

Day 11 - Python Module

파일 종류 소개:

.py	Python 파일	일반적인 파이썬 코드 파일 (텍스트 기반)
.ipynb	IPython Notebook 파일	Jupyter에서 사용하는 노트북 형식 파일 (JSON 기반)

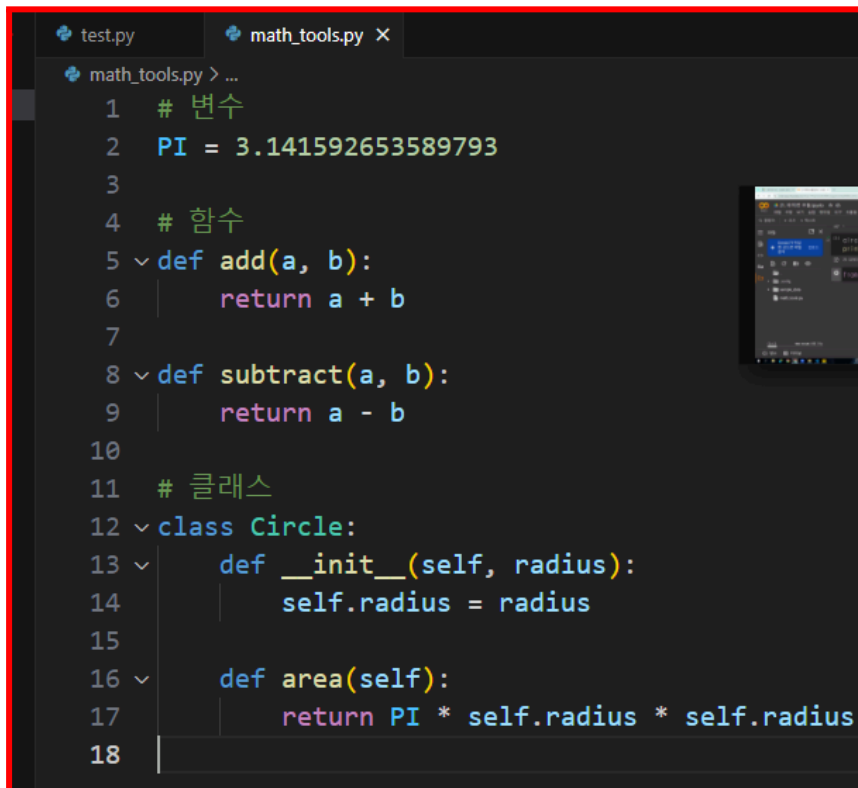
항목	.py 파일	.ipynb 파일
사용 환경	모든 텍스트 에디터, IDE (VS Code, PyCharm 등)	Jupyter Notebook, JupyterLab 등에서 실행
파일 형식	텍스트 기반 (.txt 와 유사)	JSON 기반 (코드, 출력, 마크다운 등 포함)
실행 단위	전체 스크립트를 한 번에 실행	셀 단위로 나눠서 실행 가능
결과 저장	결과는 별도로 저장하지 않음	셀의 출력 결과가 파일 안에 함께 저장됨
시각화 지원	외부 창에서 그림 출력	그림, 그래프를 셀 아래에 바로 보여줌
주 용도	스크립트, 프로그램, 배포용 코드 작성	실험, 분석, 설명이 필요한 코드 작성 및 교육용
마크다운 지원	주석(##)만 가능	마크다운 셀로 문서화 가능 (설명 + 코드 혼합)

프로젝트 개발, 배포용 코드 작성	.py 파일
데이터 분석, 실험 기록, 강의용	.ipynb 파일

Visual studio 다운:

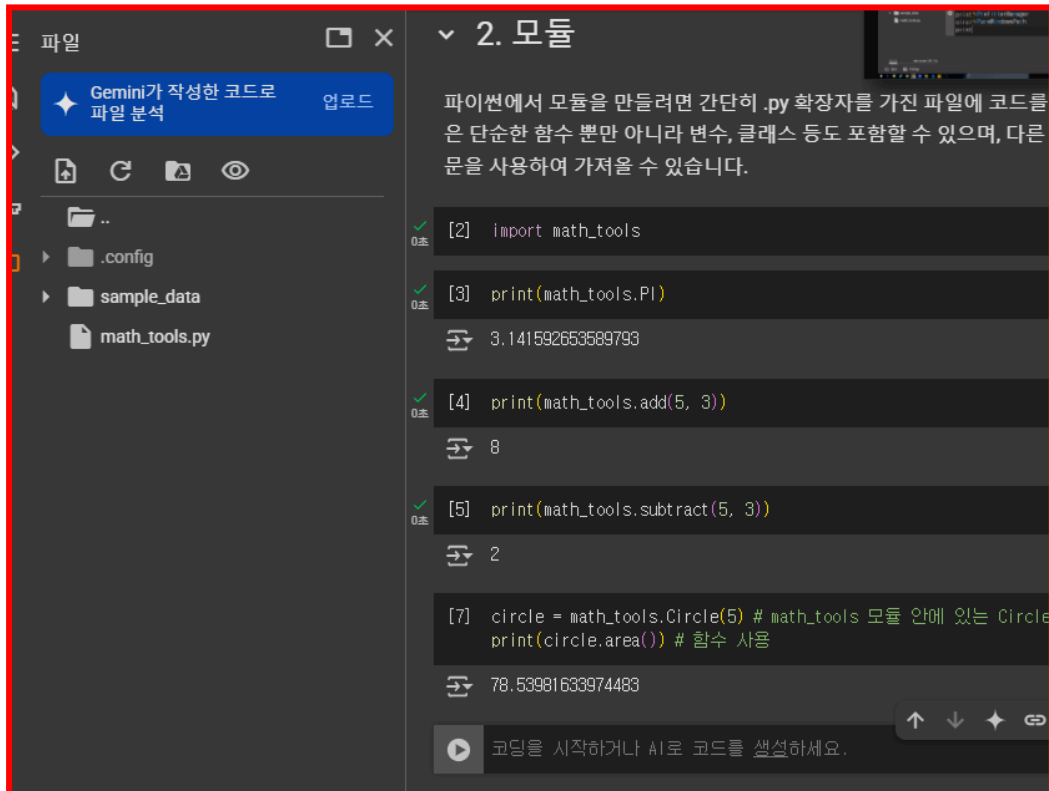
<https://code.visualstudio.com/>

모듈(Module):



```
1 # 변수
2 PI = 3.141592653589793
3
4 # 함수
5 def add(a, b):
6     return a + b
7
8 def subtract(a, b):
9     return a - b
10
11 # 클래스
12 class Circle:
13     def __init__(self, radius):
14         self.radius = radius
15
16     def area(self):
17         return PI * self.radius * self.radius
18
```

- math_tools.py 작성



파일

Gemini가 작성한 코드로
파일 분석 업로드

2. 모듈

파이썬에서 모듈을 만들려면 간단히 .py 확장자를 가진 파일에 코드를
은 단순한 함수 뿐만 아니라 변수, 클래스 등도 포함할 수 있으며, 다른
문을 사용하여 가져올 수 있습니다.

```
[2] import math_tools
[3] print(math_tools.PI)
3.141592653589793
[4] print(math_tools.add(5, 3))
8
[5] print(math_tools.subtract(5, 3))
2
[7] circle = math_tools.Circle(5) # math_tools 모듈 안에 있는 Circle
print(circle.area()) # 함수 사용
78.53981633974483
```

코딩을 시작하거나 시로 코드를 생성하세요.

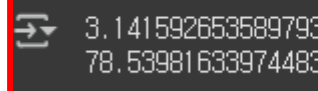
- Collab 파일에 업로드 후 모듈로 사용가능
- 클래스, 변수 등 모듈에 있는것들 이용가능

다양한 import 방법들:

```
[11] from math_tools import PI, Circle

[12] print(PI) # 3.141592653589793

      circle = Circle(5)
      print(circle.area()) # 78.53981633974483
```



```
↔ 3.141592653589793
   78.53981633974483
```

```
▶ import math_tools as mt
```

- from [모듈] import [Class, var]
- import [모듈] as [닉네임]
 - 이런식으로도 임포트 가능

구글 드라이브 파일 연동:

```
[19] path = '/content/drive/MyDrive/Python_Computer Vision With AI/2. Python Web Development and LLM Basics/module'

      import sys
      sys.path.append(path) # 시스템에 경로 추가
```

```
[20] import math_tools as mt
```

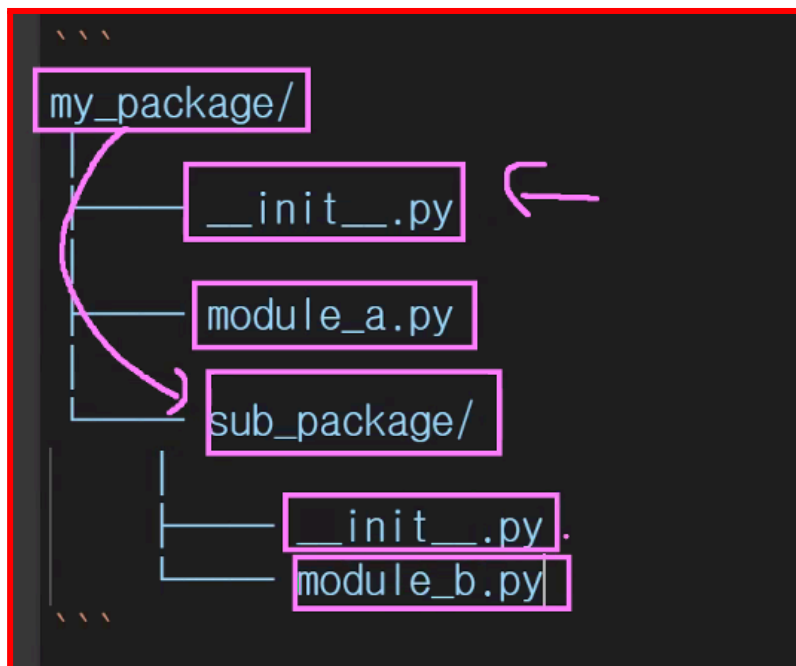
- 구글 드라이브에 모듈을 넣고 사용

`__name__` 속성

```
7 |         return PI * self.radius * self.r
8 |
9 | if __name__ == '__main__':
10 |     print('이 모듈은 직접 실행되었습니다')
11 | else:
12 |     print('이 모듈은 import 되었습니다')
```

- 그냥 실행시키면 `name`에 `main` 값이 들어간다.
- import되면, `__name__`은 원래의 모듈 이름으로 설정된다.
 - 모듈로 사용했을때, 그냥 파일을 사용했을때 를 나눠서 프로그램을 짤 수 있다.

패키지(Package)



- 모듈들을 포함하고 있는 디렉터리
- 패키지를 사용하면 관련된 기능들을 함께 묶어서 코드를 더욱 체계적으로 관리할 수 있습니다.

```
from my_package import module_a
from my_package.sub_package import module_b
```

코딩을 시작하거나 AI로 코드를 생성하세요.

- 또한 패키지 안에 다른 패키지(하위 패키지)를 포함할 수 있습니다. 이를 통해 복잡한 프로젝트의 코드를 여러 레벨의 디렉터리로 계층적으로 구성할 수 있습니다.

패키지 예시:

```
shapes/
|
├─ __init__.py
├─ circle.py
└─ rectangle.py
```

패키지: __init__.py

```
# shapes/__init__.py

__all__ = ["circle", "rectangle"]
```

- from shapes import *를 사용할 때 circle와 rectangle 모듈만 가져온다는 것을 의미.

```
[21] from shapes import circle, rectangle # shape 패키지 안에 있는 circle, rectangle 모듈 импорт
```

```
[24] print(circle.area(5)) # 각 모듈안에 있는 함수 사용 가능
      print(circle.circumference(5))
```

```
78.53981633974483
31.41592653589793
```

- 패키지를 구글 드라이브에 업로드 후 패키지 안의 각 모듈들이 잘 작동한다.

```
✓  
0초 [26] from shapes.circle import area as circle_area  
      from shapes.rectangle import area as rectangle_area  
  
✓  
0초 [27] print(circle_area(5))  
      print(rectangle_area(4, 6))  
  
⇒ 78.53981633974483  
   24
```

- 이렇게 이름이 겹칠때는 별명을 다르게 넣어서 방지할 수 있다.