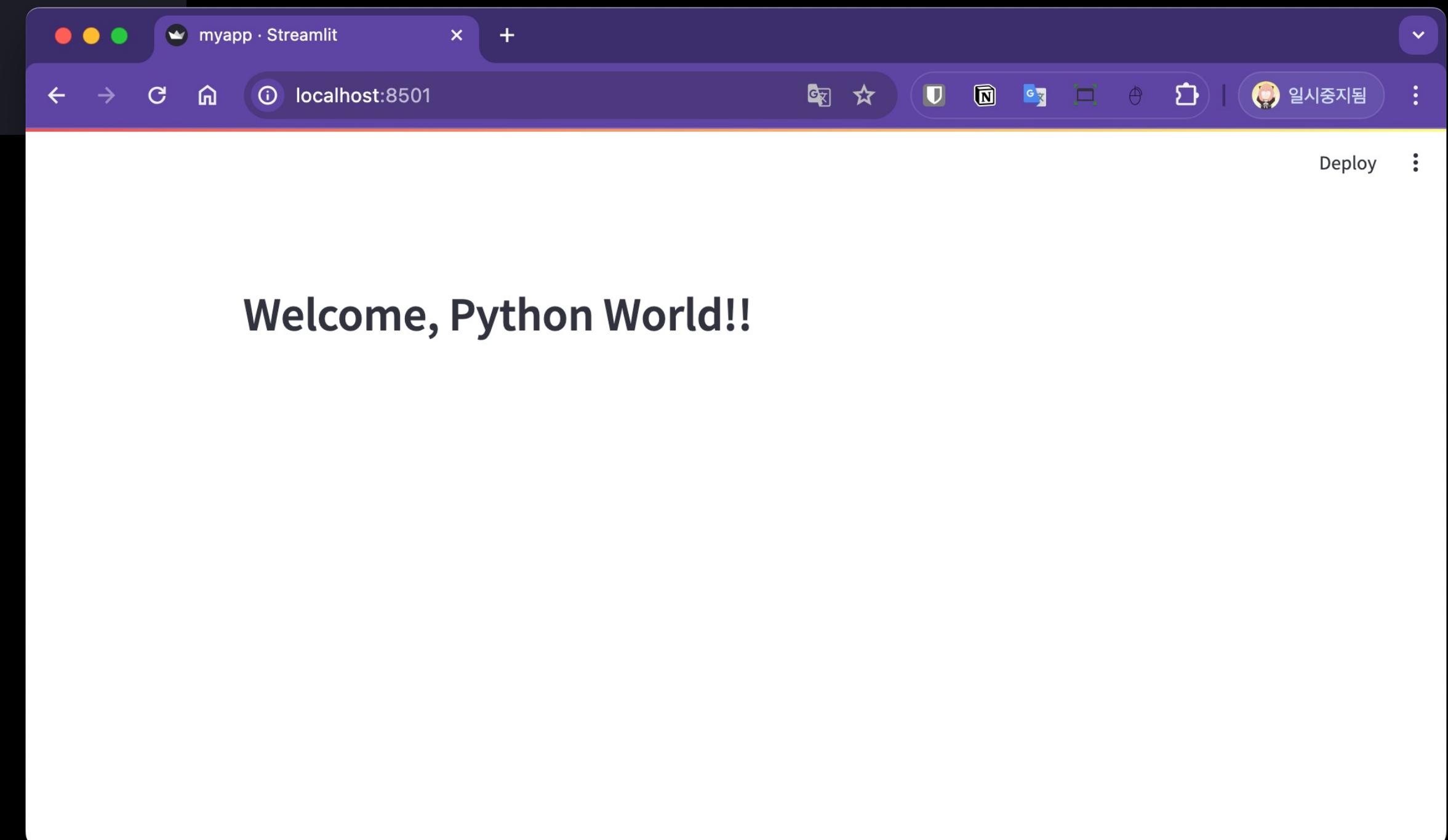
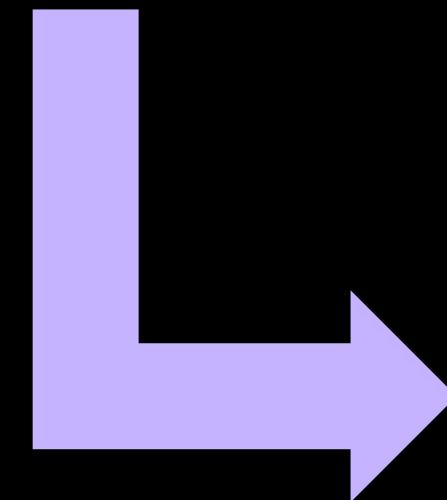




단 두 줄로 웹 애플리케이션 만들기



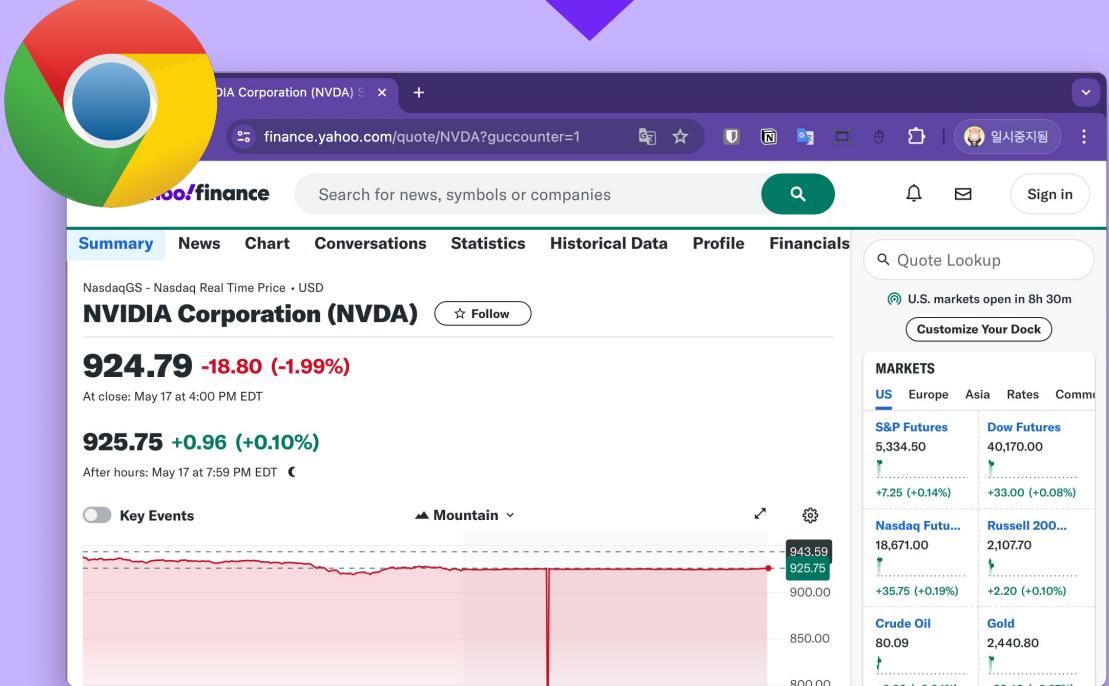
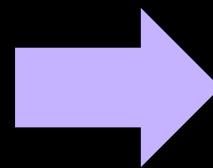
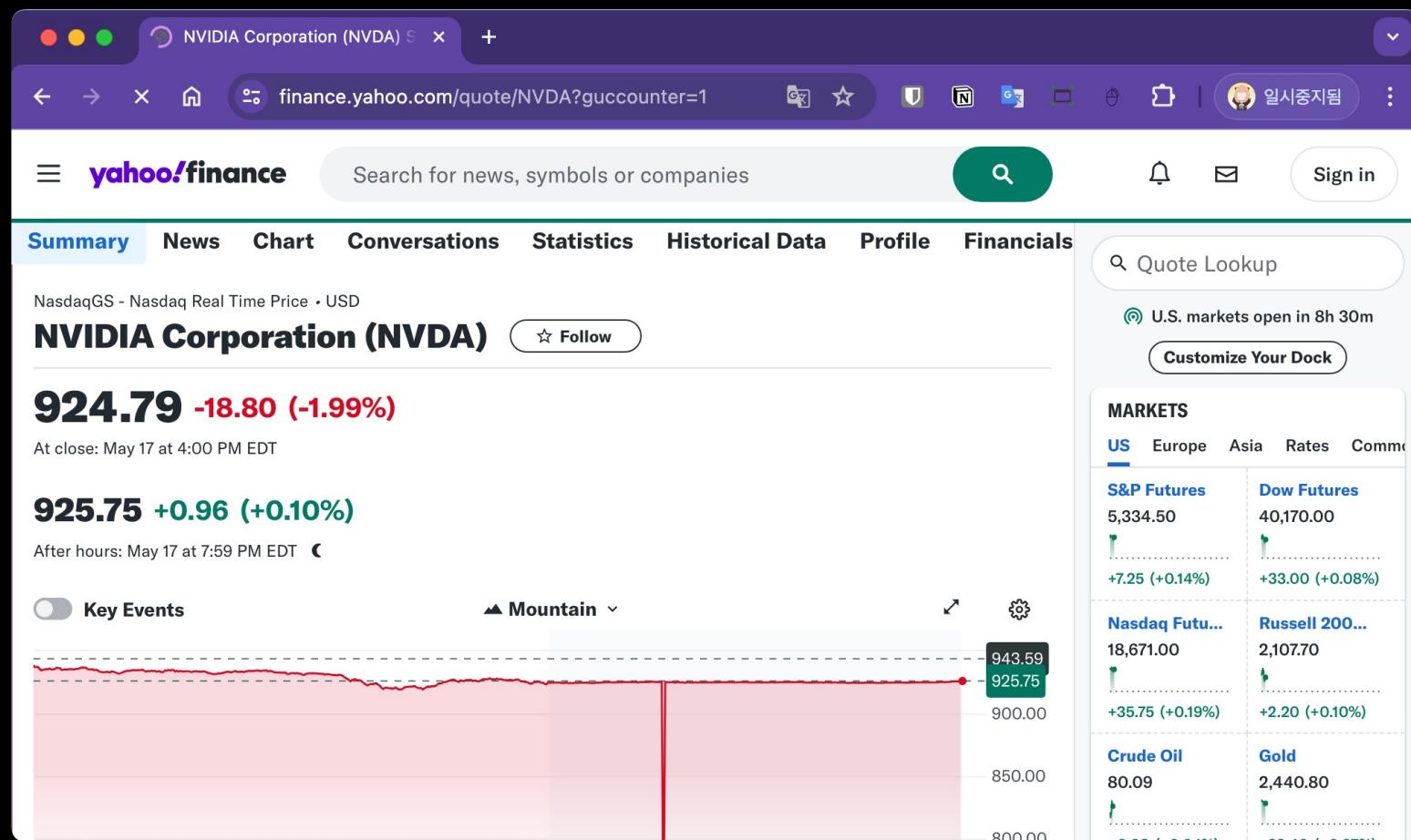
```
import streamlit as st  
  
st.header("Welcome, Python World!!")
```



웹 어플리케이션 : 인터넷을 통해 웹 브라우저에서 이용할 수 있는 응용 프로그램



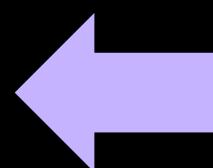
웹 스크래핑 과정 모식도



```
<div class="container svelte-mgkamr">
  <fin-streamer class="livePrice svelte-mgkamr" data-symbol="NVDA" data-testid="qsp-price" data-field="regularMarketPrice" data-trend="none" data-pricehint="2" data-value="924.79" active="">
    <span>924.79</span>
  </fin-streamer>

  <!-- Skip -->

</div>
```





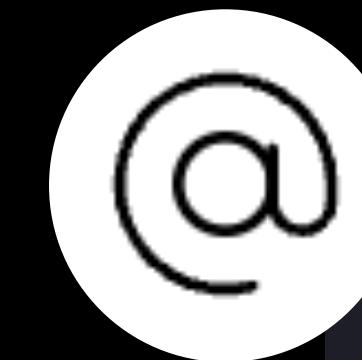
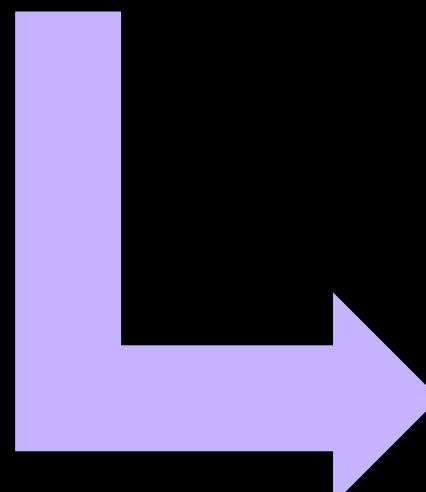
Python을 이용한 업무자동화



```
import qrcode

url_link = "https://www.python.org"

qr_img = qrcode.make(url_link)
qr_img.save("qr_img.png")
```



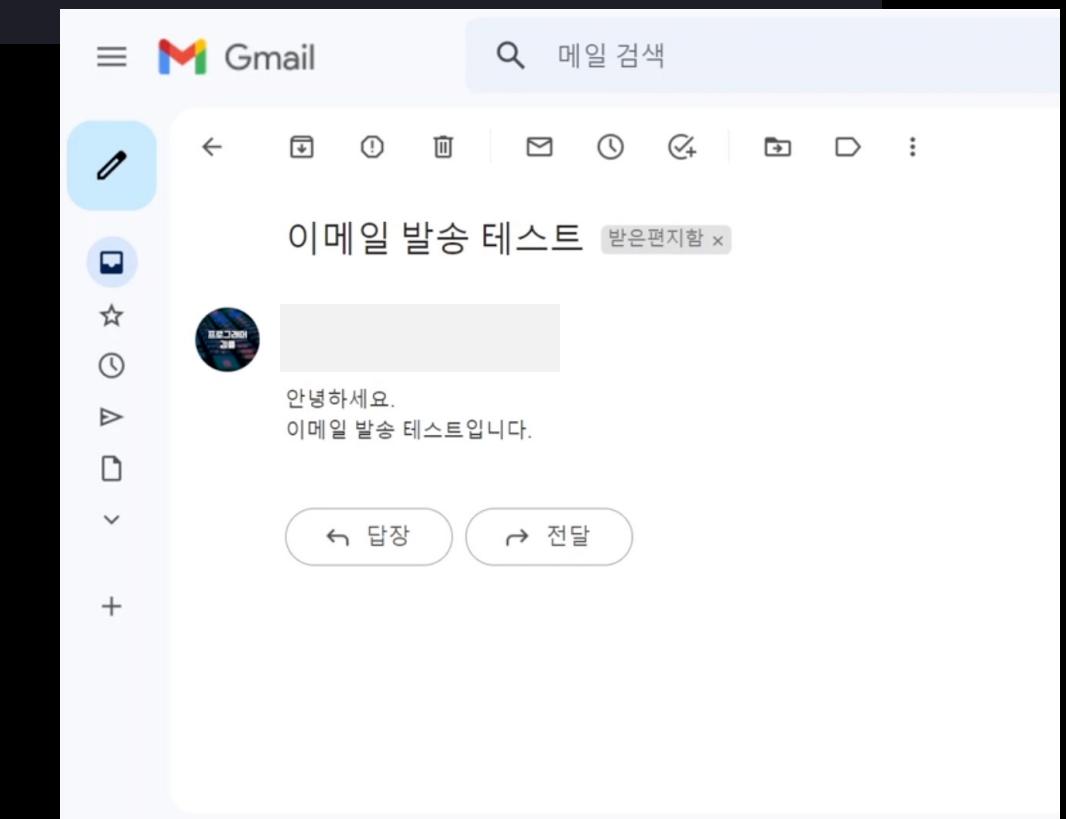
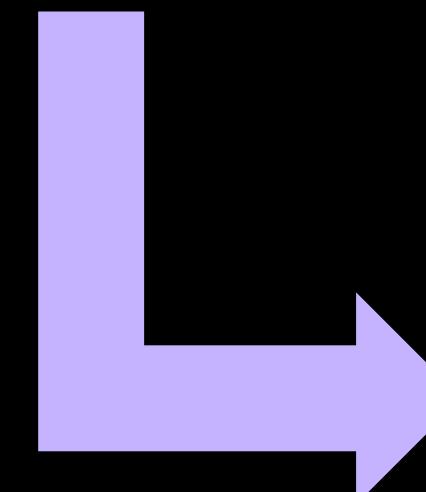
```
import smtplib
from email.mime.text import MIMEText

sender_email = "k*****@elicer.com"
sender_password = "*****"

receiver_email = "test@elicer.com"

email_title = "이메일 발송 테스트"
email_message = "안녕하세요. \n이메일 발송 테스트입니다."

msg = MIMEText(email_message)
msg['Subject'] = email_title
msg['From'] = sender_email
```





다양한 Python 사례

Python의 인기가 높아지고 다양한 개발이 가능해지면서
다양한 분야 & 기업에서 사용 사례가 많아지고 있음

사례1. 자사 제품에 특화된 Python 개량판 배포

사례2. 자사 제품에 특화된 계산 성능을 가진 Python 기반의 프로그램(오픈소스)배포

사례3. 자사에서 보유한 데이터와 Python을 이용한 거대언어모델(LLM) 개발



intel : intel CPU에 최적화된 Python 배포

The screenshot shows the Intel Distribution for Python landing page. At the top, there's a navigation bar with links for PRODUCTS, SUPPORT, SOLUTIONS, DEVELOPERS, PARTNERS, and FOUNDRY. Below that is a search bar and language selection (ENGLISH). The main heading is "Intel® Distribution for Python*". A sub-headline says, "Achieve near-native code performance with this set of essential packages optimized for high-performance numerical and scientific computing." Below the heading are three tabs: "Overview" (selected), "Download", and "Documentation & Resources". The "High-Performance Python" section describes the distribution's features, including support for latest CPU instructions, near-native performance through acceleration of core numerical and machine learning packages, and productivity tools for compiling Python code. It also mentions essential Python bindings for integrating with Intel native tools. The "Develop for Accelerated Compute" section highlights Data Parallel Extensions for Python*. The "Download the Stand-Alone Version" section provides a download link. The "Develop in the Cloud" section explains how to build and optimize oneAPI multiarchitecture applications using Intel-optimized tools and test workloads across Intel CPUs and GPUs. A "Get Access" button is available for FPGA and People's Republic of China (PRC) Based Developers. A purple speech bubble icon is also present.

Intel에서 제작하였으며 Intel 하드웨어에서 더 높은 성능을 발휘할 수 있도록 최적화된 Python
Intel Math Kernel Library (Intel MKL)를 사용하여 수학 연산의 성능을 극대화하였고
다양한 연산(벡터 계산, 선형 대수 등)에 최적화

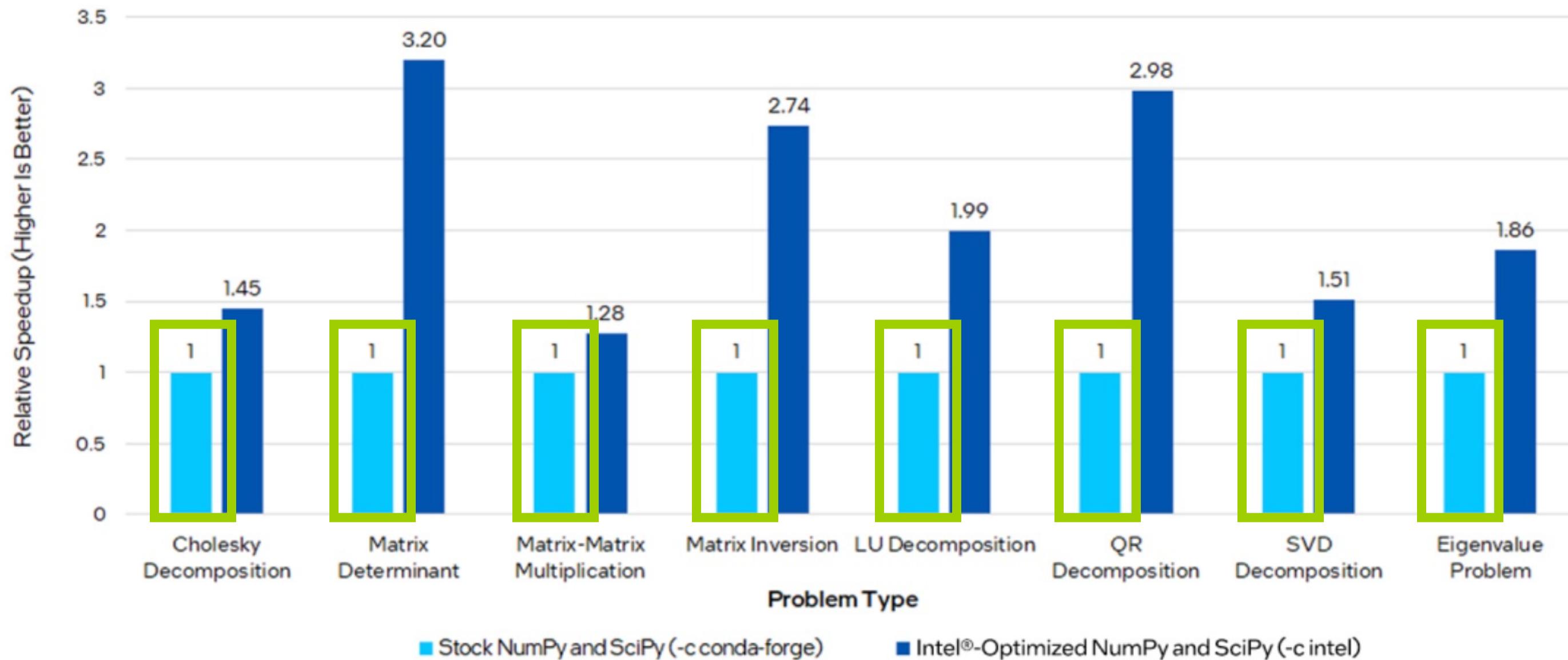
[Intel® Distribution for Python*](#)



intel : intel CPU에 최적화된 Python 배포

Intel®-Optimized NumPy* & SciPy* Linear Algebra Performance

Performance is increased up to 3.2x with Intel optimizations



Intel Optimization for NumPy and SciPy compared to conda-forge channel NumPy and SciPy performance for linear algebra on Intel® Xeon® Platform 8480+

Testing Date: Performance results are based on testing by Intel as of July 15, 2023, and may not reflect all publicly available updates.

Configuration Details and Workload Setup: System: cloud.intel.com, nodes=1:apr:ppn=2, Intel® Xeon® Platinum 8480+, 2 sockets, 56 cores per socket, HT On, Intel® Turbo Boost Technology on, Total memory 528 GB, RAM 33 MHz, Ubuntu® 20.04.5 LTS, 5.18.15-051815-generic, Microcode: 0x2b000310, benchmarks https://github.com/intelPython/ibench_Linear_Algebra, -c conda-forge environment versions: NumPy 1.23.5, SciPy 1.10.1, Numba™ 0.56.4 modules installed, -c intel environment versions: NumPy 1.21.4, SciPy 1.7.3, Numba™ 0.56.3, tbb4py 2021.8.0 modules installed.

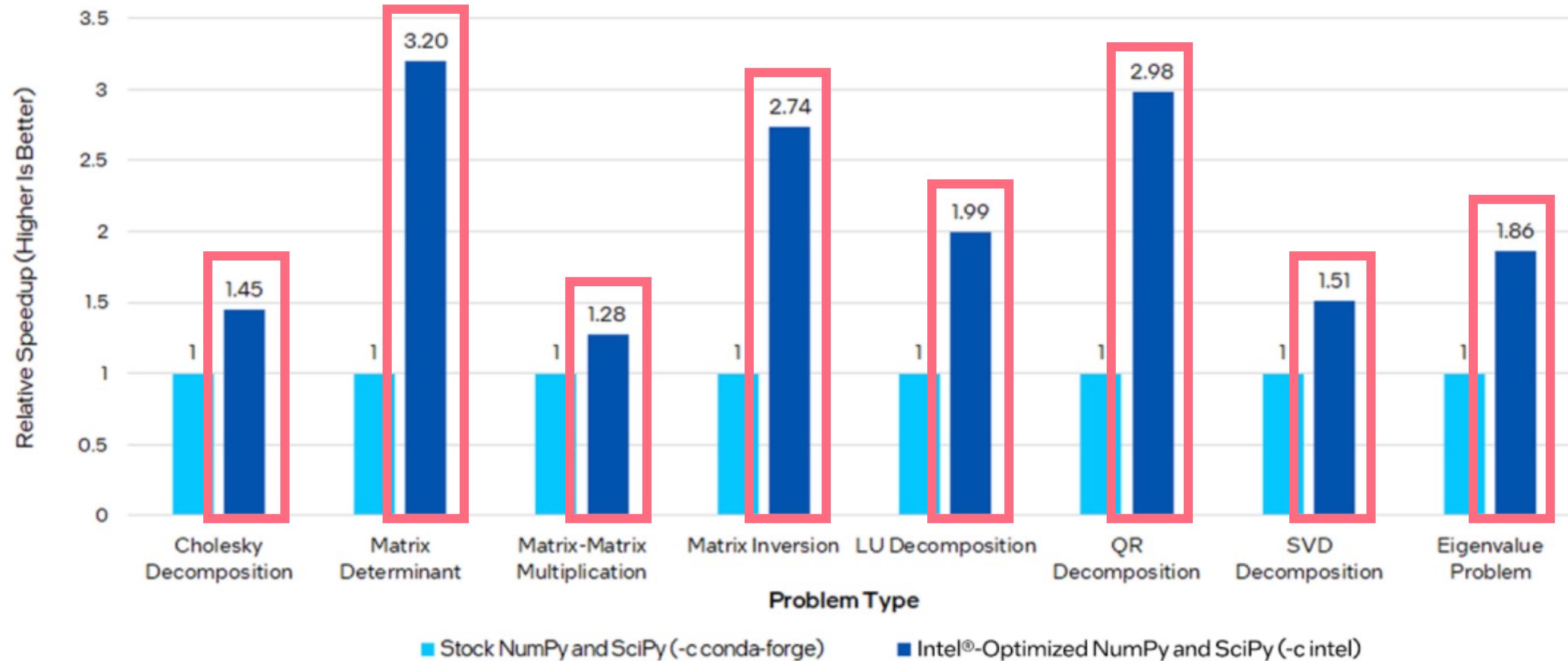
Performance varies by use, configuration and other factors. See backup for configuration details.



intel : intel CPU에 최적화된 Python 배포

Intel®-Optimized NumPy* & SciPy* Linear Algebra Performance

Performance is increased up to 3.2x with Intel optimizations



Intel Optimization for NumPy and SciPy compared to conda-forge channel NumPy and SciPy performance for linear algebra on Intel® Xeon® Platform 8480+

Testing Date: Performance results are based on testing by Intel as of July 15, 2023, and may not reflect all publicly available updates.

Configuration Details and Workload Setup: System: cloud.intel.com, nodes=1:apr:ppn=2, Intel® Xeon® Platinum 8480+, 2 sockets, 56 cores per socket, HT On, Intel® Turbo Boost Technology on, Total memory 528 GB, RAM 33 MHz, Ubuntu® 20.04.5 LTS, 5.18.15-051815-generic, Microcode: 0x2b000310, benchmarks <https://github.com/intelPython/ibench Linear Algebra>, -c conda-forge environment versions: NumPy 1.23.5, SciPy 1.10.1, Numba™ 0.56.4 modules installed, -c intel environment versions: NumPy 1.21.4, SciPy 1.7.3, Numba™ 0.56.3, tbb4py 2021.8.0 modules installed.

Performance varies by use, configuration and other factors. See backup for configuration details.



 **NVIDIA DEVELOPER** Home Blog Forums Docs Downloads Training

Solutions ▾ Platforms ▾ Industries ▾ Resources ▾

CUDA Python

CUDA® Python provides Cython/Python wrappers for CUDA driver and runtime APIs; and is installable today by using PIP and Conda. Python developers will be able to leverage massively parallel GPU computing to achieve faster results and accuracy.

Python is an important programming language that plays a critical role within the science, engineering, data analytics, and deep learning application ecosystem. However, these applications will tremendously benefit from NVIDIA's CUDA Python software initiatives.



NVIDIA의 CUDA 기술을 활용해 Python에서 GPU 가속 연산을 수행할 수 있게 해주는 오픈 소스 프로그램 Python 언어의 간결함과 편리함을 유지하면서, 대규모 데이터 처리와 복잡한 수치 계산 작업을 위한 강력한 GPU 가속 활용 가능



NVIDIA : CUDA Python

```
[1038943] python3.9@127.0.0.1
saxpy, 2021-Apr-02 08:42:41, Context 2, Stream 20
Section: GPU Speed Of Light

DRAM Frequency                                cycle/nsecond      6.47
SM Frequency                                 cycle/nsecond      1.36
Elapsed Cycles                               cycle             474,147
Memory [%]                                    %                92.84
SOL DRAM                                     %                92.84
Duration                                      usecond          348.93
SOL L1/TEX Cache                            %                15.84
SOL L2 Cache                                  %                29.26
SM Active Cycles                            cycle          459,277.39
SM [%]                                       %                12.30

OK     The kernel is utilizing greater than 80.0% of the available compute or memory performance of the device. To
      further improve performance, work will likely need to be shifted from the most utilized to another unit.
      Start by analyzing workloads in the Memory Workload Analysis section.

Section: Launch Statistics

Block Size                                         512
Function Cache Configuration                      cudaFuncCacheNone
Grid Size                                         32,768
Registers Per Thread                           register/thread    16
Shared Memory Configuration Size                 Kbyte            32.77
Driver Shared Memory Per Block                  byte/block       0
Dynamic Shared Memory Per Block                 byte/block       0
Static Shared Memory Per Block                 byte/block       0
Threads                                         thread          16,777,216
Waves Per SM                                    227.56

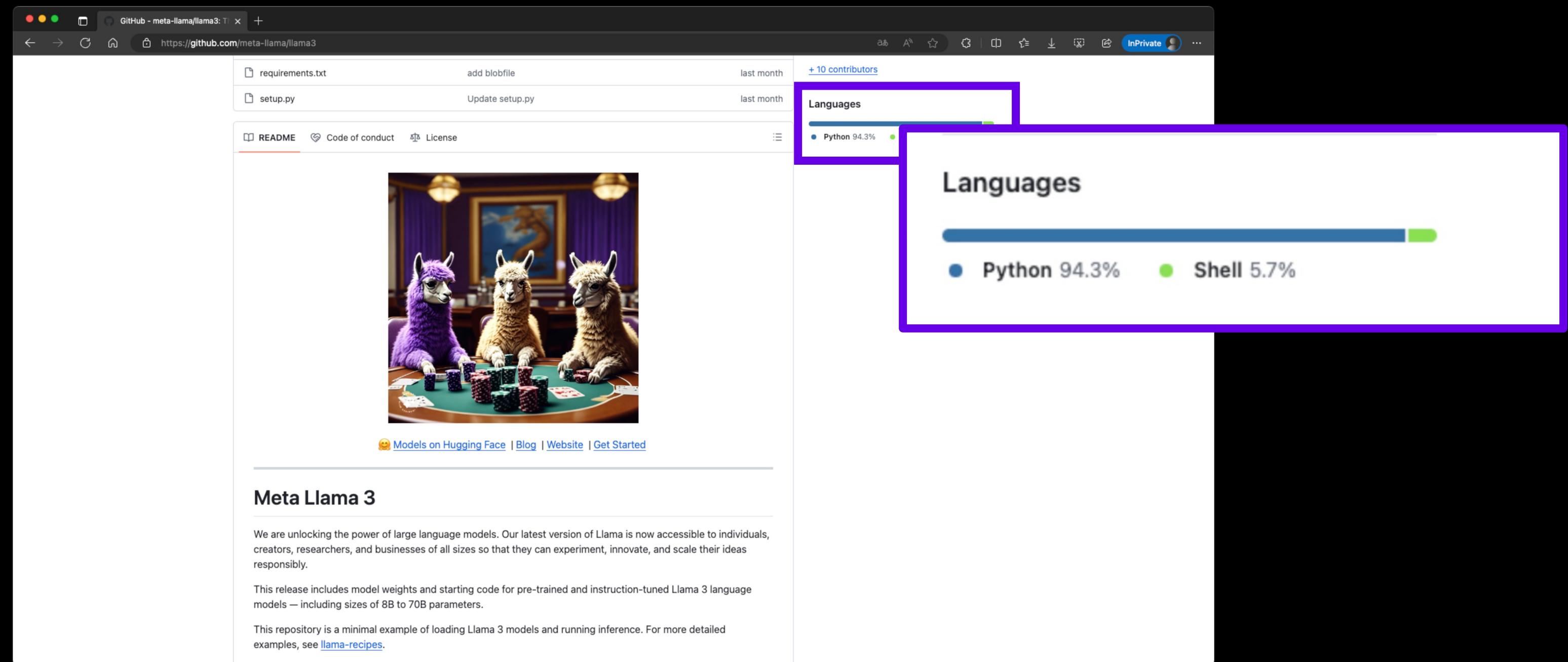
Section: Occupancy

Block Limit SM                                    block           16
Block Limit Registers                         block            8
Block Limit Shared Mem                        block           16
Block Limit Warps                            block            2
Theoretical Active Warps per SM               warp            32
Theoretical Occupancy                          %                100
Achieved Occupancy                           %                87.63
Achieved Active Warps Per SM                  warp           28.04

WRN   This kernel's theoretical occupancy is not impacted by any block limit. The difference between calculated
      theoretical and measured achieved occupancy can be the result of warp scheduling overheads or workload
      imbalances during the kernel execution. Load imbalances can occur between warps within a block as well as
      across blocks of the same kernel.
```

	C++	Python
Kernel execution	352µs	352µs
Application execution	1076ms	1080ms

Meta : Open source LLM - Llama



Meta(전 facebook)에서 제작한 오픈소스 거대언어모델(LLM)인 Llama3는 Python 기반의 Pytorch로 실행 시킬 수 있음

사용자는 자체적으로 LLM을 제작하거나 튜닝할 수 있음



Meta : Open source LLM - Llama

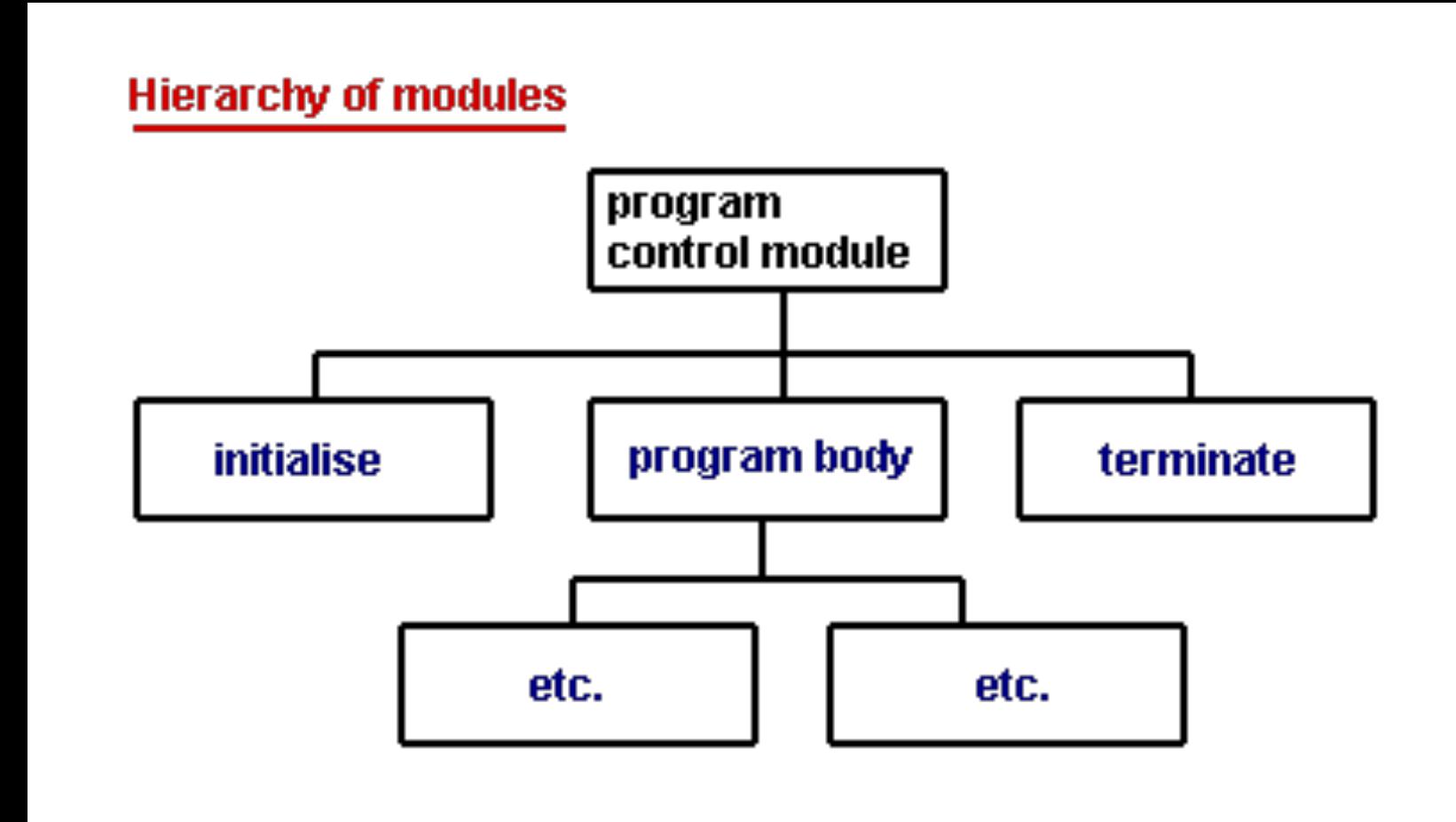
	Llama-3-8B	Gemma 7B - It	Mistral 7B Instruct
MMLU	68.4	53.3	58.4
GPQA	34.2	21.4	26.3
HUMAN EVAL	62.2	30.5	36.6
GSM-8K	79.6	30.6	39.9
MATH	30.0	12.2	11.0

	Llama-3-70B	Gemini Pro 1.5	Claude3 Sonnet
MMLU	82.0	81.9	79.0
GPQA	39.5	41.5	38.5
HUMAN EVAL	81.7	71.9	73.0
GSM-8K	93.0	91.7	92.3
MATH	50.4	58.5	40.5

새로 출시된 Llama3는 다양한 모델과 비교했을 때, 준수한 성능을 보임



모듈 (Module)

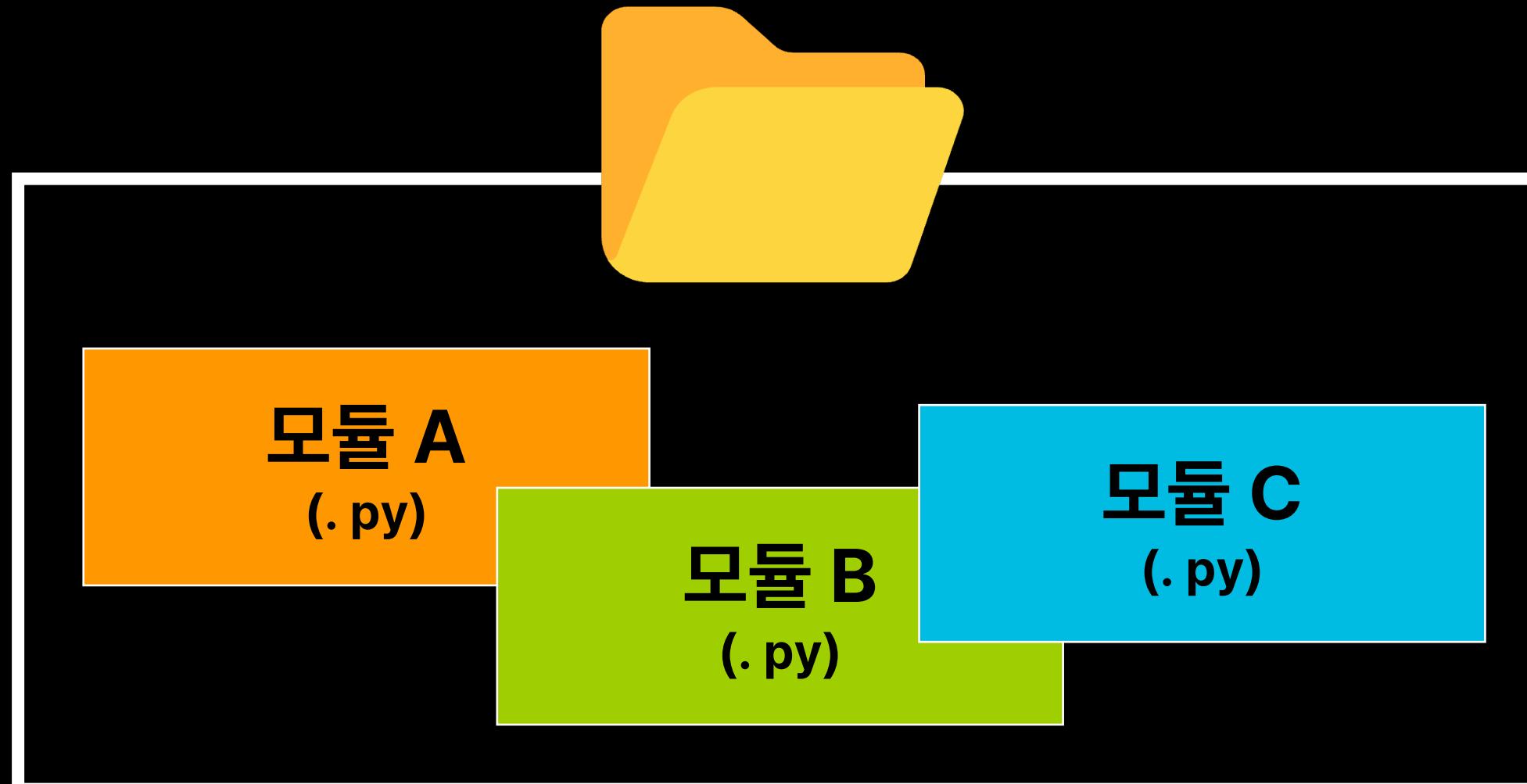


프로그래밍 모듈

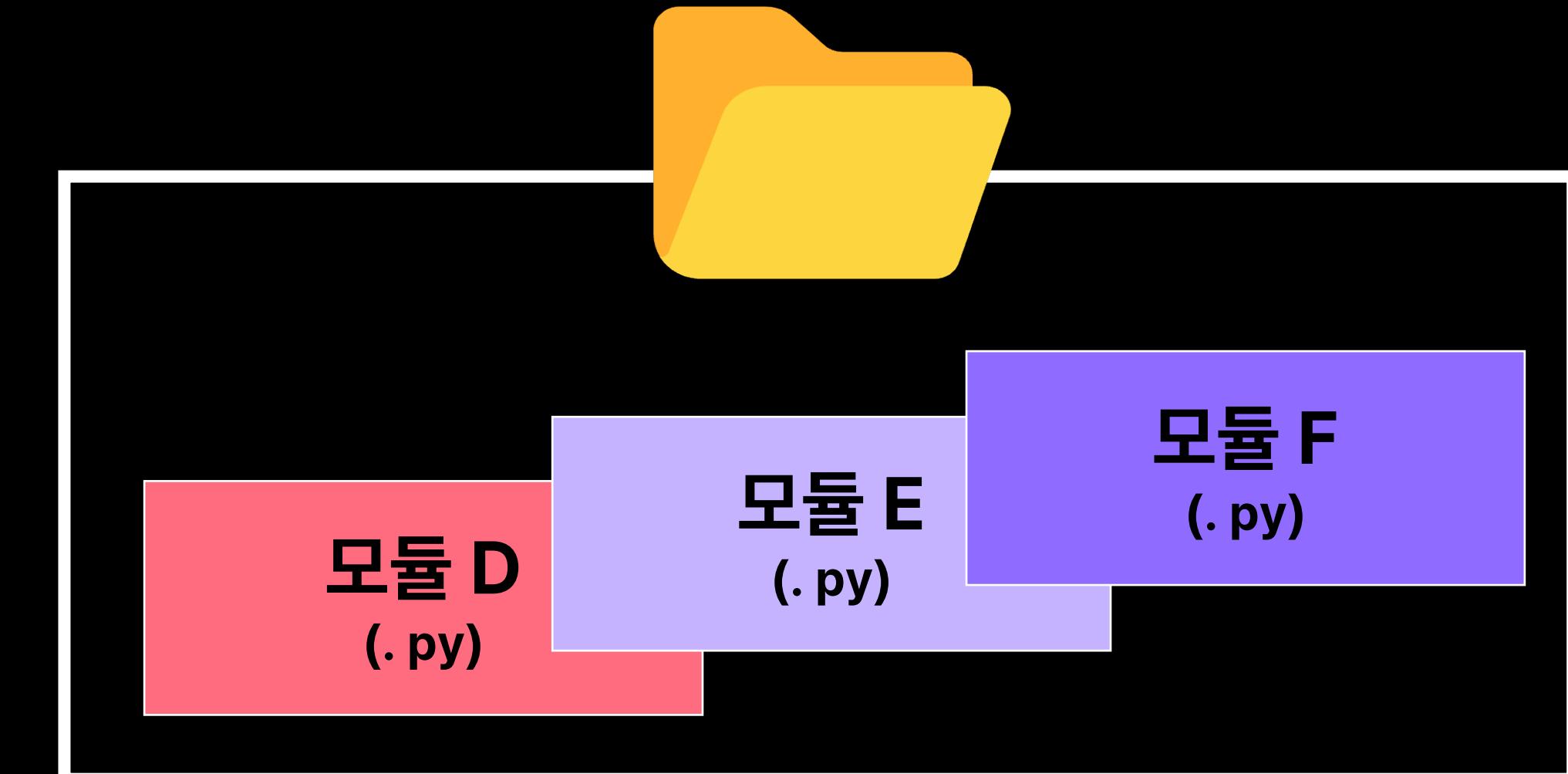
- 단일 파일로 구성되어 함수, 변수 등을 포함한 코드 블록
- 독립적으로 실행 가능하며 교체가능한 프로그램
- 일반적으로 특정한 역할을 수행



패키지 (Package)



folder A = A Package



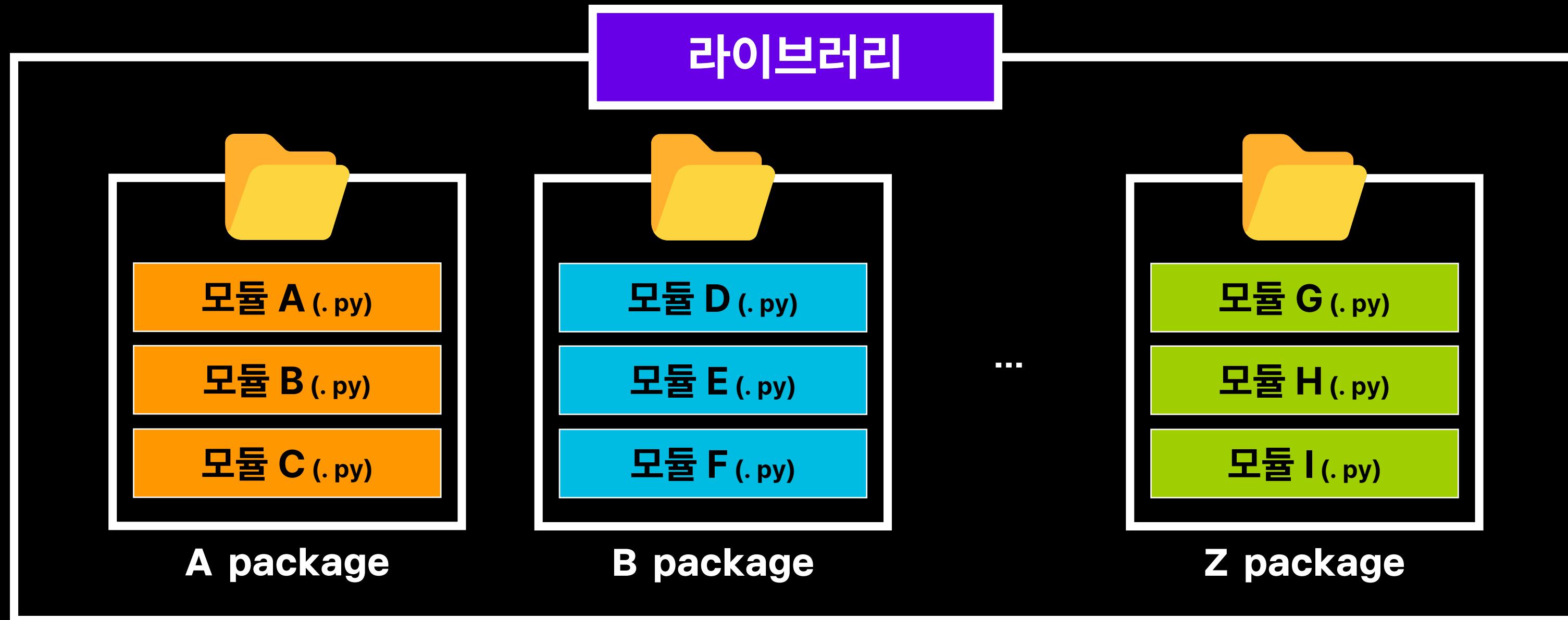
folder B = B Package

패키지

- 여러 모듈을 모아놓은 폴더(디렉토리)
- 관련된 모듈들을 함께 묶어서 관리



라이브러리 (Library)



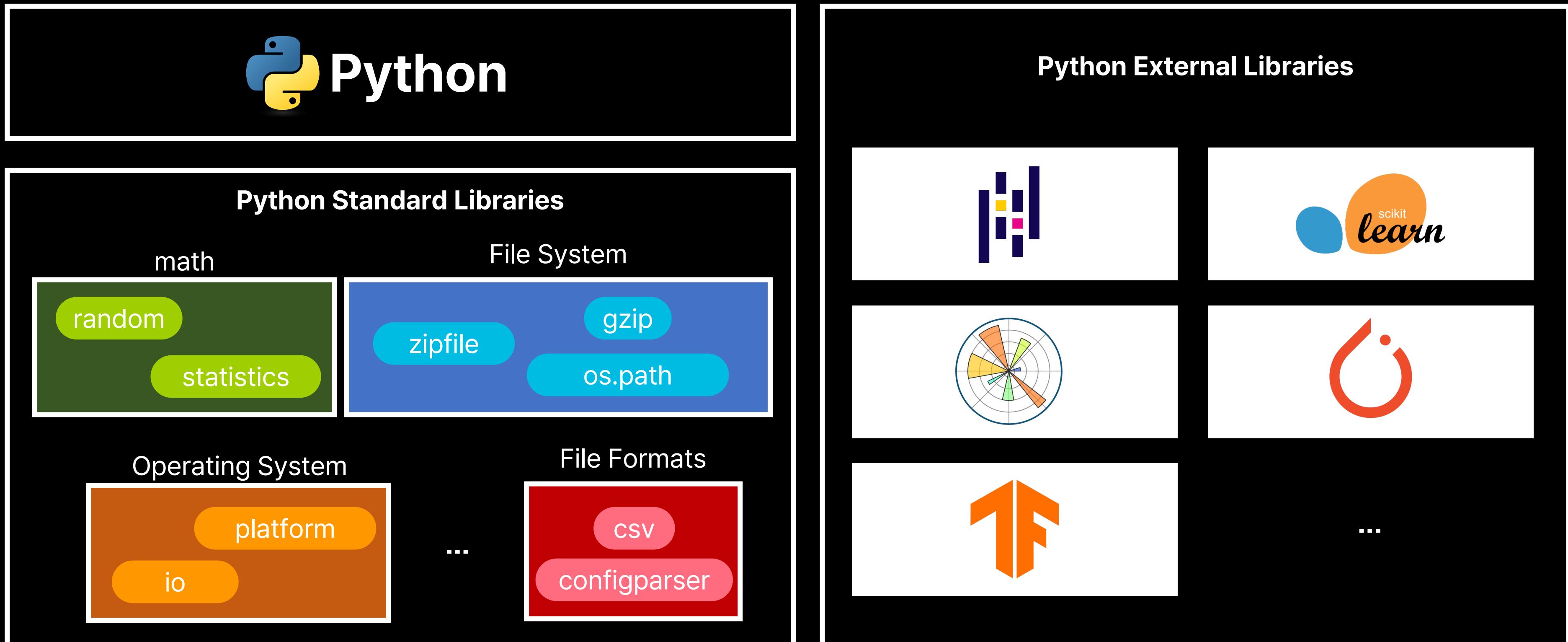
라이브러리

- 재사용 가능한 코드를 비롯한 다양한 자료들의 집합
- 하나 이상의 모듈이나 패키지가 포함될 수 있음



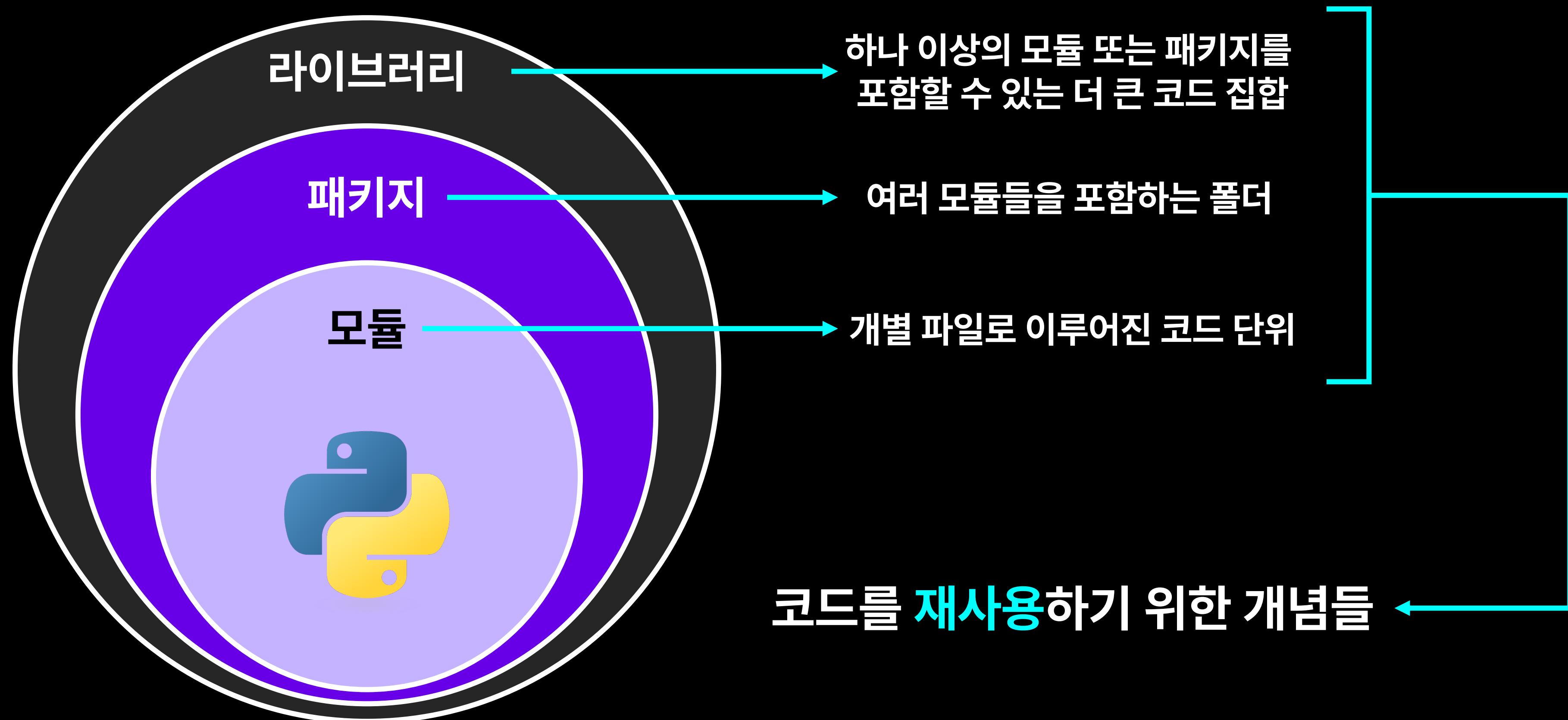
다양한 Python 라이브러리

Python에는 설치가 필요 없는 표준 라이브러리와 설치가 필요한 외부 라이브러리가 존재





모듈부터 라이브러리까지





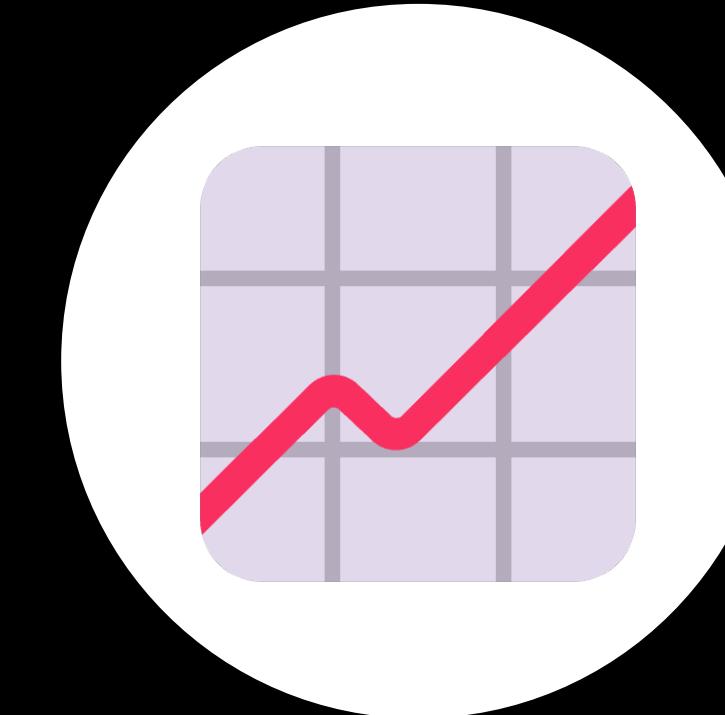
코드를 재사용하는 이유



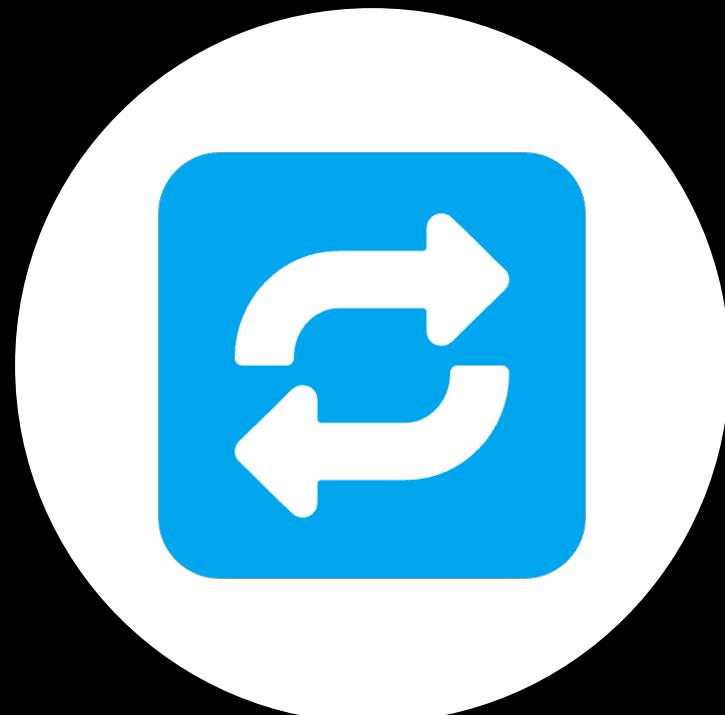
개발시간 단축



비용 절감



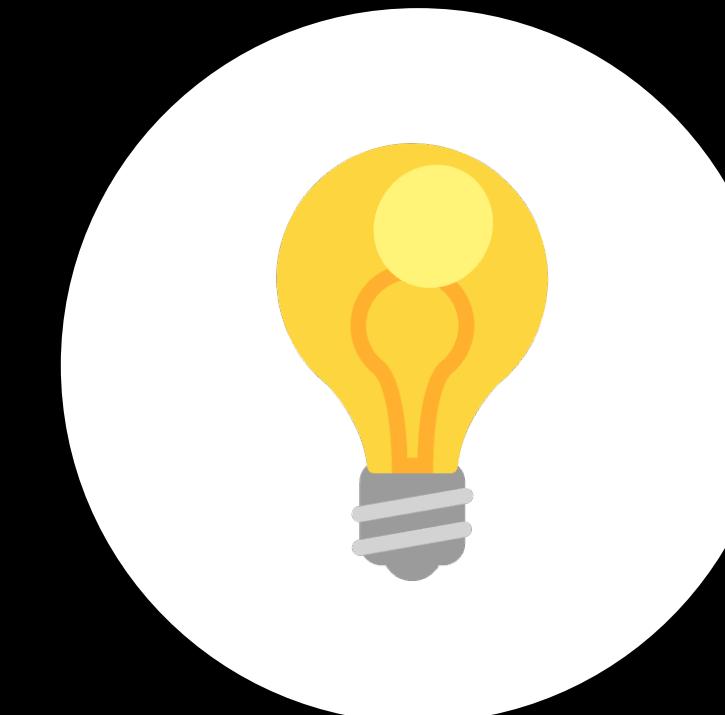
품질 향상



일관성 유지



지식 공유



혁신 촉진



오픈 소스 (Open Source)



코드를 온라인 상에 공개하여 **누구나 자유롭게 사용, 수정, 배포할 수 있도록 허용하는**
소프트웨어 개발 및 배포 방식



오픈 소스의 특징

특징	주요 내용
Free Redistribution	SW 판매나 양도를 제한하지 않고 자유롭게 재배포 허용
Source Code	소스코드와 컴파일 형태를 모두 배포
Derived Works	변경이나 2차 저작물을 허용하고 원래의 SW 라이선스와 동일한 조건으로 배포를 허용
Integrity of The Author's Source Code	수정 목적의 패치파일 형태의 배포를 허용하지만, 원칙상 변경된 소스코드로 빌드가 가능한 SW로 배포하여야 함
No Discrimination Against Persons or Groups	어떠한 개인 및 단체에 대한 차별 금지
No Discrimination Against Fields of Endeavor	SW 사용 분야에 대한 차별 금지
Distribution of Licence	라이선스는 재배포시에도 동일하게 적용
License Must Not Be a Specific to a Product	라이선스는 (유형의) 제품이 아니라 (무형의) SW에 적용
License Must Not Restrict Other Software	같이 배포되는 다른 소프트웨어 대한 제약 금지 (차별 금지)
License Must Be Technology-Neutra	라이선스는 기술에 중립적 (차별 금지)



오픈 소스의 역할과 영향

혁신 촉진

- 오픈 소스는 개발자들이 기존의 코드를 기반으로 새로운 기능을 개발하거나 기존 문제를 해결할 수 있게 함으로써 혁신을 가속화 함
- 기술 발전을 더 빠르고 효율적으로 만듦

표준화 촉진

- 많은 오픈 소스 프로젝트들이 기술의 표준을 설정하는 데 중요한 역할을 함
- 표준화된 기술은 호환성과 상호 운용성을 보장하여 다양한 시스템과 장치에서 소프트웨어가 원활하게 작동하도록 함

보안 강화

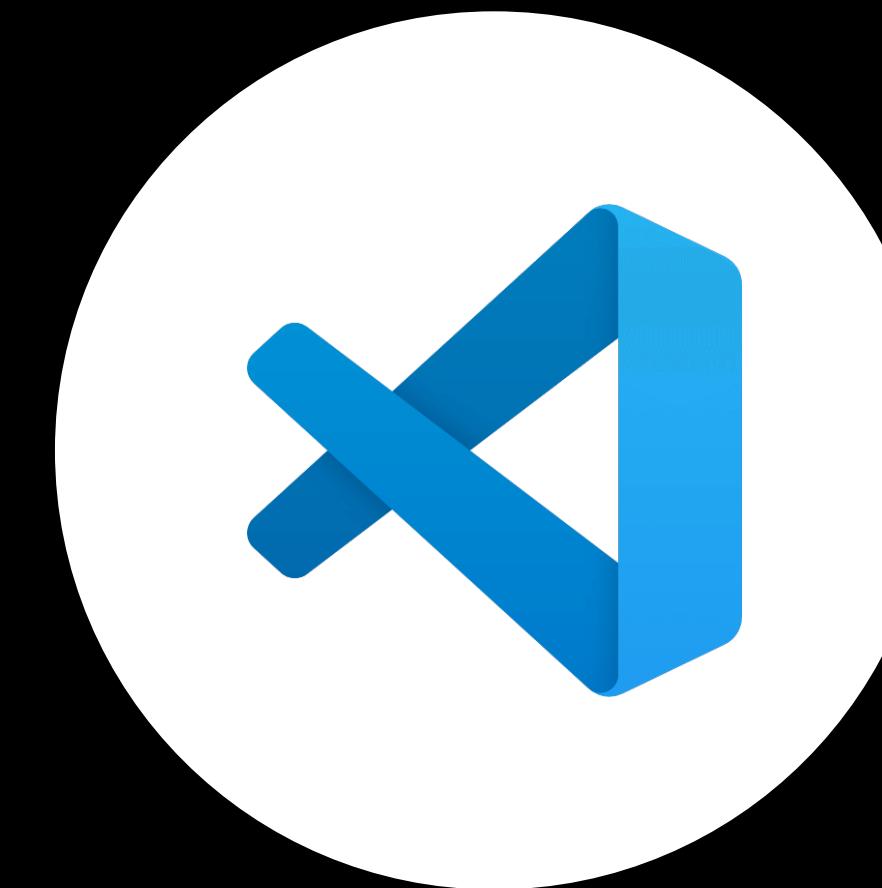
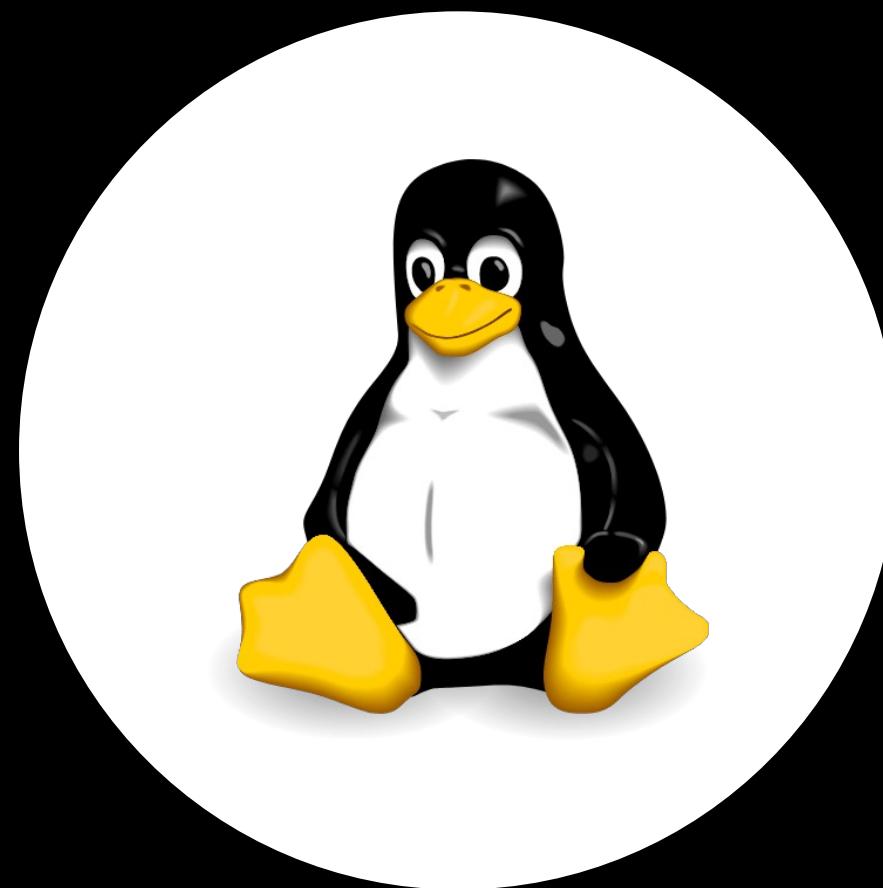
- 오픈 소스 소프트웨어는 코드가 공개되어 있어, 전 세계의 수많은 개발자들이 코드를 검토하고 보안 취약점을 발견하며 이를 개선할 수 있음
- 공동 작업을 통해 효과적으로 소프트웨어의 보안 강화

지속 가능한 개발

- 오픈 소스 프로젝트는 커뮤니티에 의해 유지되기 때문에, 단일 기업이나 개인의 사정에 좌우되지 않고 지속 가능한 개발이 가능
- 소프트웨어의 장기적인 안정성과 개선을 보장

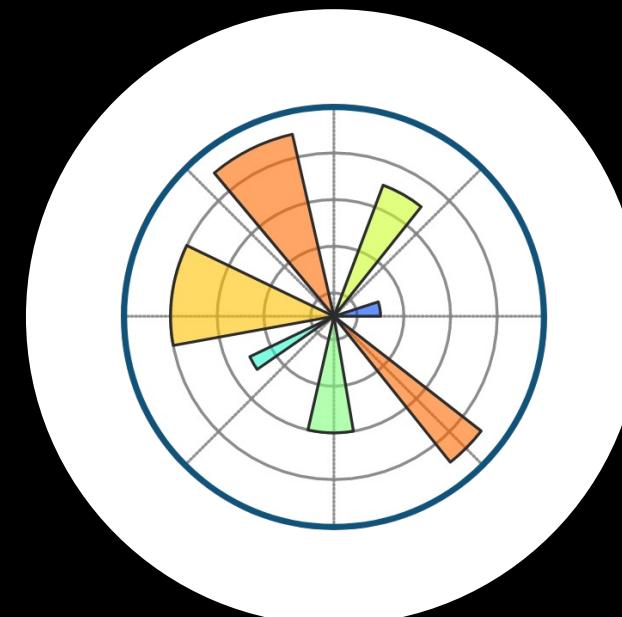
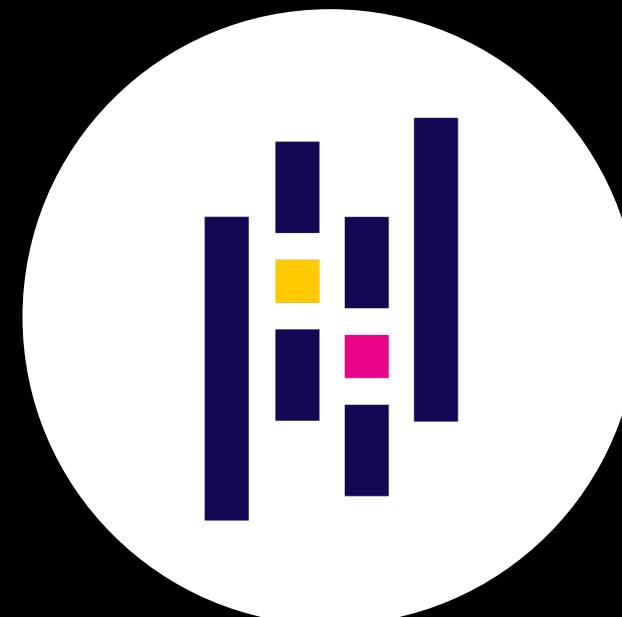


세계에서 가장 유명한 오픈 소스





대표적인 Python 관련 오픈 소스





오픈 소스 License 확인하기 : Pandas

The image shows four separate browser windows on a Mac OS X desktop, all displaying the same GitHub repository: <https://github.com/pandas-dev/pandas>.

- Top Left Window:** Shows the main README page for the Pandas repository. It features the Pandas logo, a brief description ("pandas: powerful Python data analysis toolkit"), and sections like "What is it?", "Table of Contents", and "Testing".
- Top Right Window:** Shows the repository's main page. It includes a file tree on the left, a list of recent commits, and a detailed description of the project: "Flexible and powerful data analysis / manipulation library for Python, providing labeled data structures similar to R data.frame objects, statistical functions, and much more".
- Bottom Left Window:** Shows a file browser interface with the file tree from the top right window. The "LICENSE" file is highlighted with a purple rectangle.
- Bottom Right Window:** Shows the content of the LICENSE file. The text is as follows:

```
BSD 3-Clause "New" or "Revised" License

A permissive license similar to the BSD 2-Clause License, but with a 3rd clause that prohibits others from using the name of the copyright holder or its contributors to promote derived products without written consent.

This is not legal advice. Learn more about repository licenses

StephenShawn Update copyright year (#56960) · 1cc7e19 · 4 months ago · History

Code Blame 31 lines (24 loc) · 1.6 KB

1 BSD 3-Clause License
2
3 Copyright (c) 2008-2011, AQR Capital Management, LLC, Lambda Foundry, Inc. and PyData Development Team
4 All rights reserved.
5
6 Copyright (c) 2011-2024, Open source contributors.
7
8 Redistribution and use in source and binary forms, with or without
9 modification, are permitted provided that the following conditions are met:
10
11 * Redistributions of source code must retain the above copyright notice, this
12 list of conditions and the following disclaimer.
13
14 * Redistributions in binary form must reproduce the above copyright notice,
15 this list of conditions and the following disclaimer in the documentation
16 and/or other materials provided with the distribution.
17
18 * Neither the name of the copyright holder nor the names of its
19 contributors may be used to endorse or promote products derived from
20 this software without specific prior written permission.
21
22 THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
23 AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
24 IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
25 DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
26 FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
27 DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
28 SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
29 CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
30 OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
31 OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

<https://github.com/pandas-dev/pandas>



오픈 소스 License 확인하기 : Pandas

The image shows a Mac desktop with four browser windows open, illustrating the process of verifying the open source license for the Pandas library:

- Top Left Window:** GitHub - pandas-dev/pandas repository page. It displays the repository's main README file, which includes the Pandas logo and a brief description: "pandas: powerful Python data analysis toolkit". Below this are sections for Testing, Package, and Meta information.
- Top Right Window:** A standard GitHub repository page for the pandas-dev/pandas repository. It shows the repository's activity, including recent commits by tuhinsharma121 and a list of files in the repository.
- Bottom Left Window:** A file browser showing the directory structure of the repository. The "LICENSE" file is highlighted with a purple rectangle.
- Bottom Right Window:** The contents of the LICENSE file. The text is as follows:

```
BSD 3-Clause License
Copyright (c) 2011-2024, Open source contributors.

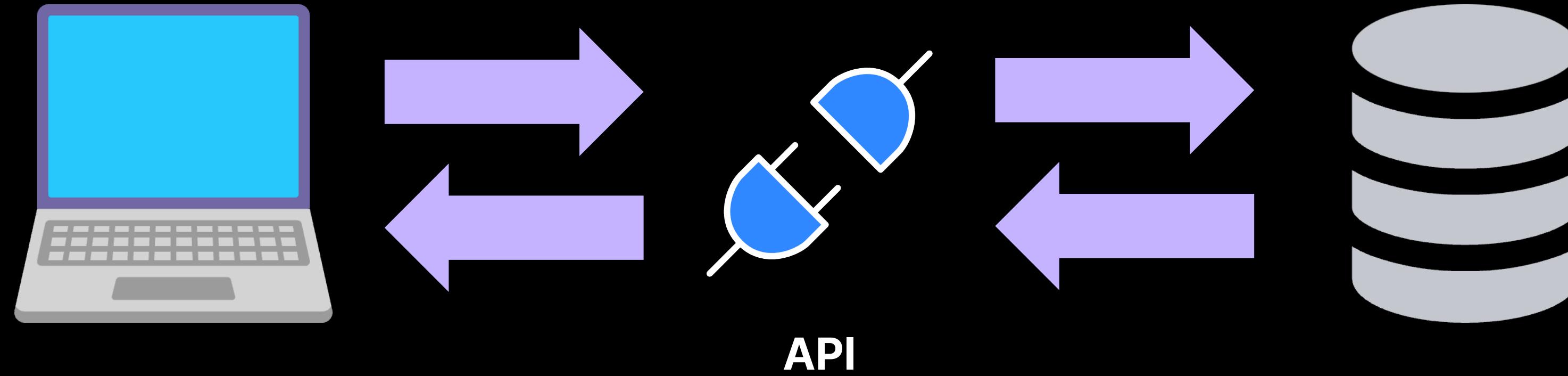
Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are met:
* Redistributions of source code must retain the above copyright notice, this
  list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above copyright notice,
  this list of conditions and the following disclaimer in the documentation
  and/or other materials provided with the distribution.
* Neither the name of the copyright holder nor the names of its
  contributors may be used to endorse or promote products derived from
  this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS"
AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE
FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER
CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY,
OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE
OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

<https://github.com/pandas-dev/pandas>



API (Application Programming Interface)

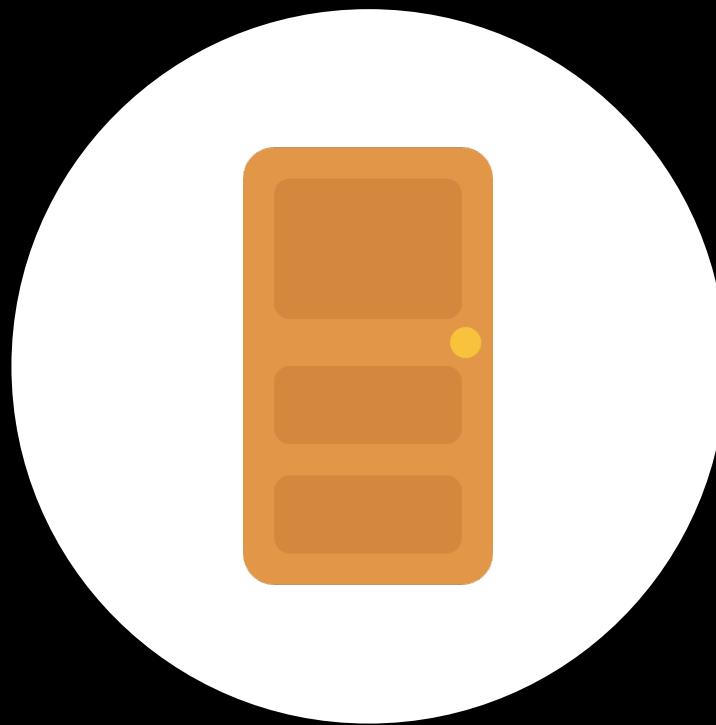


소프트웨어 간의 **상호작용**을 가능하게 하는 규약이나 인터페이스

→ 어떠한 응용 프로그램(웹 브라우저, 워드 프로세서, 데이터 베이스, 애플리케이션 등)에서
데이터를 전달해주는 방법



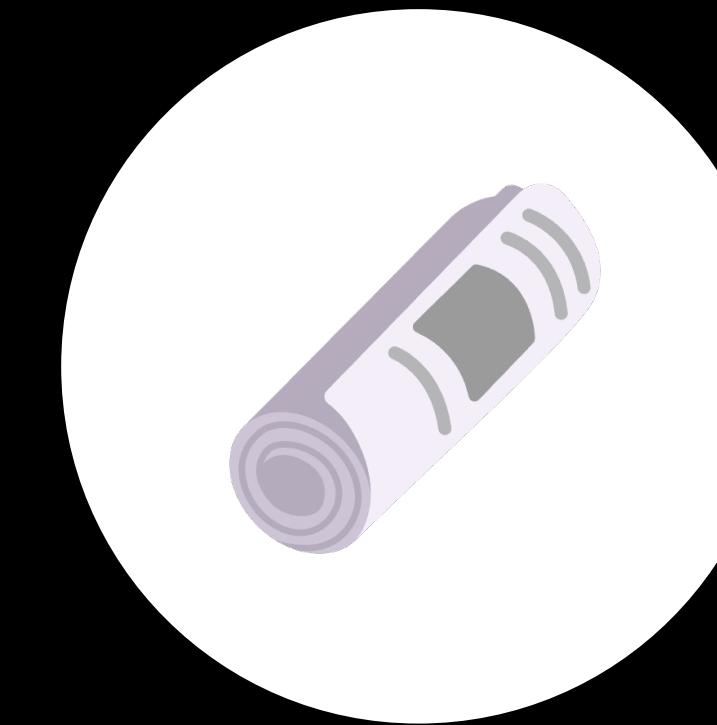
API 역할과 필요성



서버와 데이터베이스의
출입구 역할



접속 표준화



정보 제공



대표적인 API



API 예시 1 : 저장된 데이터 요청하기

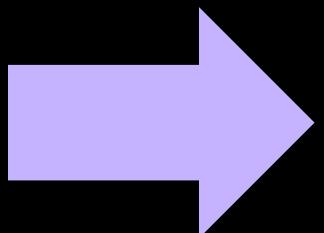
```
import requests

url = 'http://apis.data.go.kr/B090041/openapi/service/RiseSetInfoService/getAreaRiseSet
Info'

params ={'serviceKey' : '',
         'locdate' : '20150101',
         'location' : '서울'
     }

response = requests.get(url, params=params)
```

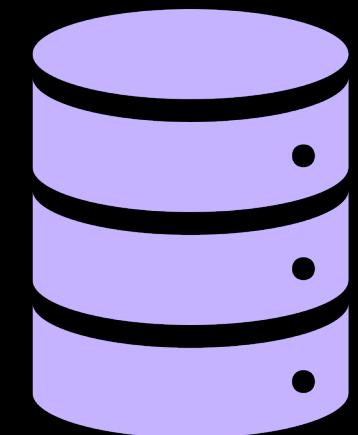
요청



오픈API 상세

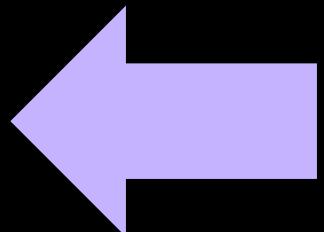
DATA . GO . KR
공공데이터포털

XML 한국천문연구원_출몰시각 정보



```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<response>
    <header>
        <resultCode>00</resultCode>
        <resultMsg>NORMAL SERVICE.</resultMsg>
    </header>
    <body>
        <items>
            <item>
                <aste>1858  </aste>
                <astm>0613  </astm>
                <civile>1753  </civile>
                <civilm>0718  </civilm>
                <latitude>3733</latitude>
                <latitudeNum>37.5500000</latitudeNum>
                <location>\xec\x84\x9c\xec\x9a\xb8  </location>
                <locdate>20150101</locdate>
                <longitude>12658</longitude>
                <longitudeNum>126.5800000</longitudeNum>
            </item>
        </items>
    </body>
</response>
```

응답





API 예시 2 : 생성AI 이용해보기

```
from openai import OpenAI

client = OpenAI(api_key="open ai api key")

response = client.chat.completions.create(
    model="gpt-3.5-turbo",
    messages=[
        {
            "role" : "user",
            "content" : "python 언어에 대해 알려주세요."
        }
    ]
)
```

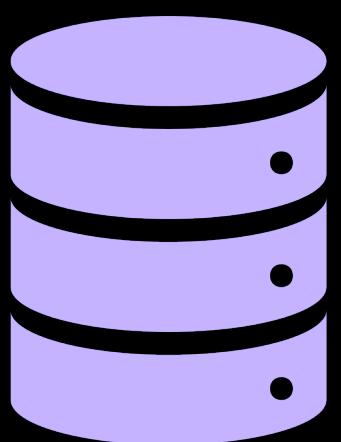
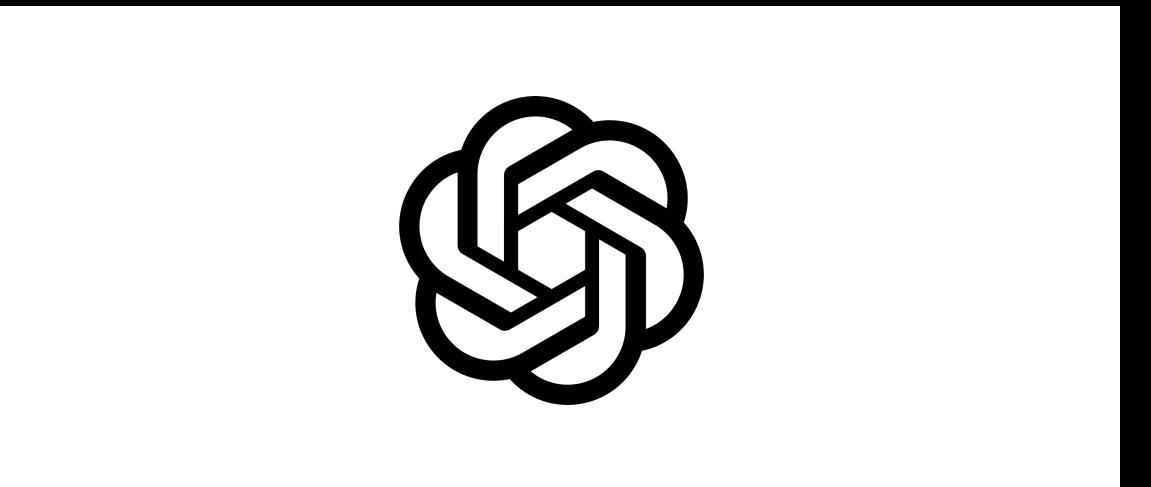
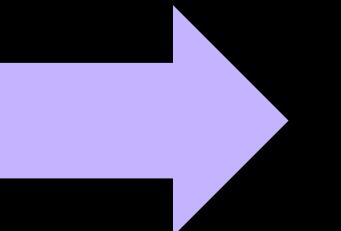
Python은 간결하고 가독성이 높은 프로그래밍 언어로, 귀도 반 로섬이 1991년에 발표했습니다. Python은 객체지향, 함수형, 동적 타이핑 등 다양한 프로그래밍 스타일을 지원하며, 다양한 플랫폼에서 사용할 수 있는 인터프리터 언어입니다.

Python은 다양한 분야에서 사용되고 있으며, 웹 개발, 데이터 분석, 인공지능, 머신러닝, 빅 데이터 등에 널리 사용됩니다. 또한 Python은 많은 오픈소스 라이브러리와 프레임워크가 제공되어 있어 개발 시간을 단축하고 효율적인 개발이 가능합니다.

또한 Python은 높은 생산성을 제공하기 때문에 초급자부터 전문가까지 다양한 사용자가 쉽게 배울 수 있습니다. 이러한 이유로 Python은 프로그래밍 언어 중에서도 인기가 매우 높고, 산업에서도 널리 사용되고 있습니다.

ChatGPT의 대답은 달라질 수 있습니다.

요청



응답



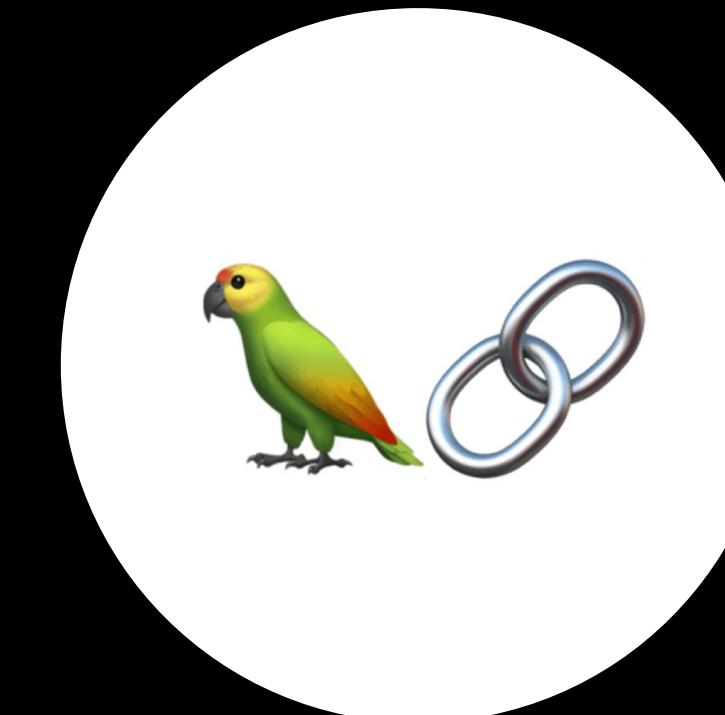
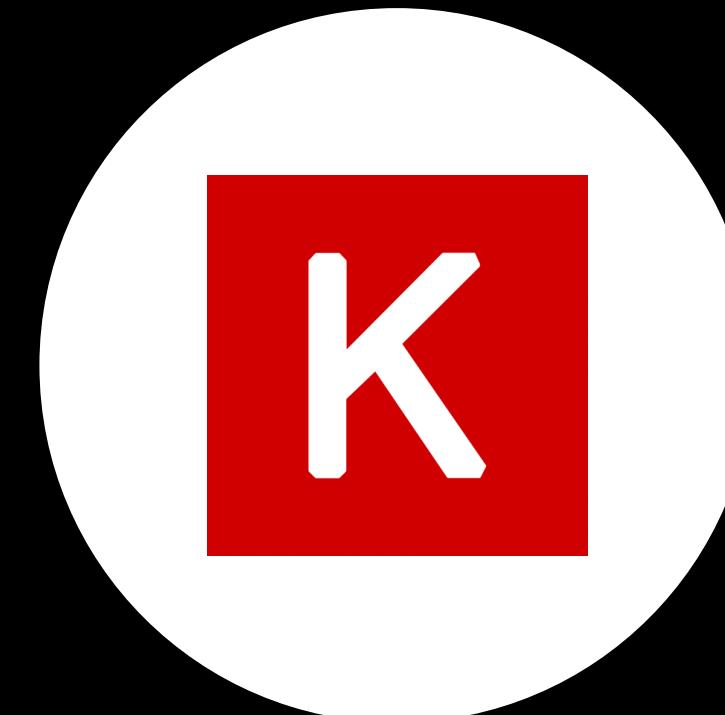
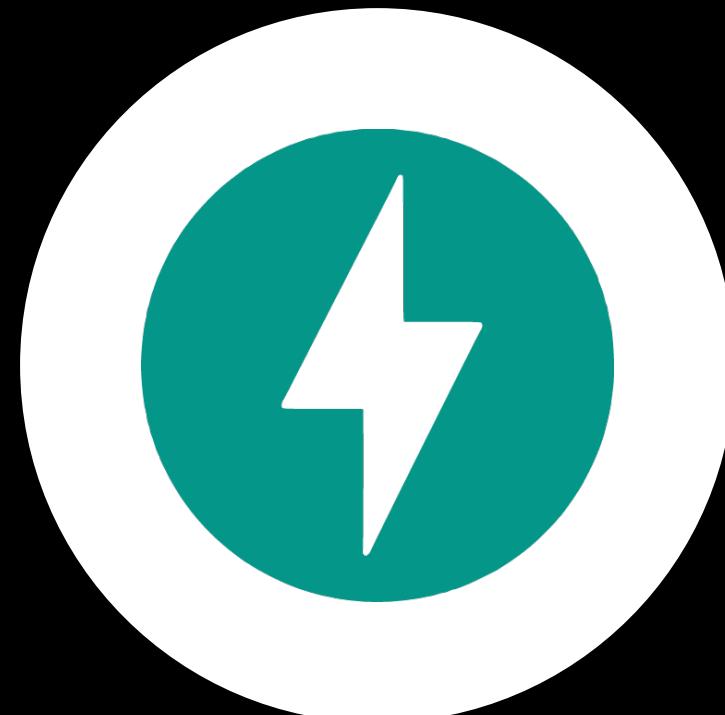
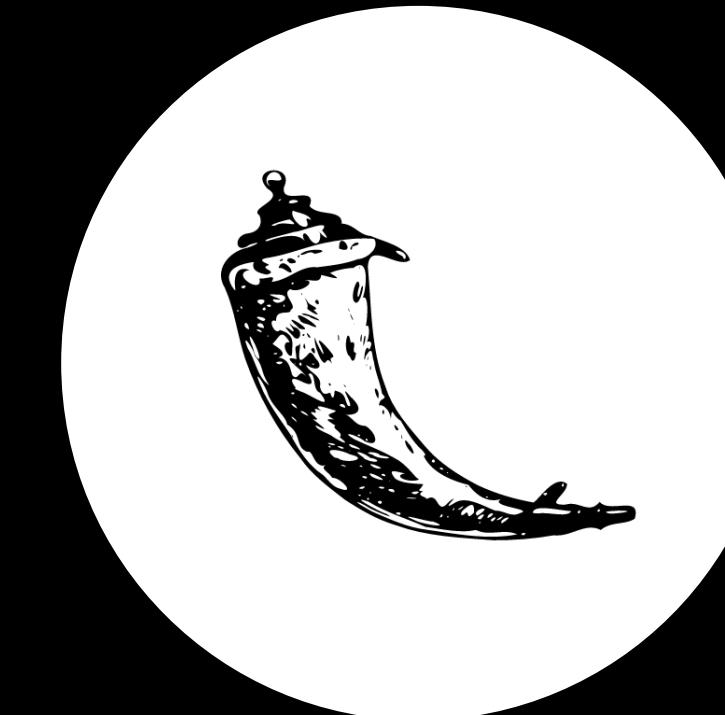
프레임워크 (Framework)



프레임워크는 특정 애플리케이션을 개발하기 위한
기본 구조와 지원을 제공하는 소프트웨어의 집합



대표적인 프레임워크





프레임워크를 사용하는 이유

case 1. Streamlit 프레임워크 미사용

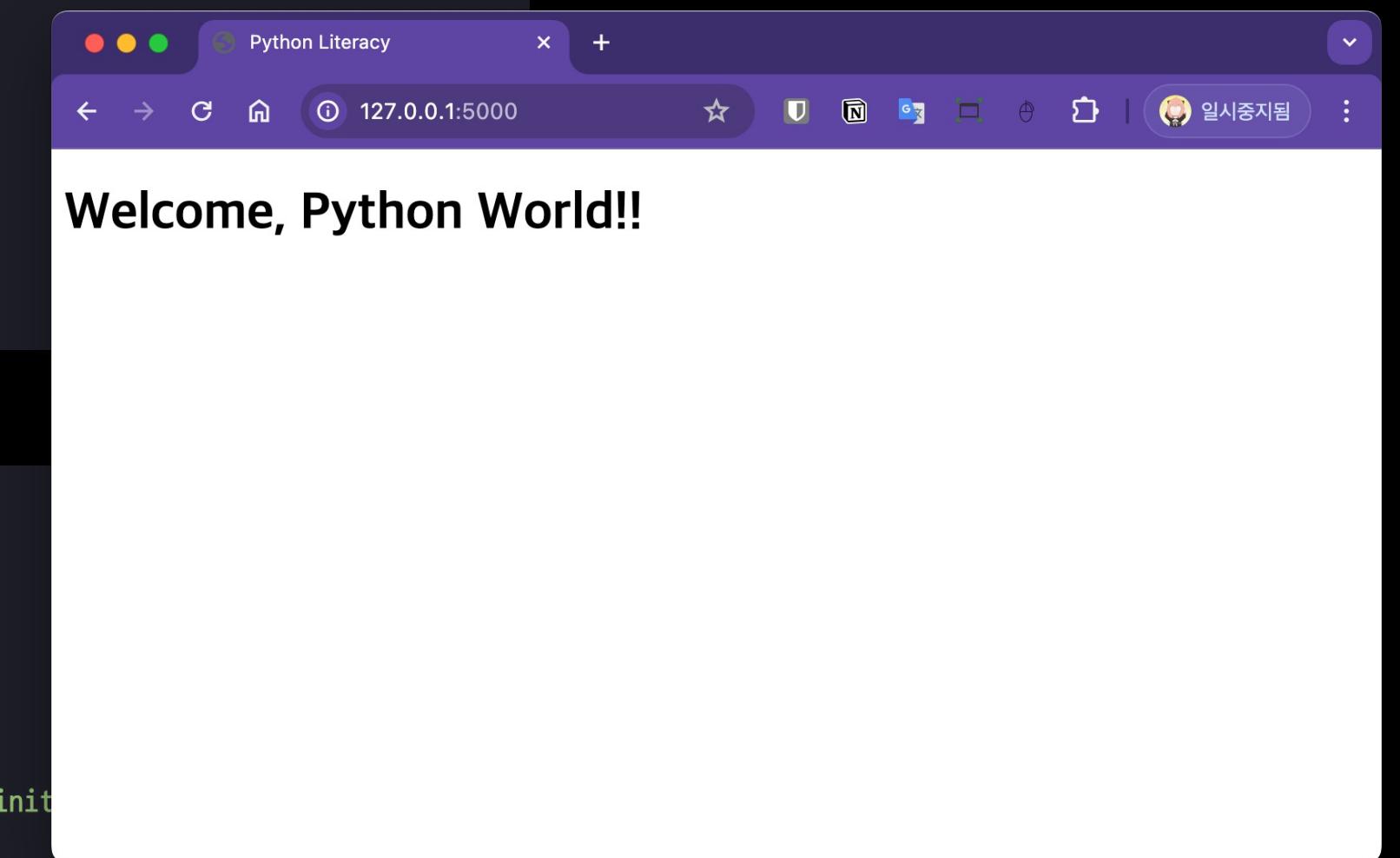


app.py

```
from flask import Flask, render_template  
  
app = Flask(__name__)  
  
@app.route('/')  
def home():  
    return render_template('index.html')  
  
if __name__ == '__main__':  
    app.run(debug=True)
```

index.html

```
<!-- templates/index.html -->  
  
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <meta charset="UTF-8">  
    <meta name="viewport" content="width=device-width, initial-scale=1.0">  
    <title>Python Literacy</title>  
</head>  
<body>  
    <h1>Welcome, Python World!!</h1>  
</body>  
</html>
```

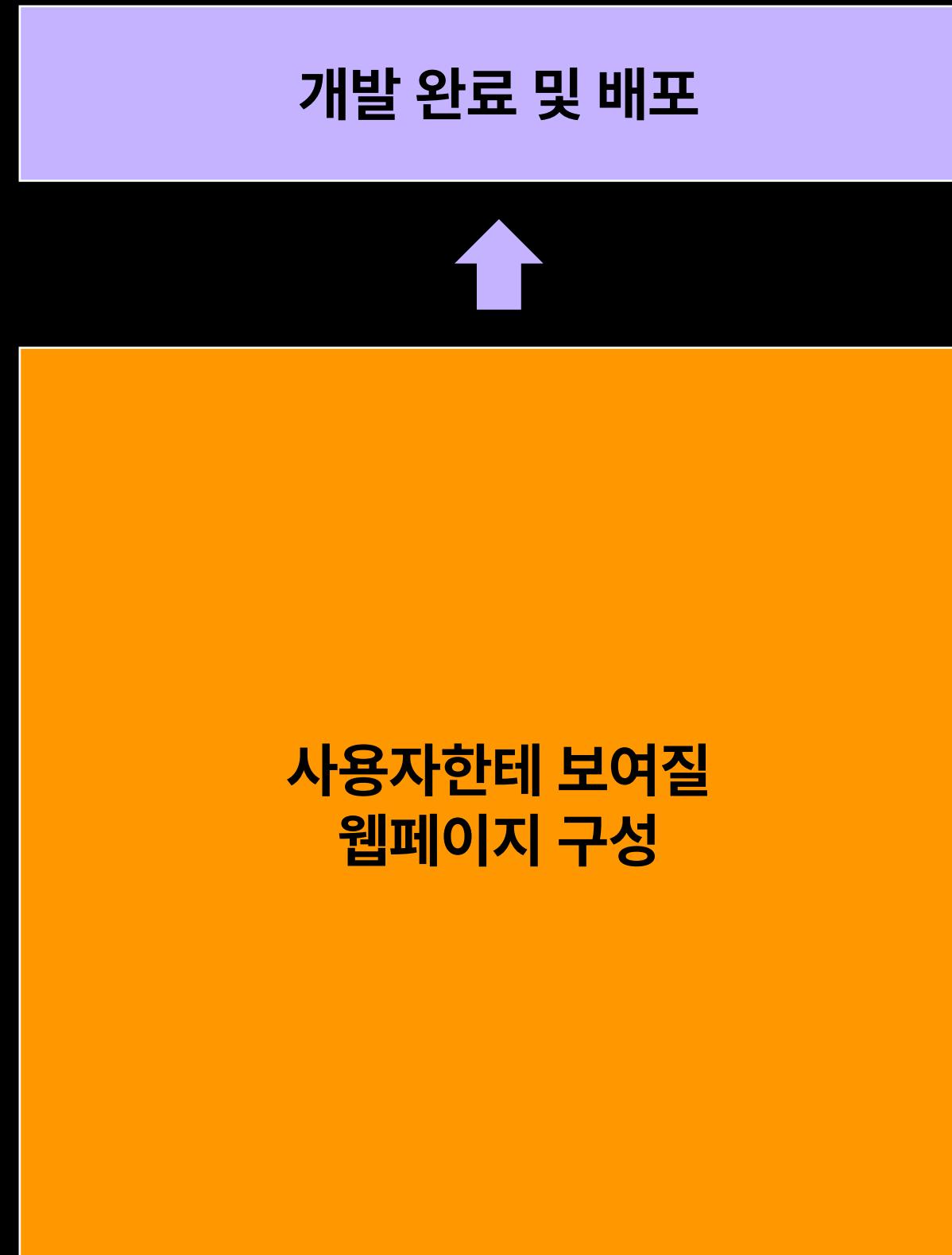


소개되는 개발 과정은 반드시 이 순서에 따르지 않으며 조직에 따라 다른 단계로 개발될 수 있습니다



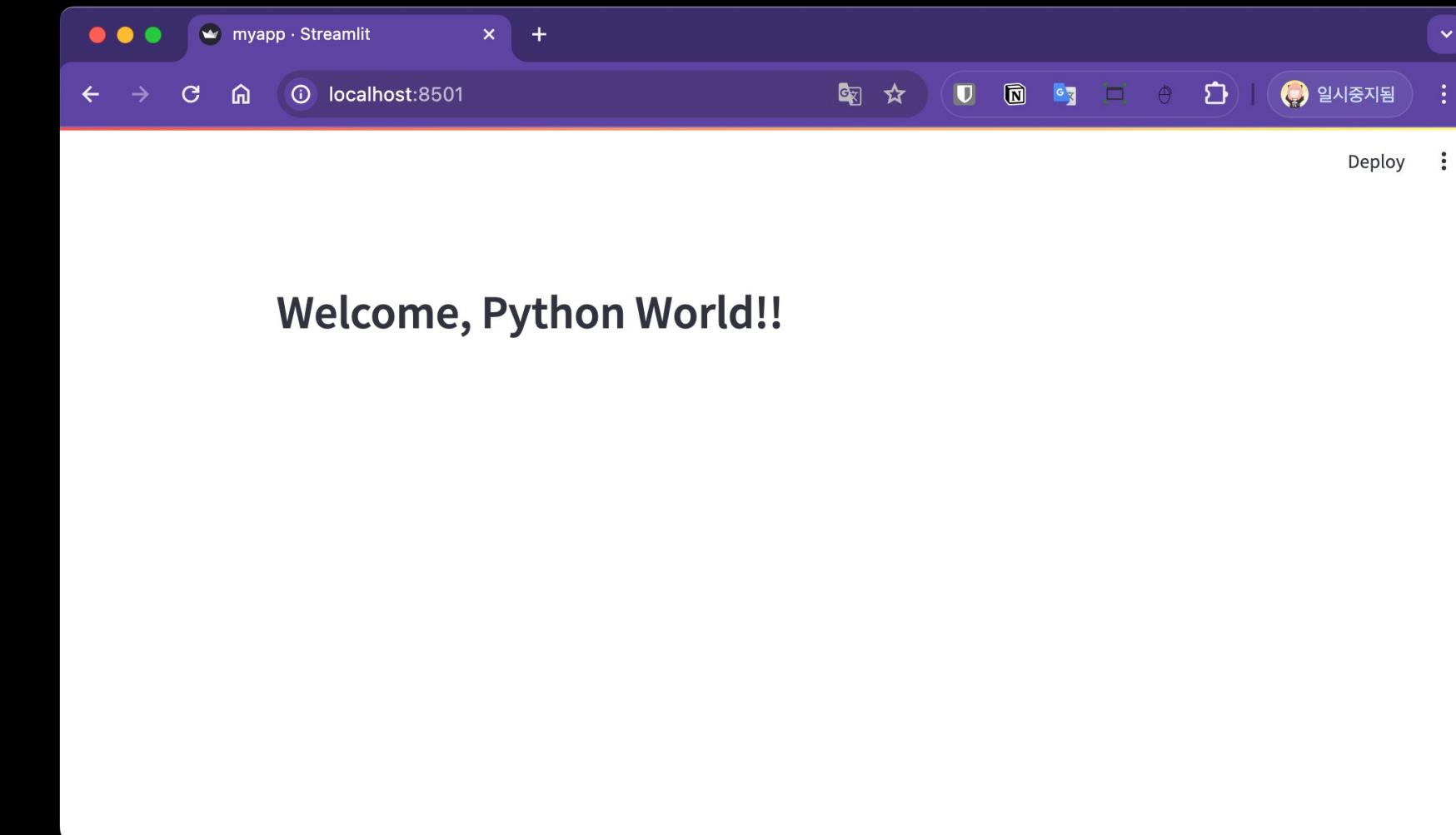
프레임워크를 사용하는 이유

case 2. Streamlit 프레임워크 사용



main.py

```
import streamlit as st  
  
st.header("Welcome, Python World!!")
```





프레임워크 vs API

- **프레임워크, API는 소프트웨어 개발 과정에서 중요한 역할을 담당**
- **프레임워크**
 - 개발하고자 하는 애플리케이션의 전체적인 구조와 개발 방식을 제공
 - “어떻게” 개발할 것인지에 대한 방향과 도구를 제공
- **API**
 - 소프트웨어 간의 상호작용을 위한 규약
 - “무엇”을 할 수 있는지, 특정 기능이나 데이터에 접근하는 방법 제공

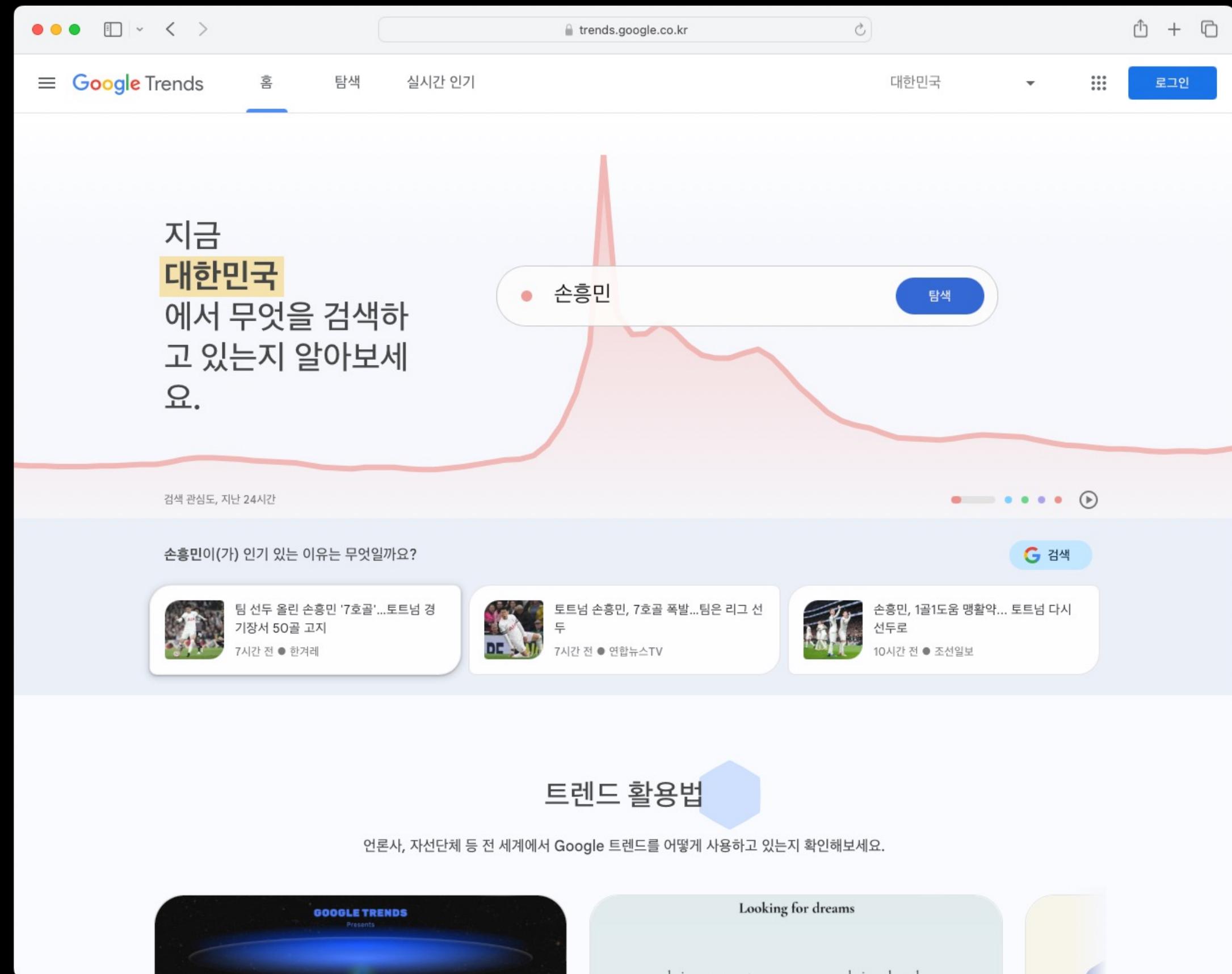
다양하고 빠르게 변하는 소비자 선호도

- Facebook, Instagram 등의 SNS가 발전하면서 소비자의 선호도 변화가 활발해짐
- SNS에 올라오는 게시글을 수작업으로 수집하는 것에 비해 생산량을 따라갈 수 없음
- 빠르고 원하는 데이터만 수집할 수 있는 자동화된 기술 필요
- Python과 크롤링 기술을 도입함으로써 검색 트렌드를 확인하거나 효과적으로 원하는 웹페이지의 데이터를 수집 가능





트렌드 분석하기 위한 사이트



- 구글 트렌드는 여러 지역과 언어로 구글 검색의 검색 쿼리의 인기도를 분석하는 웹사이트
- 메인 화면에서는 현재 인기 검색 트렌드와 구글 트렌드에 대한 튜토리얼과 같은 아티클을 무료로 볼 수 있음



구글 트렌드 확인하기

- Pytrends는 구글 트렌드 데이터를 가져오는데 사용되는 Python 라이브러리
- Python을 이용하면 원하는 키워드 별로 트렌드 데이터를 확인하고 시각화하여 트렌드 파악 가능

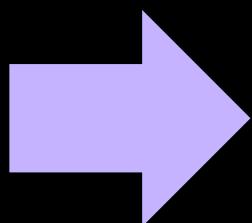
```
from pytrends.request import TrendReq

pytrends = TrendReq(hl = 'ko', tz = 540)

kw_list = ["귤", "수박", "레몬"]

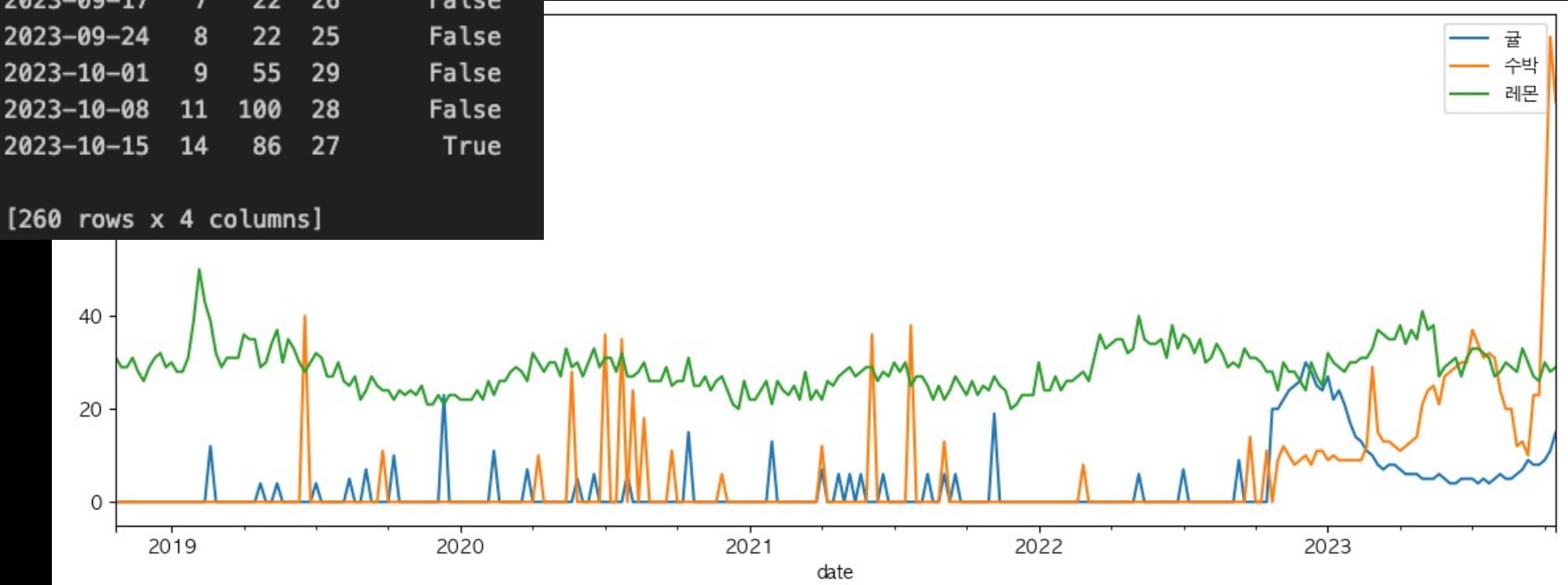
pytrends.build_payload(kw_list,
                       cat=0,
                       timeframe='today 5-y',
                       geo='KR',
                       gprop='')

data = pytrends.interest_over_time()
print(data)
```



	귤	수박	레몬	isPartial
date				
2018-10-28	0	0	29	False
2018-11-04	0	0	28	False
2018-11-11	0	0	28	False
2018-11-18	0	0	29	False
2018-11-25	0	0	26	False
...
2023-09-17	7	22	26	False
2023-09-24	8	22	25	False
2023-10-01	9	55	29	False
2023-10-08	11	100	28	False
2023-10-15	14	86	27	True

[260 rows x 4 columns]





크롤링 (Crawling)

- **크롤링(웹 크롤링, Web Crawling)**은 인터넷 상에서 웹 페이지를 방문하여 데이터를 수집하는 자동화 방법
- 인터넷의 방대한 양을 사람이 일일이 파악하는 것은 불가능
- 컴퓨터 프로그램을 이용해서 미리 지정한 “규칙”에 따라 웹 페이지를 탐색
- 크롤링 행위를 하는 소프트웨어를 **크롤러(Crawler)**라고 함



식품 리뷰 데이터 수집

The screenshot displays the Sometrend platform's user interface for food review data collection. At the top, there are date filters (시작일: 2024.05.20, 종료일: 2024.05.26), time intervals (1개월, 3개월, 6개월, 12개월), and various data sources (커뮤니티, 인스타, 블로그, 뉴스, X(트위터), 리트윗제거, 채널적용) with toggles. Below this, the main dashboard includes:

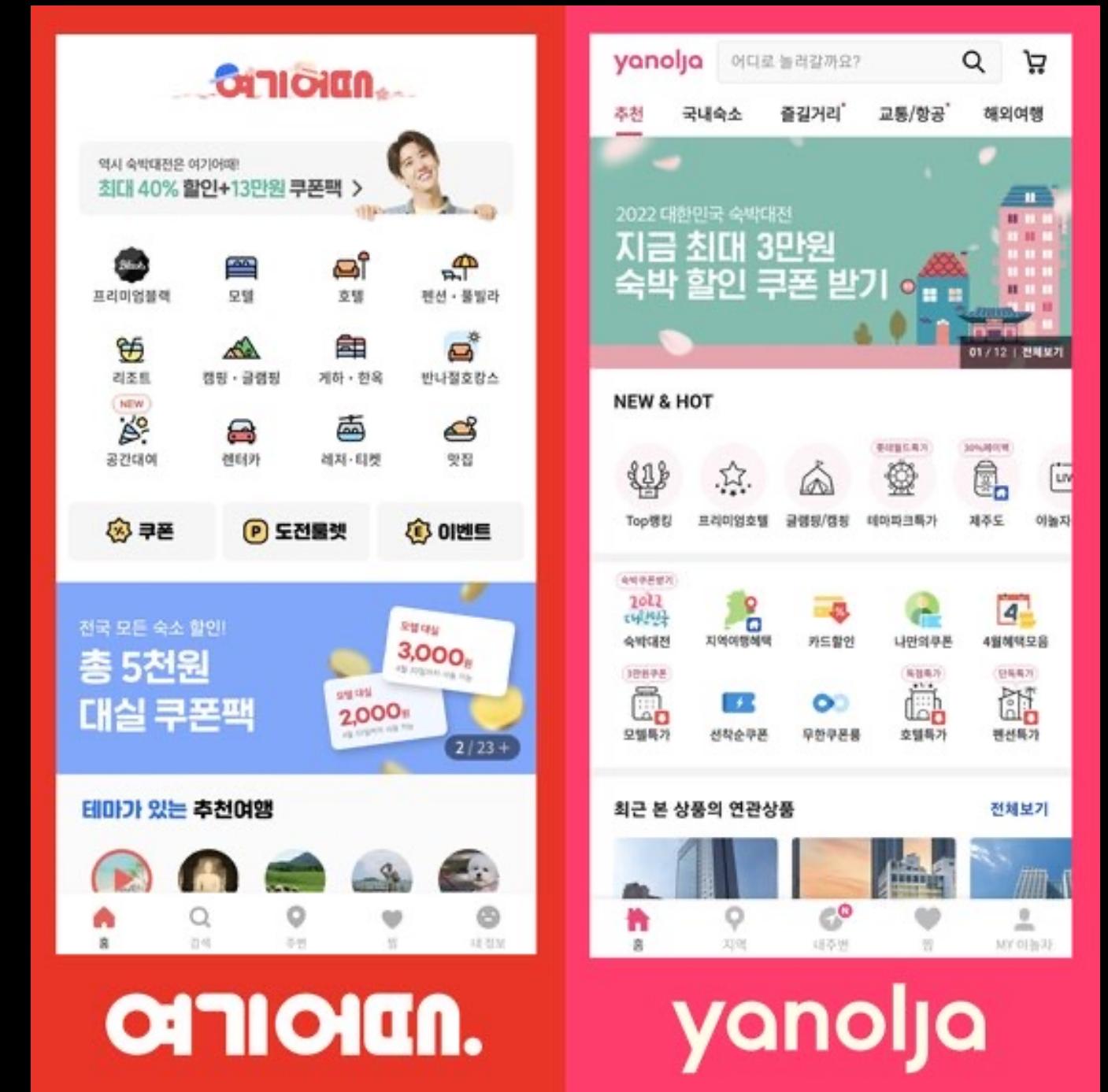
- 언급량 분석 (Mention Volume Analysis):** Shows a chart for '블로그' (Blog) mentioning the lowest volume on May 20, 2024.
- 언급량 추이 (Trend Analysis):** A line graph showing the trend of mentions from May 20 to May 26, 2024, comparing '블로그' (green line) and '뉴스' (yellow line). The green line shows a general downward trend, while the yellow line shows a slight increase around May 25.
- 오뚜기 (Ottogi) Posts:** A sidebar listing several posts from the Ottogi brand, such as reviews of their ramen products and a comparison of Ottogi ramen with other brands like Nongshim.
- 유튜브 반응 확인 (YouTube Reaction Check):** A section showing thumbnails of YouTube videos related to Ottogi products.

- 사용자에 의해 실시간으로 생성되는 SNS 게시글, 댓글 등의 데이터를 크롤러를 이용하여 수집
- 브랜드 및 제품의 호감도 분석, 고객 만족도 등을 파악할 수 있음
- 시장조사에 소요되는 시간을 대폭 축소할 수 있으며 보다 정성적인 데이터를 수집할 수 있음



크롤링 사용 시 주의사항

- 수집한 데이터를 **영리적, 상업적 용도로 활용할 때 주의 필요**
- 데이터의 가치가 높아지면서 많은 플랫폼에서 데이터를 법적으로 기술적으로 보호하는 추세
- “여기어때”에서 “야놀자”의 정보를 수시로 취합할 수 있는 크롤링 프로그램을 개발하여 법적 분쟁 발생
 - 형사 재판 결과 : 무죄
 - 민사 재판 결과 : 여기어때에 대해 10억원의 손해배상책임을 인정



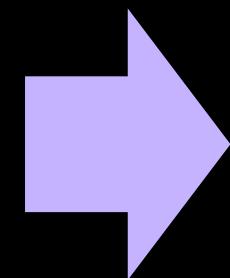


윤리적으로 크롤링하기

1. Robots.txt 파일 확인하기

- 로봇 배제 표준(Robots Exclusion Standard)으로도 잘 알려진 Robots.txt 파일은 웹 사이트에 로봇이 무단으로 접근하는 것을 방지하기 위한 프로토콜
- 크롤링하기 전에 웹 사이트의 URL 뒤에 /robots.txt를 덧붙여 해당 URL에 접속하면, robots.txt 파일 내용 확인 가능

```
User-agent: *
Disallow: /
Allow : /$
```



[의미]

모든 크롤러가 사이트의 어떤 내용도 크롤링하는 것을 금지하되, 예외적으로 홈페이지(루트 페이지)에 대해서만 접근을 허용하도록 설정

<https://naver.com/robots.txt> 결과



윤리적으로 크롤링하기

2. 이용 약관 충분히 숙지하기

- 웹 사이트의 콘텐츠를 상업적으로 활용하거나 소유자 혹은 제작자의 동의 없이 무단으로 복제하는 경우, 저작권 침해로 간주될 수 있음
- 저작권이 있거나 개인적인 자료를 동의 없이 상업적인 목적으로 사용하지 않도록 주의
- 특정 정보만을 취사선택해 수집하더라도 법적 문제가 없는지 미리 확인

3. 크롤링하는 홈페이지 서버에 부하 주지 않기

- 웹 사이트에서 데이터를 추출하기 위해 한 번에 너무 많은 요청을 보내면 해당 웹 서버에 과부하 문제 발생
- 보통은 웹 사이트에서 반복적으로 너무 많은 요청이 들어올 경우 로봇으로 간주해 데이터 요청을 차단
- 웹 서버에 피해를 주지 않도록 웹 크롤링 작업 속도를 줄여 하나의 웹 페이지를 추출한 후, 다음 페이지를 크롤링하는 것을 권장



리스크가 많은 사람의 운전



사람의 운전은 사람의 실수(피로도, 방심 등)로 인해 사고가 발생



자율주행 자동차



자율주행 자동차는 운전자의 과실로 발생하는 교통사고를 줄여
운전자와 보행자의 안전을 높이고, 교통 약자들의 이동장벽을 제거하며,
교통 정체를 완화시키는 역할 등을 수행



자율주행 시스템

- 자율주행은 운전자가 아닌 자율주행 시스템이 상황을 인식하여 즉각적인 반응을 수행해야 하므로 주변 사물에 대한 높은 정확도의 인식율과 실시간 처리가 핵심
- 신속성과 정확성을 모두 만족해야 하므로 기술 구현의 난이도가 높음





객체 인식

- **객체 인식(Object Detection)**은 이미지나 비디오 안에서 여러 사물을 인식하고 위치를 파악하는 과정
- Python 코드를 사용하면 이미지 안에서 객체를 찾아 사물을 인식할 수 있음

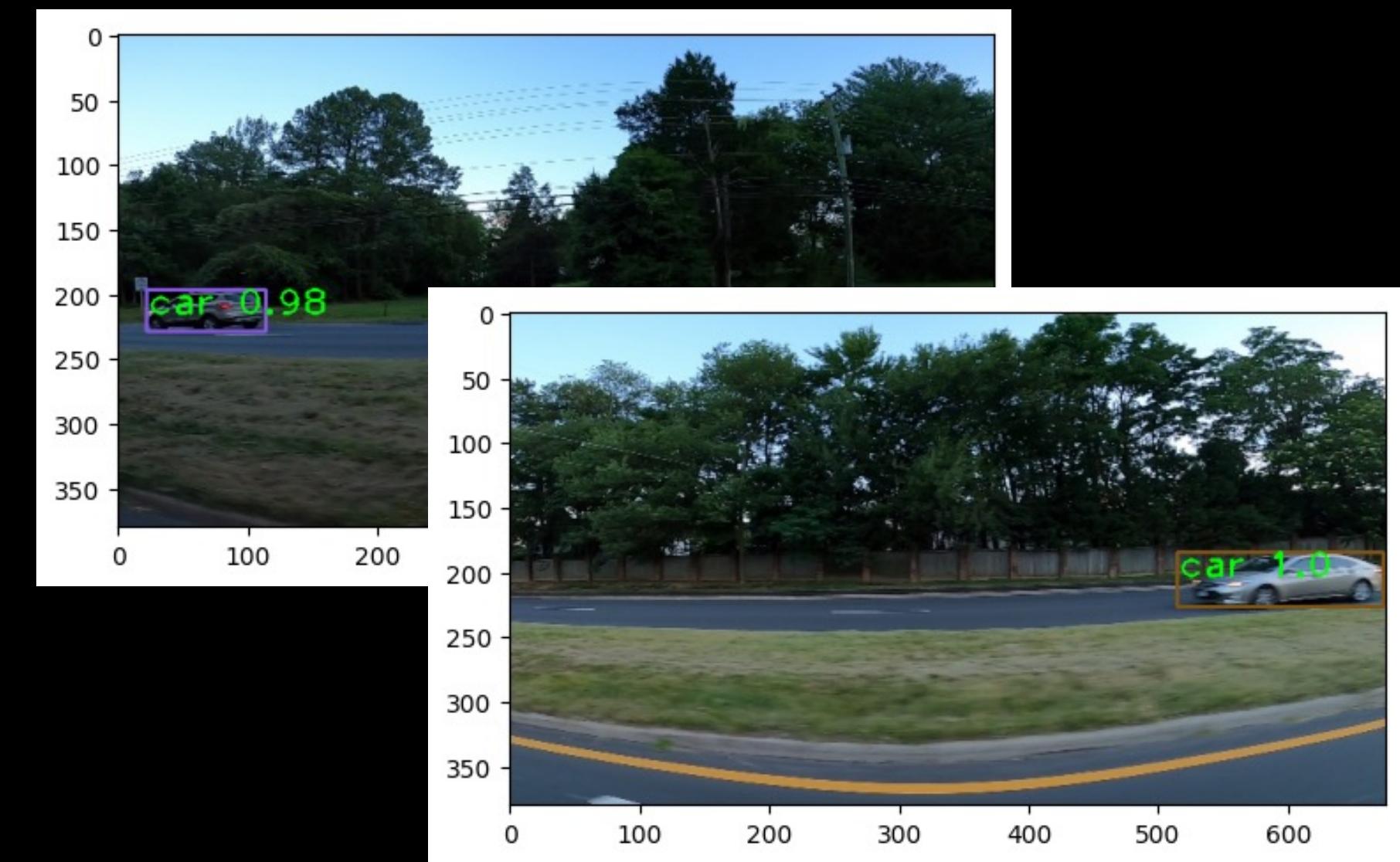
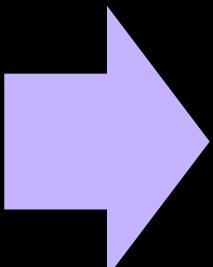
```
for i in indexes.flatten():
    x, y, w, h = boxes[i]

    label = str(classes[class_ids[i]])
    confidence = str(round(confidences[i], 2))
    color = colors[i]

    cv2.rectangle(img, (x, y), ((x+w), (y+h)), color, 2)

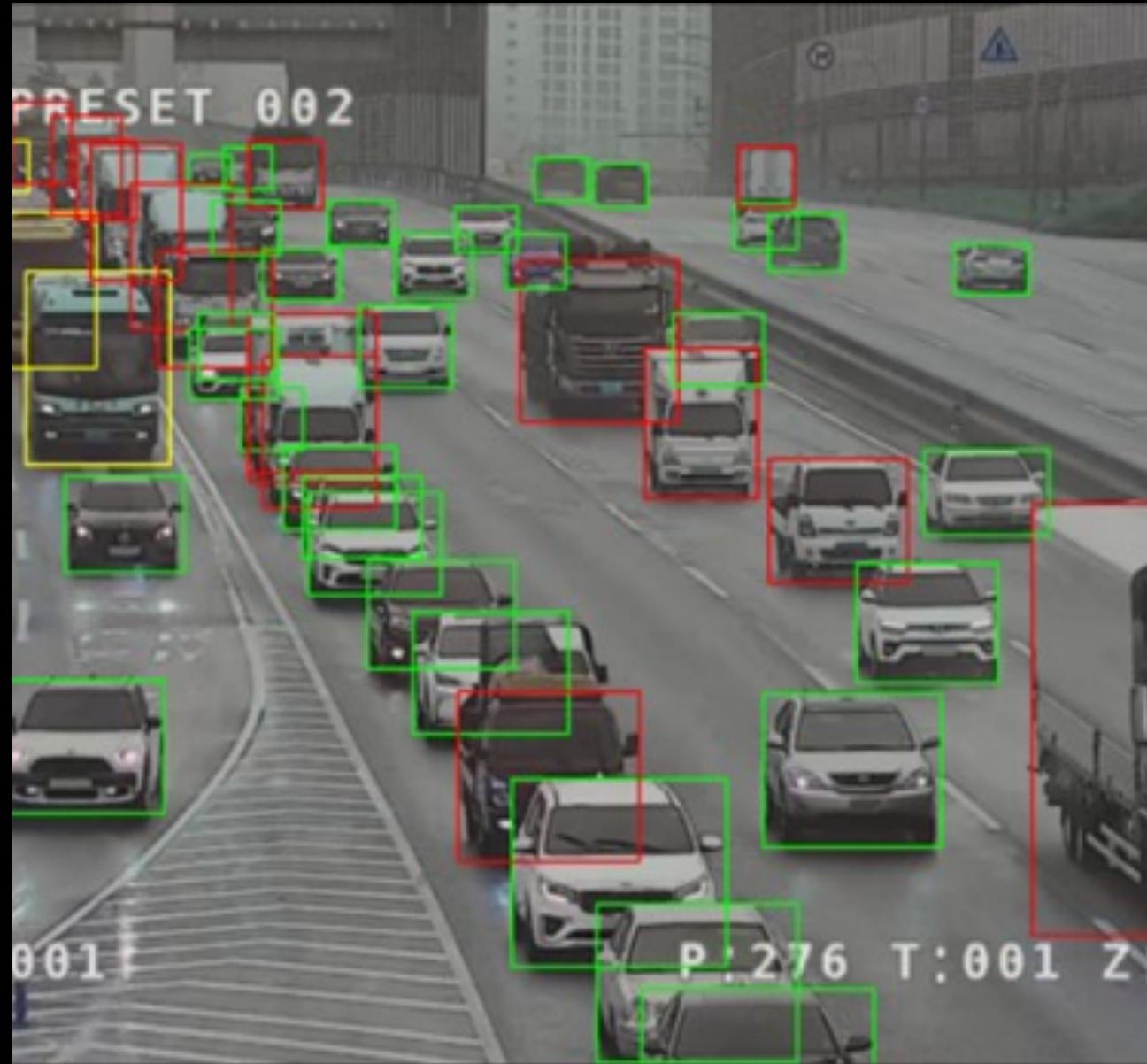
    cv2.putText(img, label + " " + confidence, (x, y+20), font, 2, (0, 255, 0), 2)

plt.imshow(img)
```





객체 인식 사용 사례



자율주행 자동차



CCTV를 이용한 화재 감지



드론을 이용한
작업자 안전모 착용 감지



제조 업계에서 발생하는 문제들

생산 설비에서
발생되는 막대한
데이터에 의한 데이터
분석과 처리의 복잡성

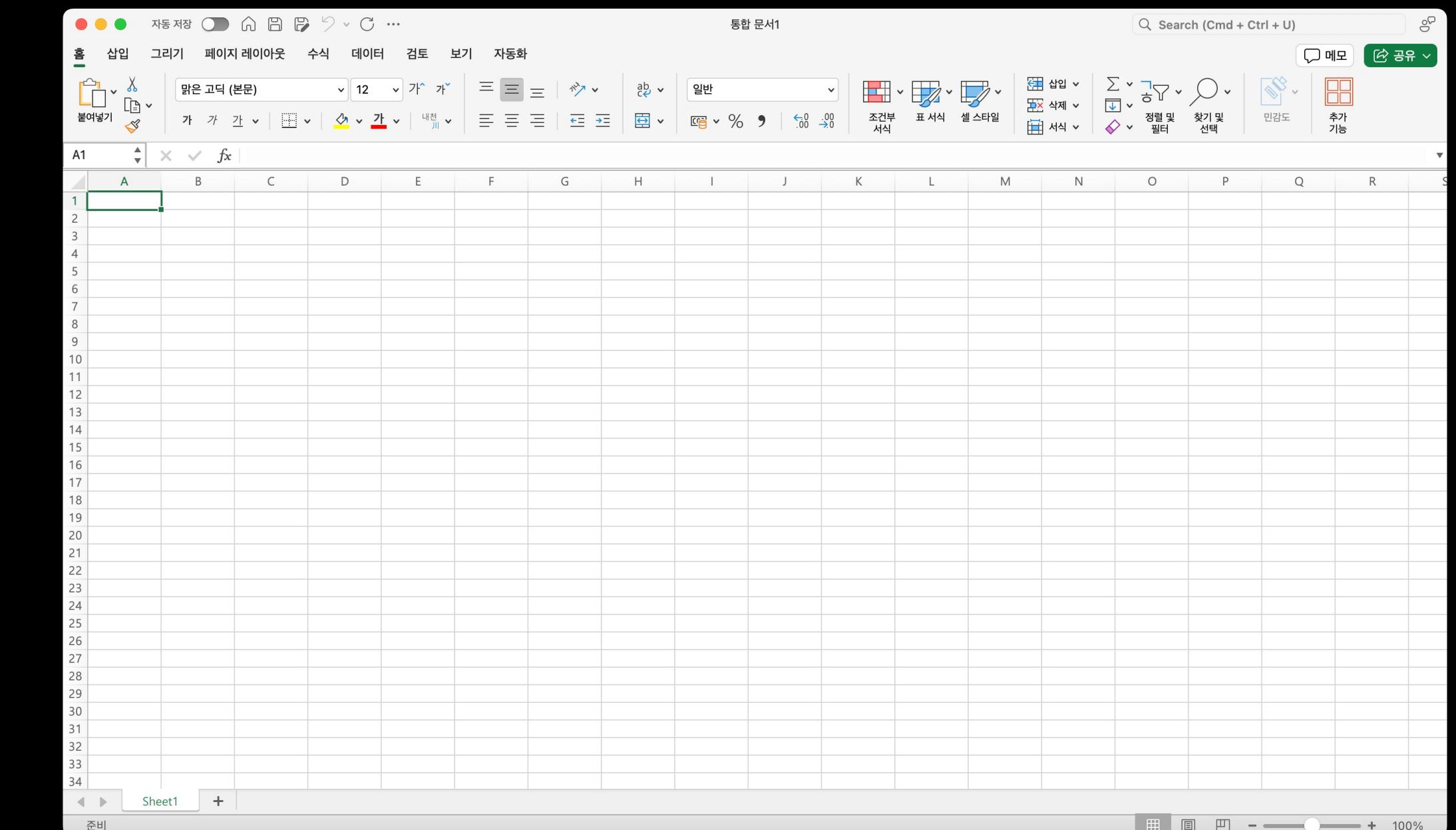
생산 공정 최적화의
어려움

물리적 한계에 따른
설계 난이도 상승



1. 막대한 데이터에 의한 데이터 분석과 처리의 복잡성

- 반도체 제조 과정은 초당 수많은 데이터를 생성
- 생성되는 데이터를 효과적으로 저장, 처리, 분석을 해야 하지만 생성되는 데이터는 종류가 다양하고 양이 많음
- 데이터 분석을 위해 주로 사용되는 스프레드 시트 프로그램(e.g., Excel)은 대용량 데이터 처리가 쉽지 않음
- Python을 이용하면 대용량 데이터 처리 및 분석이 가능하고 실시간으로 시각화를 할 수 있음





제조 공정 중에서 수집된 센서 데이터 분석 및 시각화

X0	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	
1	time_stamp	AmbientConc	AmbientConc	Machinet1	Machinet1	Machinet1	Machinet1	Machinet1	ZoMachine1	ZoMachine1	ZoMachine1	ZoMachine1	ZoMachine2												
2	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1241.26	72.3	48.03	10.48	436.76	76.3	75.1	12.59	236	200.75	69.37	69.9	73.25	13.89				
3																									
4	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1246.09	72	48.03	10.48	436.77	76.3	75.1	12.59	236	601.11	257	220.16	69.35	69.05	73.19	13.89		
5	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1246.29	72	48.16	10.48	425.46	76.3	75.1	12.59	236	601.11	257	216.8	69.37	69.05	73.19	13.85		
6	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1246.59	72	48.57	10.48	425.18	76.4	75.1	12.59	236	601.11	257	208.61	69.38	69.06	72.71	13.9		
7	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1252.83	72.1	48.57	10.48	425.18	76.4	75.1	12.59	236	601.11	257	213.31	69.07	69.07	73	13.89		
8	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1306.3	72.1	72.4	48.7	10.45	435.25	76.4	75	12.59	236	601.11	257	210.17	69.27	69.06	73.19	13.87	
9	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1306.13	72.1	72.4	49.78	10.48	439.68	76.4	75	12.59	236	601.11	257	210.21	69.25	68.94	73.19	13.93	
10																									
11	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1307.4	72.1	72.4	49.94	10.52	431.83	76.5	75	12.59	236	601.11	257	211.53	69.25	69.04	73.06	13.92	
12	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1308.57	72.1	72.4	49.97	10.52	431.88	76.5	75.1	12.59	236	601.11	257	212.26	69.05	69.05	73	13.89	
13	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1248.62	72.1	72.5	49.51	10.48	437.53	76.5	75.1	12.59	236	601.11	257	223.89	69.18	69.08	73.25	13.91	
14	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1252.53	72.1	72.5	49.51	10.48	434.45	76.6	75.1	12.59	236	601.11	257	215.14	69.16	69.06	72.81	13.86	
15	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1246.91	72.1	72.5	49.91	10.52	442.15	76.6	75.1	12.59	236	601.11	257	214.8	69.16	69.09	72.81	13.87	
16	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1191.22	72.1	72.5	49.64	10.48	431.37	76.6	75.1	12.59	236	601.11	257	224.29	69.16	69.2	73.06	13.92	
17	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1103.3	72.1	72.5	50.18	10.52	423.65	76.6	75	12.59	236	601.11	257	218.9	69.09	69.25	73.13	13.93	
18	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1179.7	72.1	72.5	49.64	10.52	431.54	76.6	75.1	12.59	236	601.11	257	213.6	69.07	69.41	72.94	13.91	
19	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1239.59	72.1	72.5	49.34	10.52	436.02	76.7	75.1	12.59	236	601.11	257	237.5	69.09	69.4	73.1	13.89	
20	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1241.96	72.1	72.5	48.97	10.48	435.75	76.7	75.1	12.59	236	601.11	257	220.19	69.09	69.59	73.19	13.91	
21	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1242.83	72.1	72.5	49.24	10.48	422.7	76.7	75.1	12.59	236	601.11	257	220.45	69.09	69.62	72.88	13.92	
22	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1241.70805	72.1001093	72.55	49.6843794	10.481546	429.067694	76.7499999	75.0246479	12.59	236	601.11	257	218.969247	69.027002	69.656147	72.9400961	13.9244395	
23	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1242.29	72.1	72.6	49.91	10.48	436.11	76.8	75	12.59	236	601.11	257	215.89	69.98	69.7	73	13.91	
24	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1242.72	72.1	72.6	49.64	10.48	436.24	76.8	75.1	12.59	236	601.11	257	210.85	69.72	69.72	72.99	13.87	
25	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1244.21	72.1	72.6	49.24	10.48	435.52	76.8	75	12.59	236	601.11	257	223.86	68.97	69.7	73	13.86	
26	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1246.2	72.1	72.6	49.51	10.48	435.76	76.8	75	12.59	236	601.11	257	215.15	68.99	69.72	72.94	13.89	
27	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1245.59	72.2	72.6	50.32	10.52	440.6	76.9	75	12.59	236	601.11	257	214.27	68.97	69.69	72.81	13.93	
28	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1247.52	72.2	72.6	50.18	10.48	424.61	76.9	75	12.59	236	601.11	257	200.22	69.01	69.71	72.69	13.93	
29	2019.3.6 10:52	17.24	23.53	11.54	200	963	247	1248.21	72.2	72.6	50.32	10.52	431.41	76.9	75	12.59	236	601.11	257	219.22	68.99	69.79	72.81	13.91	
30	2019.3.6 10:53	17.24	23.53	11.54	200	963	247	1248.77	72.2	72.6	49.24	10.52	431.4	76.9	75.1	12.59	236	601.11	257	206.42</					



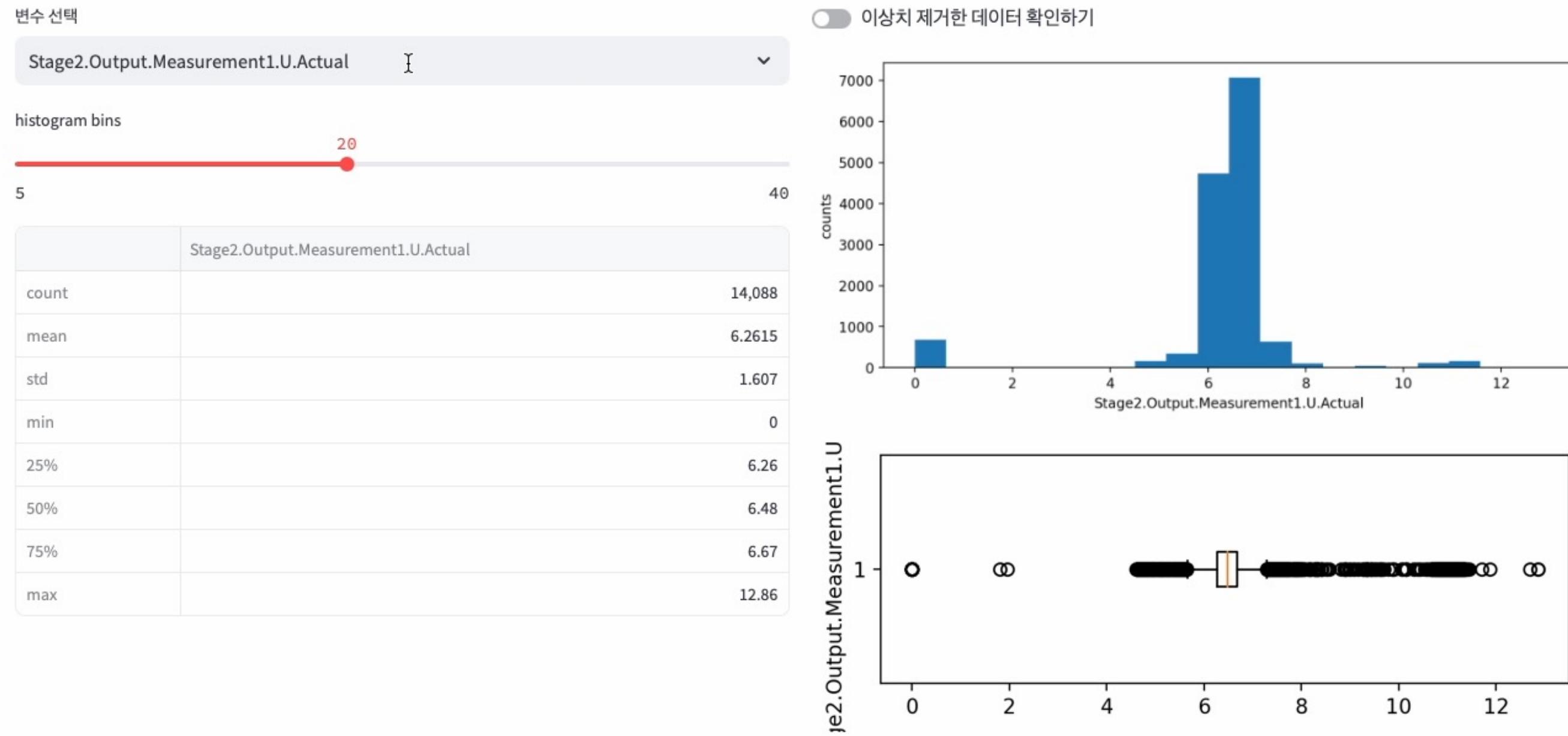
제조 공정 중에서 수집된 센서 데이터 분석 및 시각화

2.3 "데이터 분포"를 확인하고 이를 시각화하여 확인하기

상관관계를 분석하기 위해서는 통계적인 개념이 필요하며 현재 데이터가 방대하여 분석하기가 어렵습니다.

데이터 분포를 확인하기 위해 [히스토그램\(Histogram\)](#) 과 [박스플롯\(boxplot\)](#) 을 이용하여 확인해 보겠습니다.

- [히스토그램\(Histogram\)](#) 을 이용하면 데이터의 분포를 쉽게 확인할 수 있습니다. 아래의 슬라이더를 조절하면 막대의 개수를 조절할 수 있습니다.
- [박스플롯\(boxplot\)](#) 을 이용하면 데이터의 간단한 통계 정보와 함께 이상치를 확인할 수 있습니다. 아래의 표는 선택한 변수의 간단한 통계 정보를 확인할 수 있습니다.
 - 이상치를 제거한 데이터의 그래프를 보고 싶으면 상단의 체크박스를 클릭합니다.



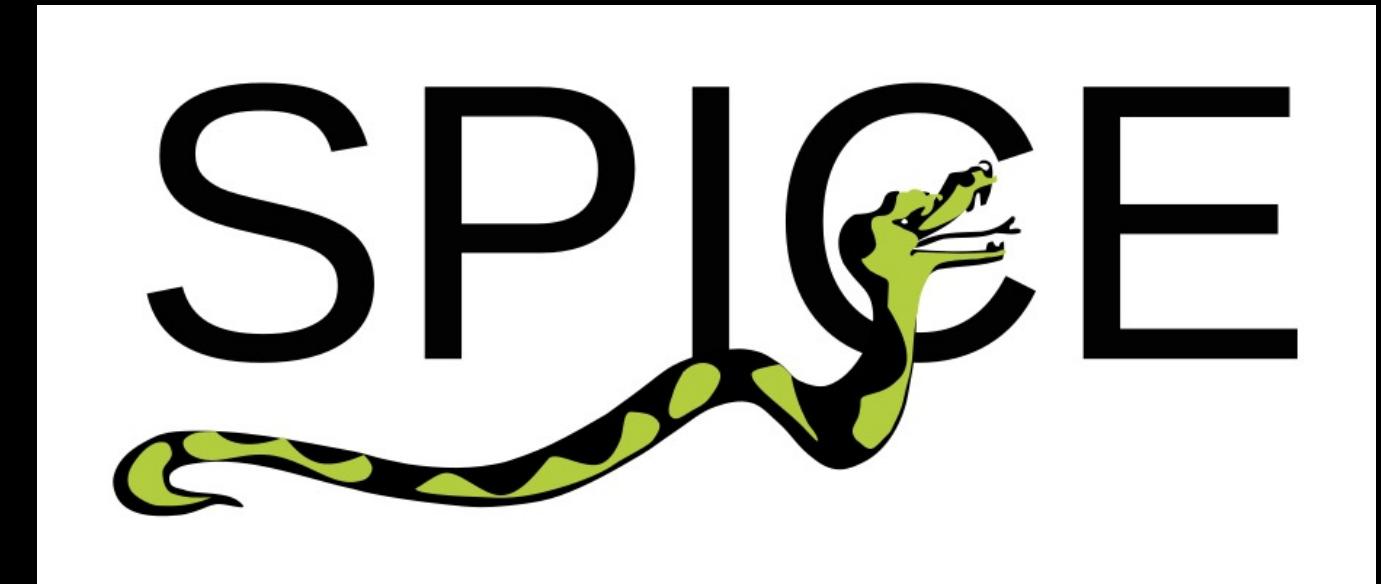


2. 생산 공정 최적화의 어려움

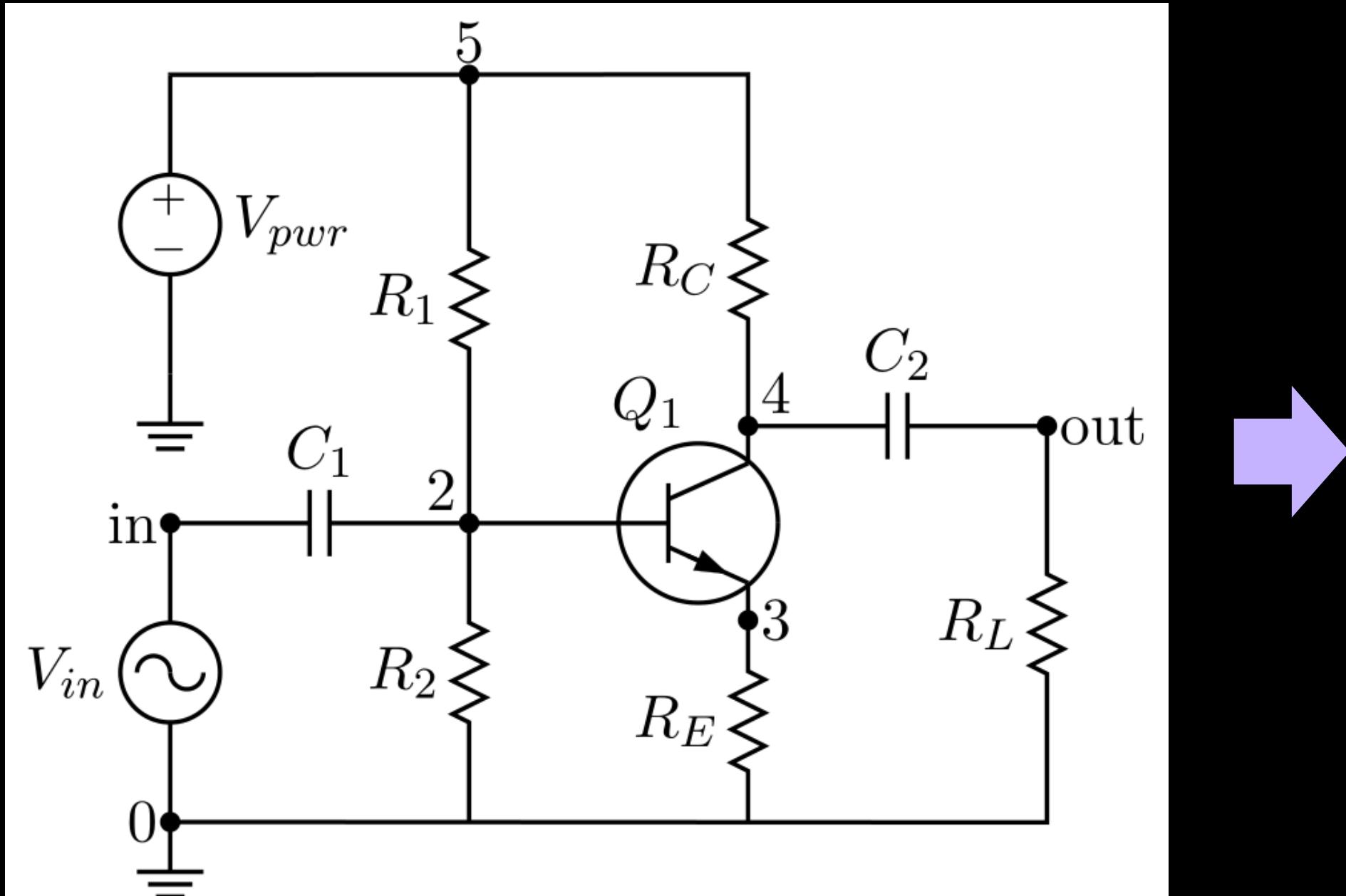
- 반도체 공정은 매우 복잡하며 공정의 효율성과 제품 품질에 직접적인 영향을 미침
- 각 단계에서 엄격한 품질 요구사항 적용
- 최적화를 위해서는 제품 설계, 제조 공정의 최적화가 필요
- Python을 이용하면 전자 회로 설계 및 시뮬레이션 작업이 가능하며 인공지능을 도입함으로써 제조 공정 개선 가능

Python 기반의 오픈소스 라이브러리를 이용한 전자 회로 설계 및 시뮬레이션 작업

- PySpice는 Python 환경에서 전자 회로 시뮬레이션을 위해 사용되는 라이브러리
- SPICE(Simulation Program with Integrated Circuit Emphasis) 시뮬레이터 기반
- Python의 편리함과 SPICE의 강력한 시뮬레이션 기능을 결합하여, 전자 회로의 설계 및 분석을 위한 효과적인 도구를 제공



Python 기반의 오픈소스 라이브러리를 이용한 전자 회로 설계 및 시뮬레이션 작업



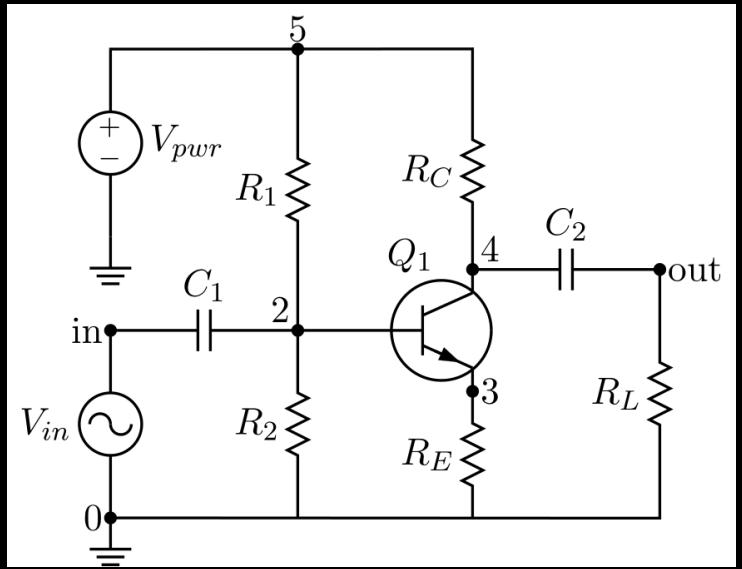
```
circuit = Circuit('Transistor')

circuit.V('power', 5, circuit.gnd, 15@u_V)
source = circuit.SinusoidalVoltageSource('in', 'in', circuit.gnd, amplitude=.5@u_V, frequency=1@u_kHz)
circuit.C(1, 'in', 2, 10@u_uF)
circuit.R(1, 5, 2, 100@u_kΩ)
circuit.R(2, 2, 0, 20@u_kΩ)
circuit.R('C', 5, 4, 10@u_kΩ)
circuit.BJT(1, 4, 2, 3, model='bjt') # Q is mapped to BJT !
circuit.model('bjt', 'npn', bf=80, cjc=pico(5), rb=100)
circuit.R('E', 3, 0, 2@u_kΩ)
circuit.C(2, 4, 'out', 10@u_uF)
circuit.R('Load', 'out', 0, 1@u_MΩ)
```

Python과 PySpice으로 구현한

“트랜지스터를 사용한 기본적인 증폭 회로”

Python 기반의 오픈소스 라이브러리를 이용한 전자 회로 설계 및 시뮬레이션 작업

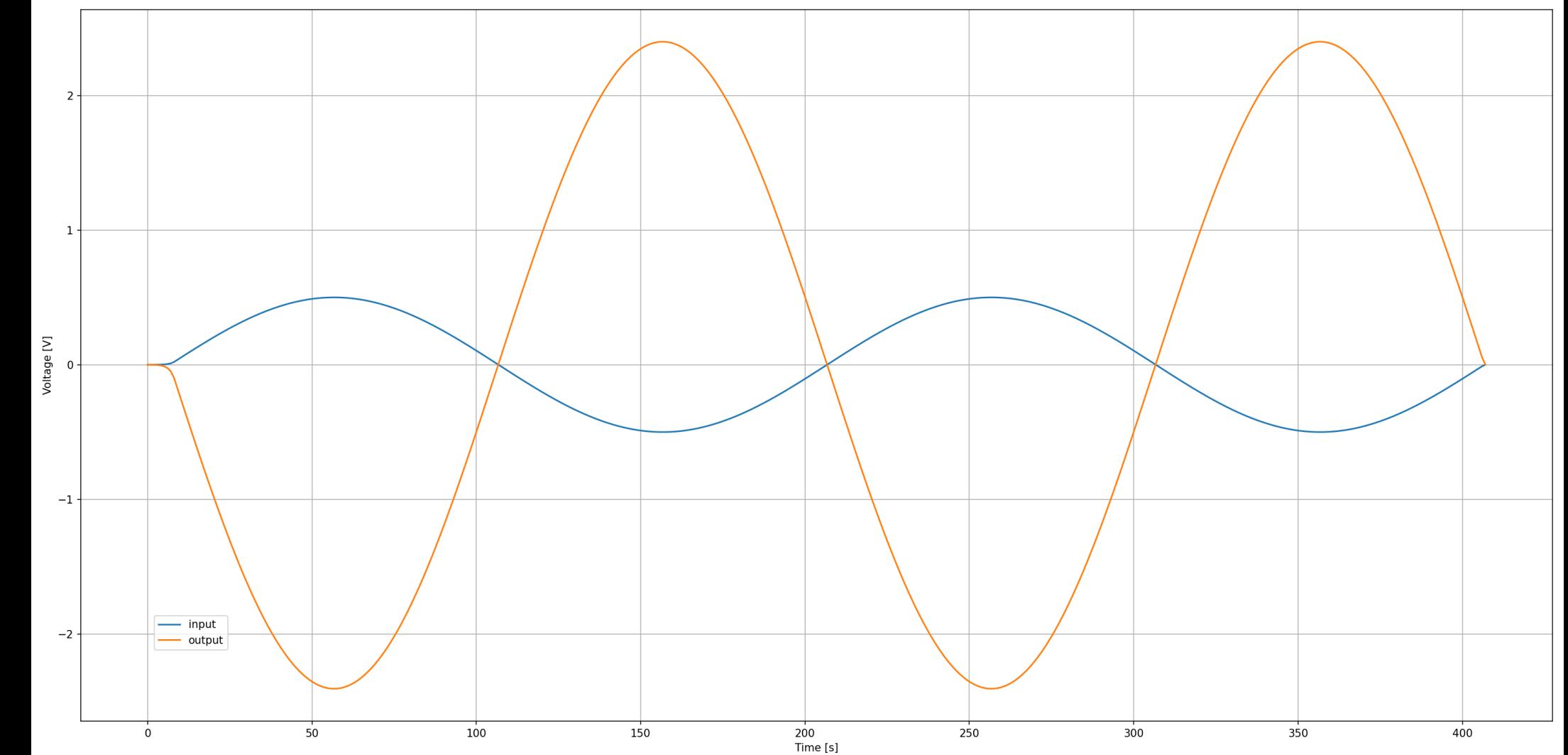
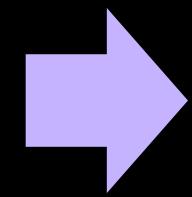


```
figure, ax = plt.subplots(figsize=(20, 10))

simulator = circuit.simulator(temperature=25, nominal_temperature=25)
analysis = simulator.transient(step_time=source.period/200, end_time=source.period*2)

ax.set_title('')
ax.set_xlabel('Time [s]')
ax.set_ylabel('Voltage [V]')
ax.grid()
ax.plot(analysis['in'])
ax.plot(analysis.out)
ax.legend(['input', 'output'], loc=.05,.1))

plt.tight_layout()
```



Python과 PySpice으로 구현한

“트랜지스터를 사용한 기본적인 증폭 회로” 시뮬레이션 결과

☰ 3. 물리적 한계에 따른 설계 난이도 상승

- 최근 몇 년간 반도체 기술은 빠르게 발전하고 있음
- 나노 단위의 설계, 트랜지스터 집약, 효율성 고려 등의 이유로 인해 반도체 설계의 난이도가 기하급수적으로 상승하고 있음
- Python과 인공지능을 기술을 도입함으로써 효과적인 반도체 설계 가능
- 최근에는 생성AI를 도입함으로써 자연어 질의를 통해 반도체 설계를 최적화 사례가 있음



Google의 반도체 설계 사례

- TPU (Tensor Processor Unit) 설계 중

Placement & Routing (P&R) 문제에 기계학습 알고리즘 사용

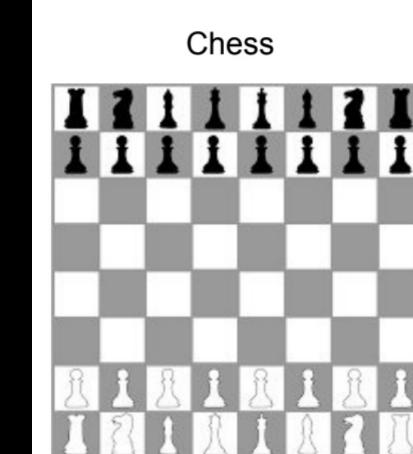
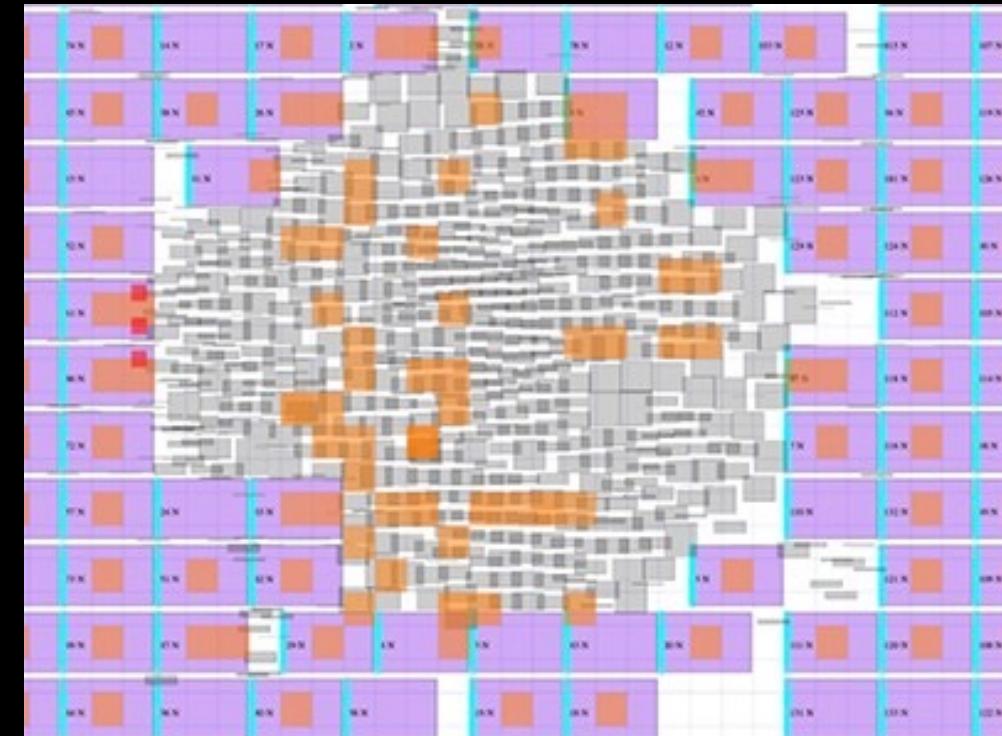
- 평면 배치 (floorplanning)

- 칩 안에 수백만 개의 반도체 소자와 부품을 효율적으로 배치하는 과정

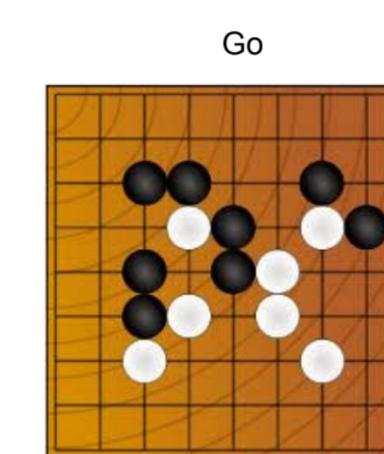
- 반도체 제조는 대부분 자동화되어 있지만,

여전히 설계는 수동 프로세스에 의존

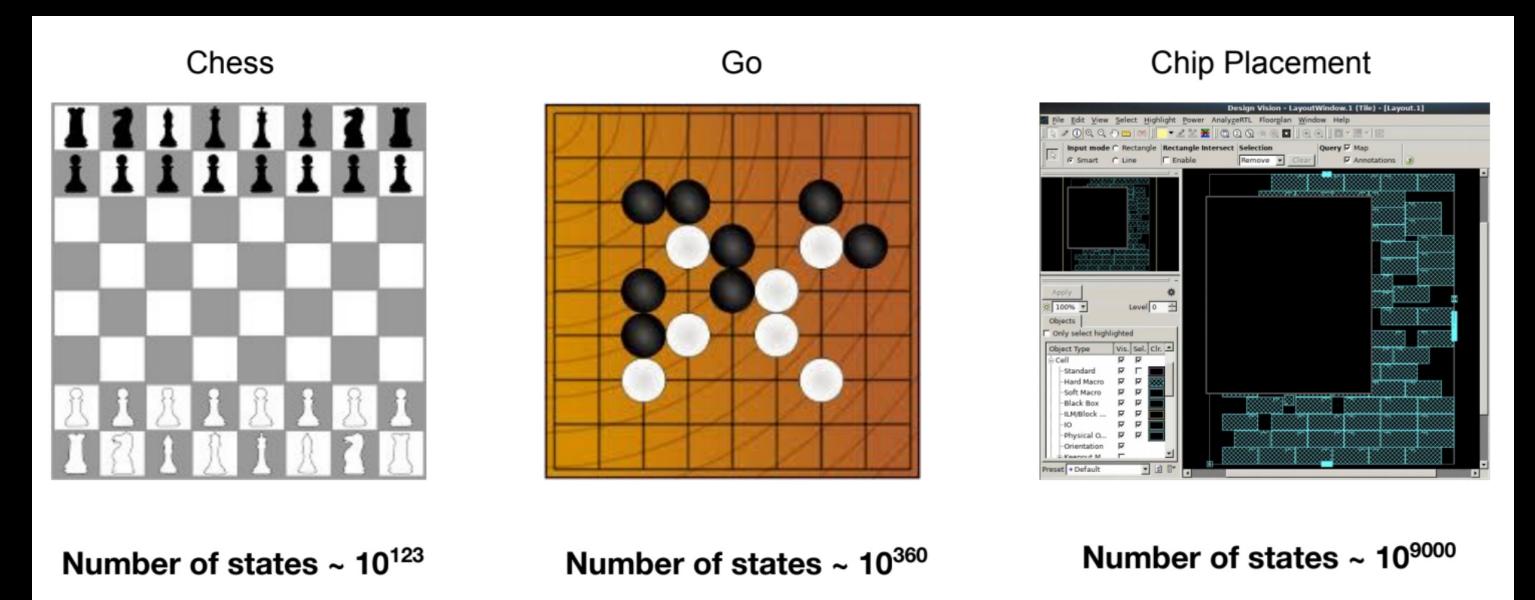
- 설계 시 부품과 이를 연결하는 전선을 효율적으로 배치하기 위해서 수개월의 시간이 필요



Number of states ~ 10^{123}



Number of states ~ 10^{360}



Number of states ~ 10^{9000}



Google의 반도체 설계 사례

Article | Published: 09 June 2021

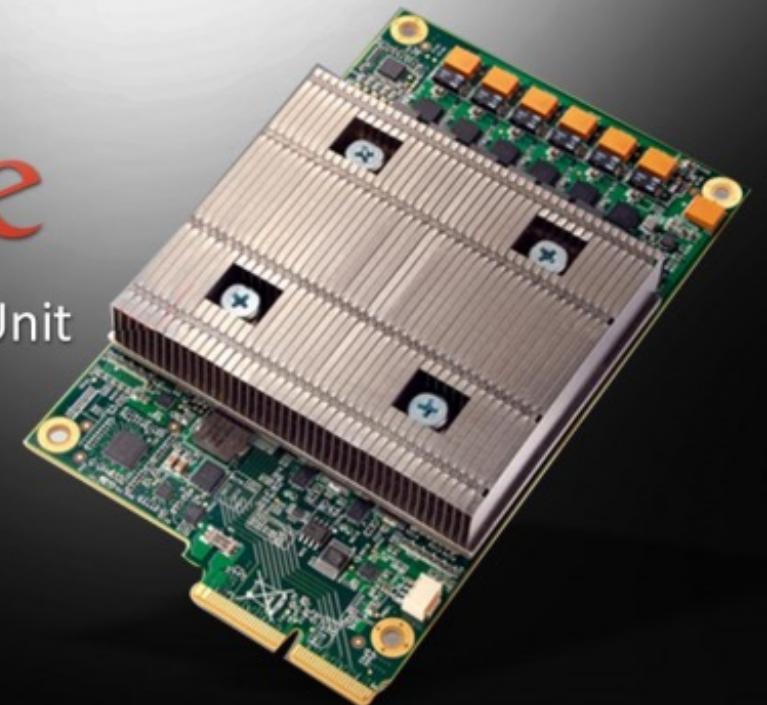
A graph placement methodology for fast chip design

[Azalia Mirhoseini](#) [Anna Goldie](#) [Mustafa Yazgan](#), [Joe Wenjie Jiang](#), [Ebrahim Songhori](#), [Shen Wang](#), [Young-Joon Lee](#), [Eric Johnson](#), [Omkar Pathak](#), [Azade Nazi](#), [Jiwoo Pak](#), [Andy Tong](#), [Kavya Srinivasa](#), [William Hang](#), [Emre Tuncer](#), [Quoc V. Le](#), [James Laudon](#), [Richard Ho](#), [Roger Carpenter](#) & [Jeff Dean](#)

[Nature](#) 594, 207–212 (2021) | [Cite this article](#)

36k Accesses | 38 Citations | 2064 Altmetric | [Metrics](#)

Google
Tensor Processing Unit

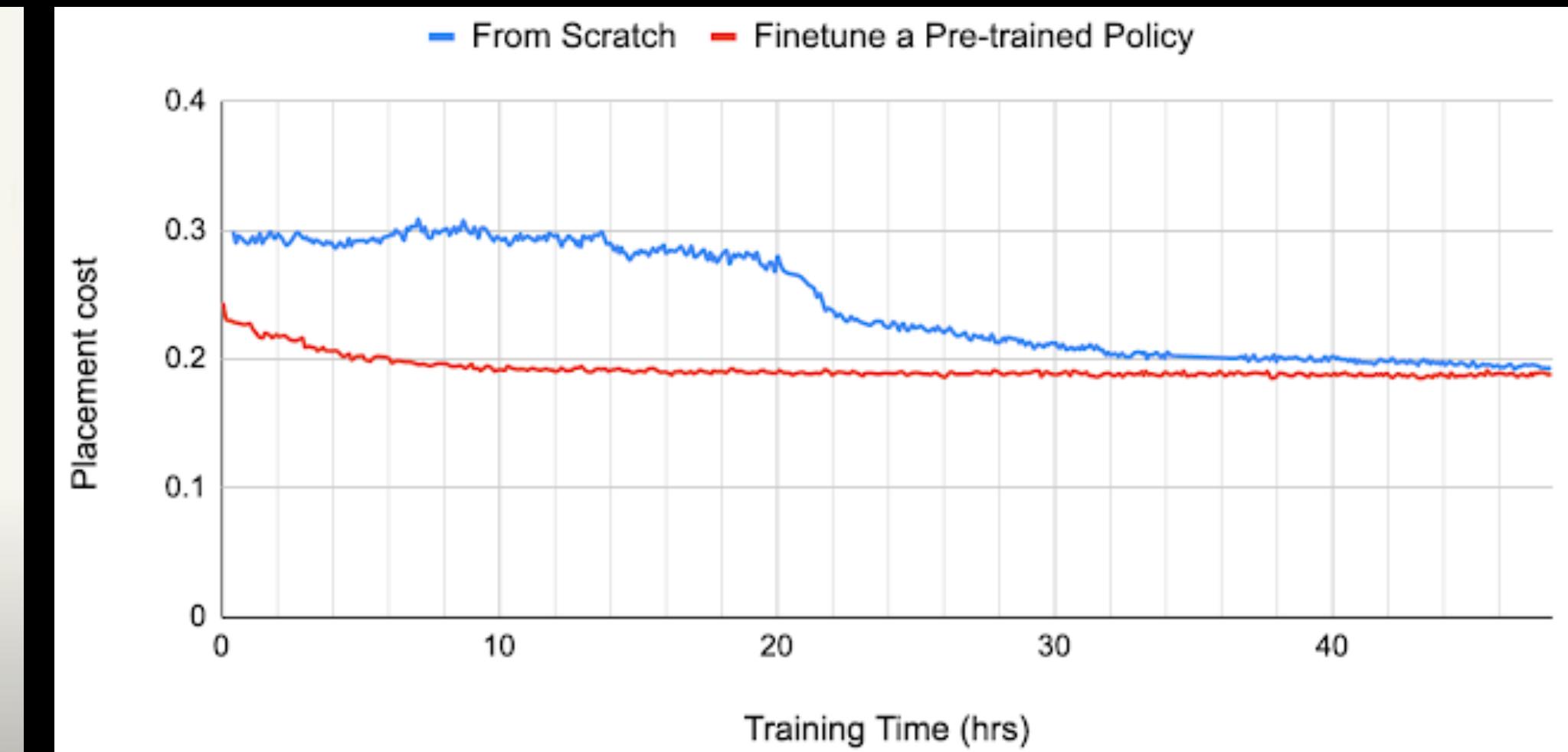
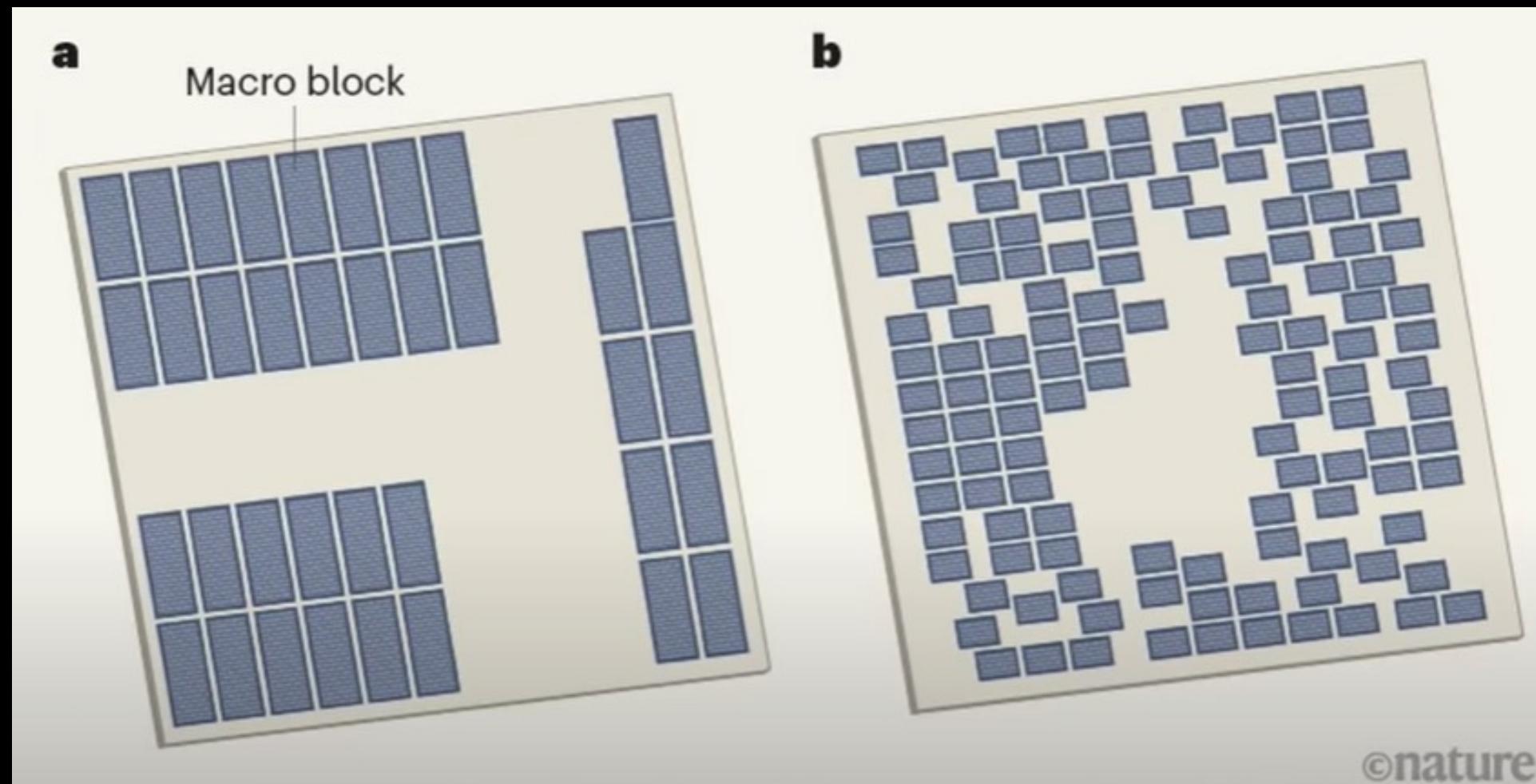


2021년, 구글에서는 Chip Placement 문제에 AI를 적용한 연구 결과 발표
해당 논문에서는 인간이 수개월 걸리던 해당 문제 해결을 AI를 적용해 단 6시간만에 완료했다고 발표



Google의 반도체 설계 사례

- 전문가가 수주에 걸쳐 완성한 배치 결과보다 수치적으로 우수
 - 전력 소모량, 퍼포먼스 및 칩 면적 포함
- 단, 사람에 의해 만들어진 칩 레이아웃들을 기반으로 훈련했을 때 훨씬 좋은 결과를 얻을 수 있었음





NVIDIA의 LLM을 이용한 반도체 설계 최적화

- NVIDIA는 30년 동안 수집한 데이터를 이용하여 반도체 설계 노하우를 갖춘 챗봇(ChipNeMo) 공개
- 생성 AI를 칩 설계의 모든 단계에 적용해 전반적인 생산성을 향상할 목적으로 개발
- 칩 설계용 전문 언어 '베릴로그나' 'VHDL'을 통해 최대 20줄에 달하는 소프트웨어 코드를 한번에 생성할 수 있고 반도체 제작에 필요한 데이터 분석 가능

The screenshot shows a chat interface titled "ChipNeMo-Python". The input field says "Enter your command/question to start a new conversation...". Below it, a message from "ChipNeMo-Python [chipnemo_43b_chat_delta]" asks for a Python code to print flop cells in a rectangle. The code completion window displays the following Python code:

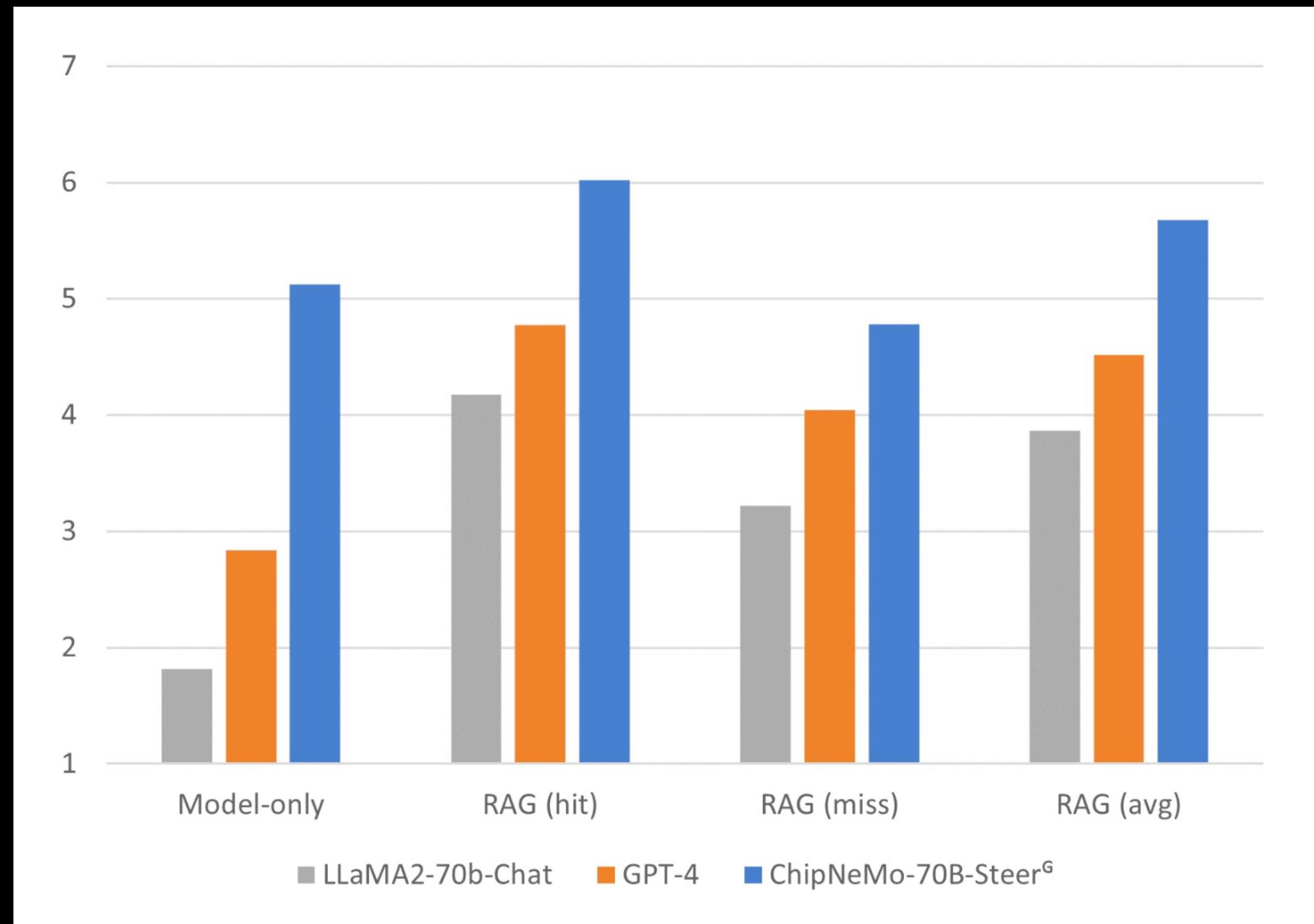
```
# Sure, here is a code to print the number of flop cells in the given rectangle (0,0,100,100):
# This code gets the top_ref of the design
top_ref = env().design()
# This code gets the cells of the rect in the top_ref
cells = env().physical().cells_in_rect(top_ref, rectangle(0, 0, 100, 100), hierarchical)
# This code gets the cells that are flop cells
flop_cells = cells.refine("flop")
# This code prints the number of flop cells
print(flop_cells.count())
```

At the bottom left is a blue "Exec Code" button. At the bottom right are three small icons: a square, a triangle, and a circle.



NVIDIA의 LLM을 이용한 반도체 설계 최적화

엔지니어링 보조 및 EDA(Engineering Design Automation) 스크립트 생성 사례에서
OpenAI의 GPT-4를 능가하는 성능을 보임



Human Evaluation of Different Models. Model Only represents results without RAG. RAG (hit)/(miss) only include questions whose retrieved passages hit/miss their ideal context, RAG (avg) includes all questions. 7 point Likert scale.



NVIDIA의 LLM을 이용한 반도체 설계 최적화

- **반도체 설계 과정의 자동화** : ChipNeMo는 반도체 설계 과정의 많은 부분을 자동화하여 설계 시간을 단축하고 설계자의 생산성을 향상시킬 것으로 기대
- **설계 오류 감소** : ChipNeMo는 코드 검증 기능을 통해 설계 오류를 감소시키고 설계의 정확성 향상 기대
- **설계 최적화** : 설계 최적화 기능을 통해 성능, 면적, 전력 소비 등을 최적화하여 더 나은 반도체를 설계할 것으로 기대