

# 으뜸 파이썬

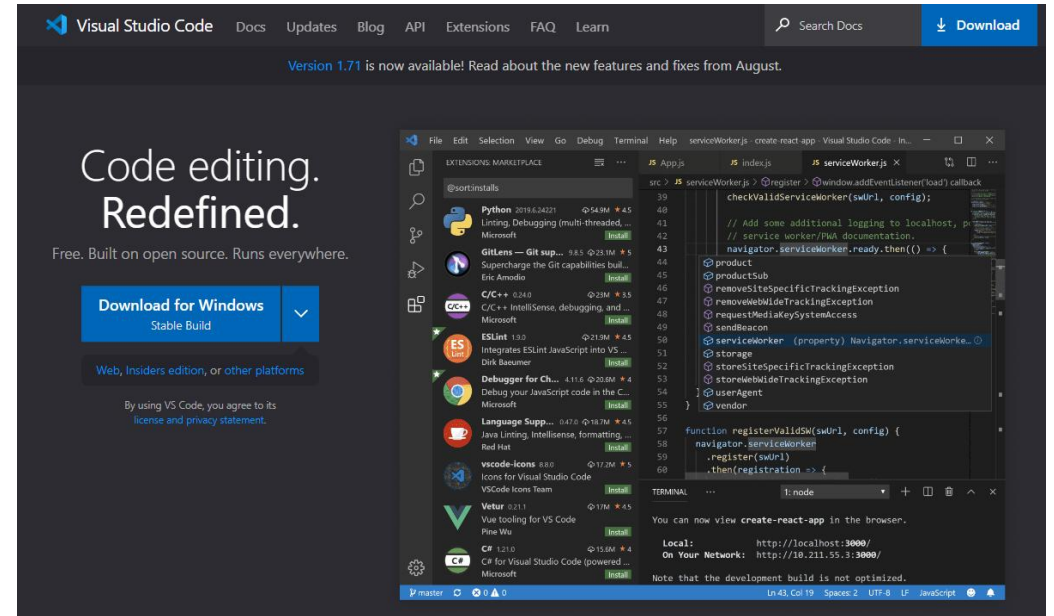


## 7강 IDE

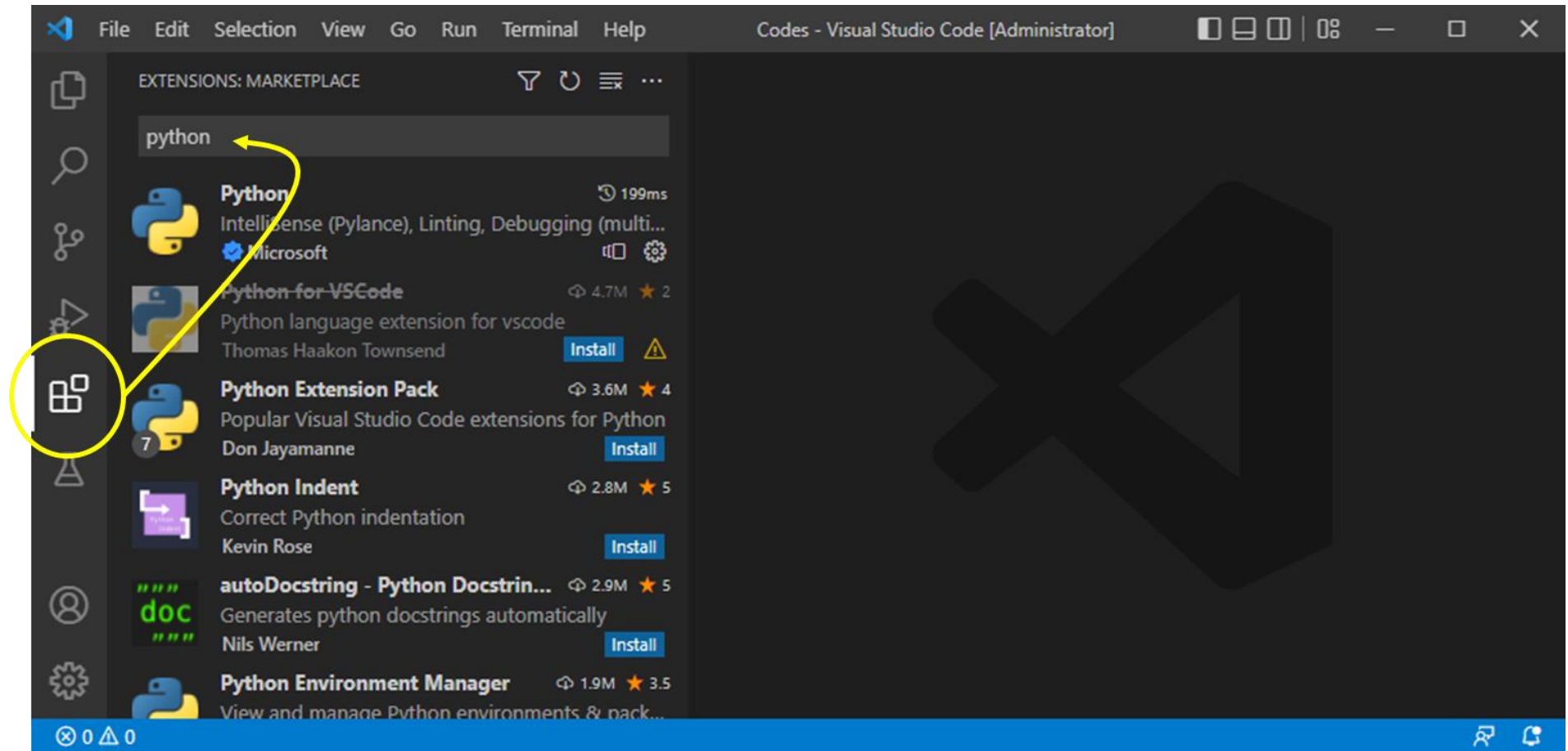
- 본 강의노트는 으뜸 파이썬(박동규, 강영민 著) 1판의 강의자료를 활용하여 교양수업에 맞게 편집되었습니다.

# Python 환경 구축

- 수업에서 사용할 환경
  - Python 재단의 인터프리터 설치
    - <https://www.python.org>
  - 코드 편집
    - Visual Studio Code
    - <https://code.visualstudio.com>
      - Microsoft의 무료 통합 개발 환경(IDE)
        - 다양한 언어 지원 - 강력한 코드 편집 - 디버깅
        - 확장 가능성
        - 버전 관리 지원
        - 테마 및 사용자 정의
        - 무료 및 오픈 소스

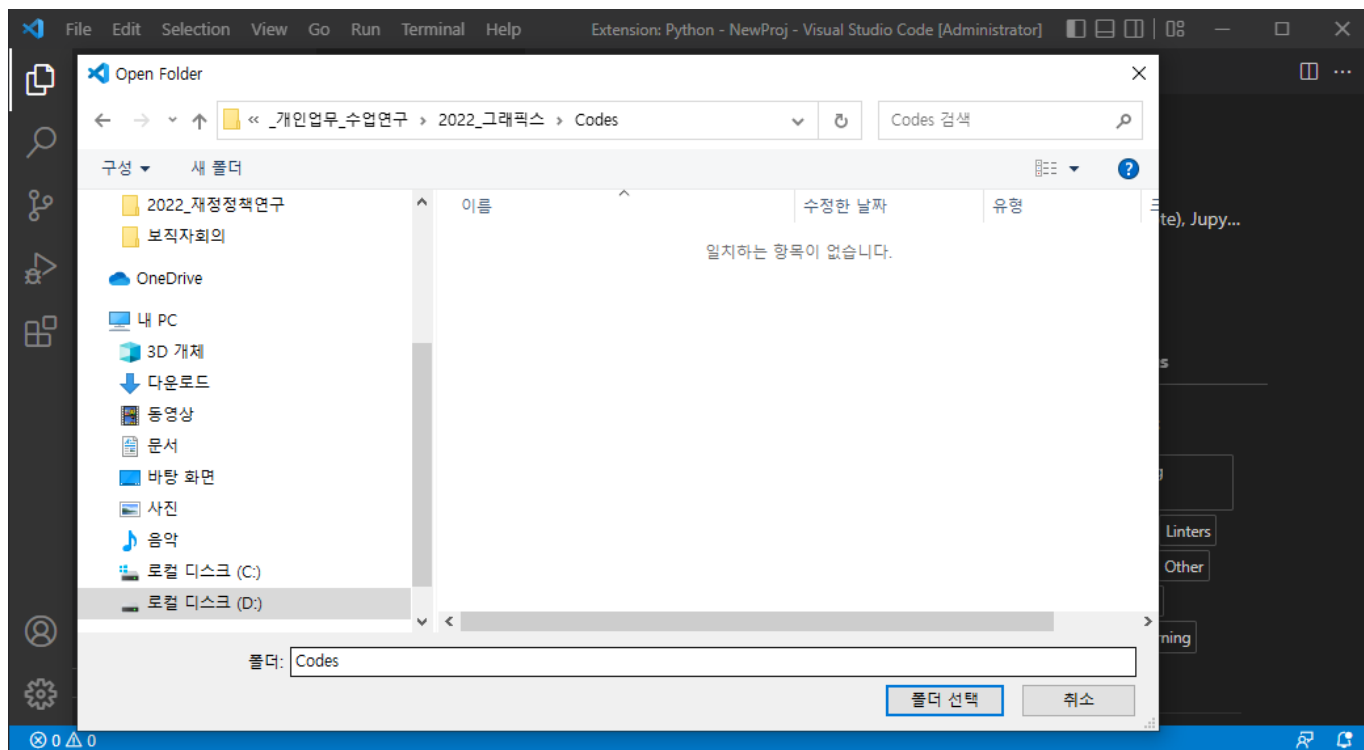


# Python 확장 설치



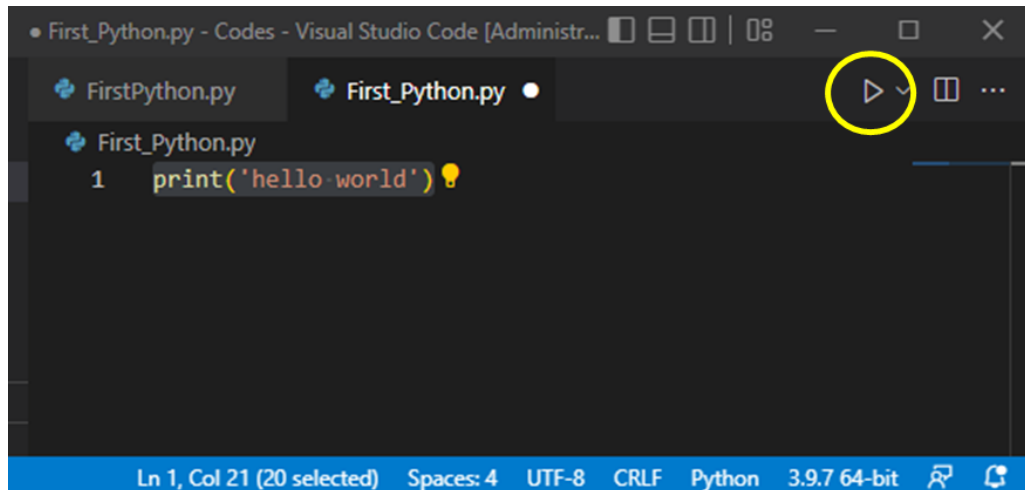
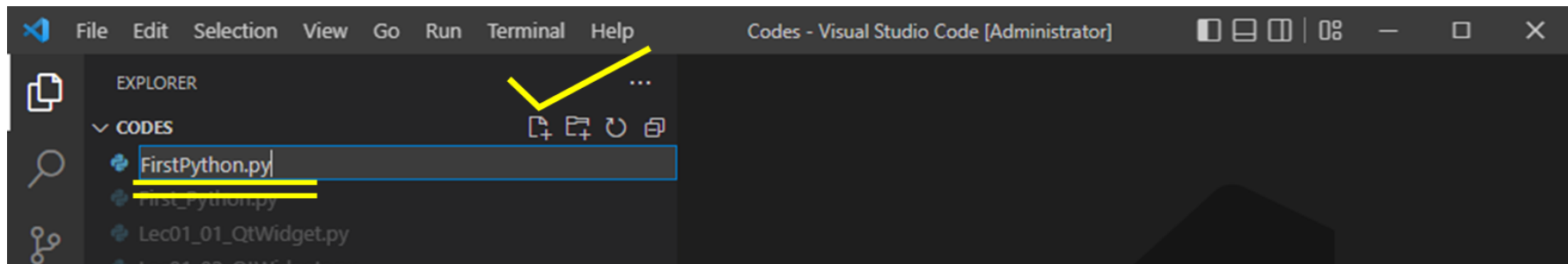
# Python 연습

- 폴더를 만들어 프로그램 코드를 만들어 보자
  - File 메뉴의 Open Folder로 자신의 작업공간 지정



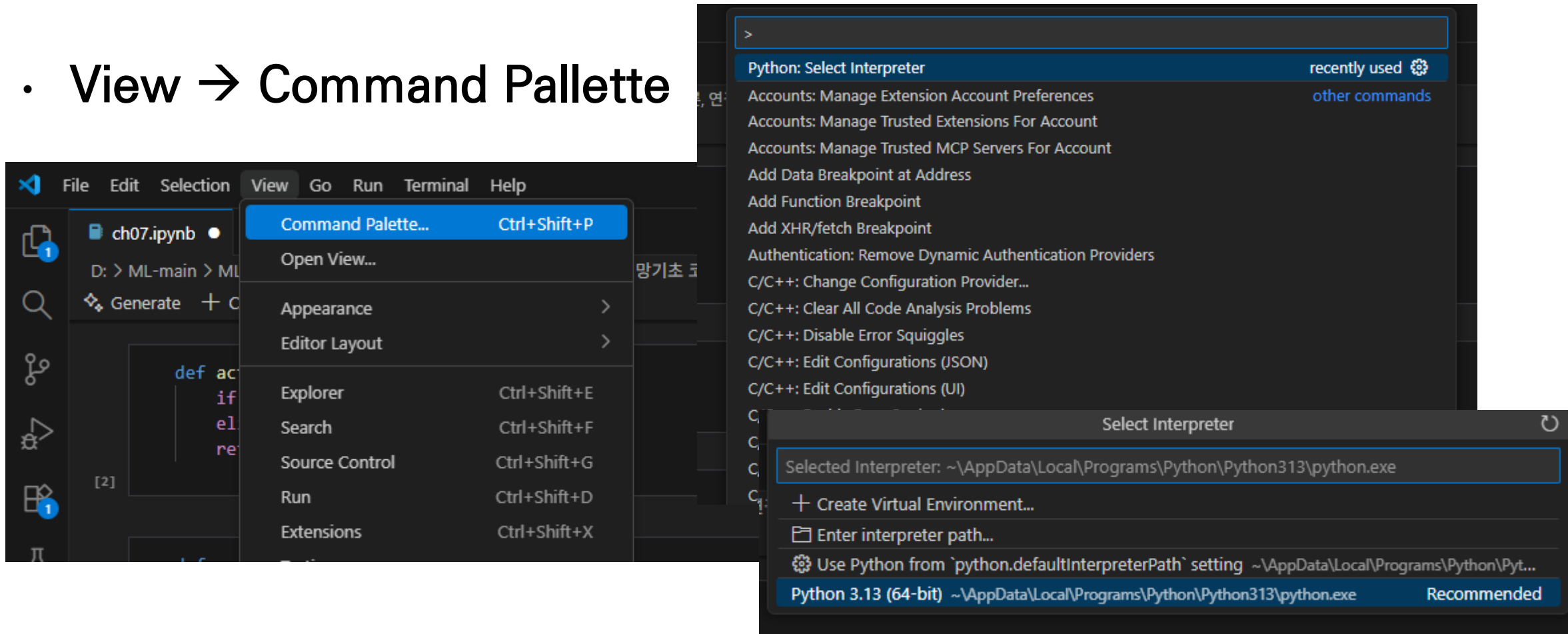
# Python 연습

- 폴더 내에 파일을 만들어 코딩 연습



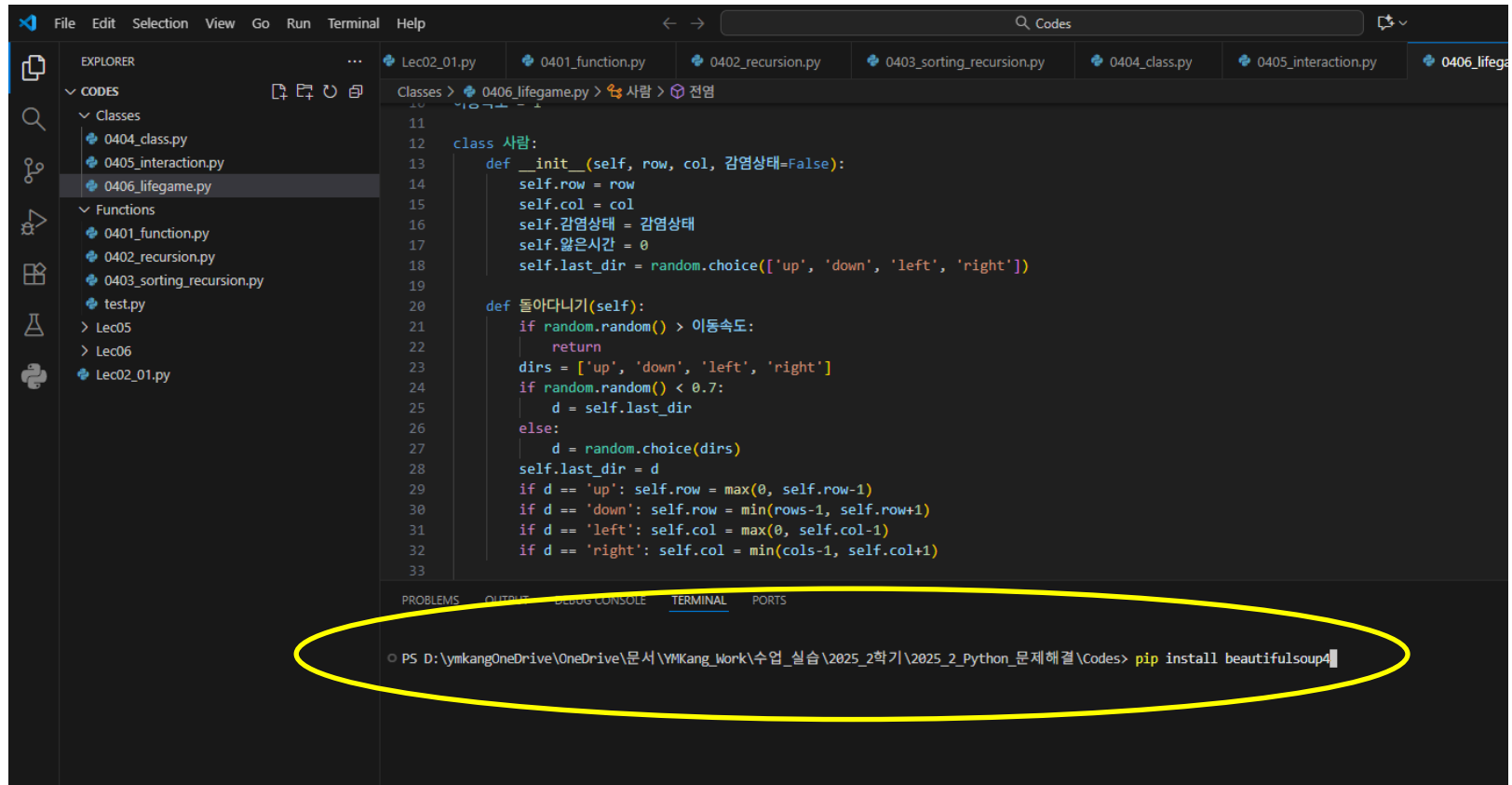
# Python 인터프리터 버전 선택

- View → Command Palette



- 프로젝트 1
  - 웹 사이트 내용 추출해 보기 (위키피디아)

C:\> pip install requests beautifulsoup4



The screenshot shows the Visual Studio Code interface. The Explorer panel on the left shows a project structure with files like 0404\_class.py, 0405\_interaction.py, 0406\_lifegame.py, 0401\_function.py, 0402\_recursion.py, 0403\_sorting\_recursion.py, test.py, Lec05, Lec06, and Lec02\_01.py. The main editor area shows the code for 0406\_lifegame.py, which defines a class '사람' (Person) with methods for initialization and movement. The terminal window at the bottom shows the command 'pip install beautifulsoup4' being executed in a PowerShell prompt.

```
class 사람:
    def __init__(self, row, col, 감염상태=False):
        self.row = row
        self.col = col
        self.감염상태 = 감염상태
        self.알은시간 = 0
        self.last_dir = random.choice(['up', 'down', 'left', 'right'])

    def 돌아다니기(self):
        if random.random() > 이동속도:
            return
        dirs = ['up', 'down', 'left', 'right']
        if random.random() < 0.7:
            d = self.last_dir
        else:
            d = random.choice(dirs)
        self.last_dir = d
        if d == 'up': self.row = max(0, self.row-1)
        if d == 'down': self.row = min(rows-1, self.row+1)
        if d == 'left': self.col = max(0, self.col-1)
        if d == 'right': self.col = min(cols-1, self.col+1)
```

PS D:\ymkangOneDrive\OneDrive\문서\YMKang\_Work\수업\_실습\2025\_2학기\2025\_2\_Python\_문제해결\Codes> pip install beautifulsoup4

```
import requests
from bs4 import BeautifulSoup
import re
```

```
url = "https://en.wikipedia.org/wiki/CORTIS"
```

```
def scrape_wikipedia_words(url):
    headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) Chrome/58.0.3029.110'}
    response = requests.get(url, headers=headers)
```

```
    if response.status_code != 200:
        print(f"Failed to retrieve page: {response.status_code}")
        return []
```

```
    soup = BeautifulSoup(response.text, 'html.parser')
```

```
    paragraphs = soup.find_all('p')
    full_text = ""
    for para in paragraphs:
        text = para.get_text()
        if text.strip():
            full_text += text + "\n"
```

```
    # 단어 추출: 영어 단어 기준 (\w+ 패턴으로 알파벳/숫자 추출, 소문자 변환)
    # 단어 경계 (\b) + 1개 이상의 단어 문자 (\w+) + 단어 경계 (\b)"
    # 개별 단어(공백이나 구두점으로 구분된 단어들)를 찾음.
    # 예: "Hello, world! 123" ["hello", "world", "123"] (소문자 변환 후).
    words = re.findall(r'\b\w+\b', full_text.lower())
    return words
```

```
word_list = scrape_wikipedia_words(url)
print("추출된 단어 리스트 (처음 50개):", word_list[:50])
print(f"총 단어 수: {len(word_list)}")
```

```
PS D:\...\2025_2_Python_문제해결\Codes\Lec06> & beautifulsoupTest.py
추출된 단어 리스트 (처음 50개): ['cortis', 'korean', '코르티스', 'stylised', 'in', 'all', 'caps', 'is', 'a',
'south', 'korean', 'boy', 'band', 'formed', 'by', 'big', 'hit', 'music', 'the', 'group', 'consists', 'of', 'five',
'members', 'james', 'juhoon', 'martin', 'seonghyeon', 'and', 'keonho', 'they', 'debuted', 'on', 'august',
'18', '2025', 'with', 'the', 'digital', 'single', 'what', 'you', 'want', 'the', 'lead', 'single', 'of', 'their', 'first',
'extended']
총 단어 수: 213
```

- 프로젝트 2
  - 워드 클라우드

```
C:\> pip install wordcloud nltk
```

```
from collections import Counter # 빈도 계산용 추가
from wordcloud import WordCloud # 워드 클라우드 생성용 추가
import matplotlib.pyplot as plt # 그래프 표시용 추가
```

```
import matplotlib.font_manager as fm # 폰트 경로 찾기용 추가
font_path = fm.findfont('NanumGothic')
```

```
...
```

```
word_list = scrape_wikipedia_words(url)
print("추출된 단어 리스트 (처음 50개):", word_list[:50])
print(f"총 단어 수: {len(word_list)}")
```

```
# 워드 클라우드 생성 부분 추가 (기존 코드 유지)
if word_list:
    # 단어 빈도 계산 (자주 등장하는 단어 위주)
    word_freq = Counter(word_list)
```

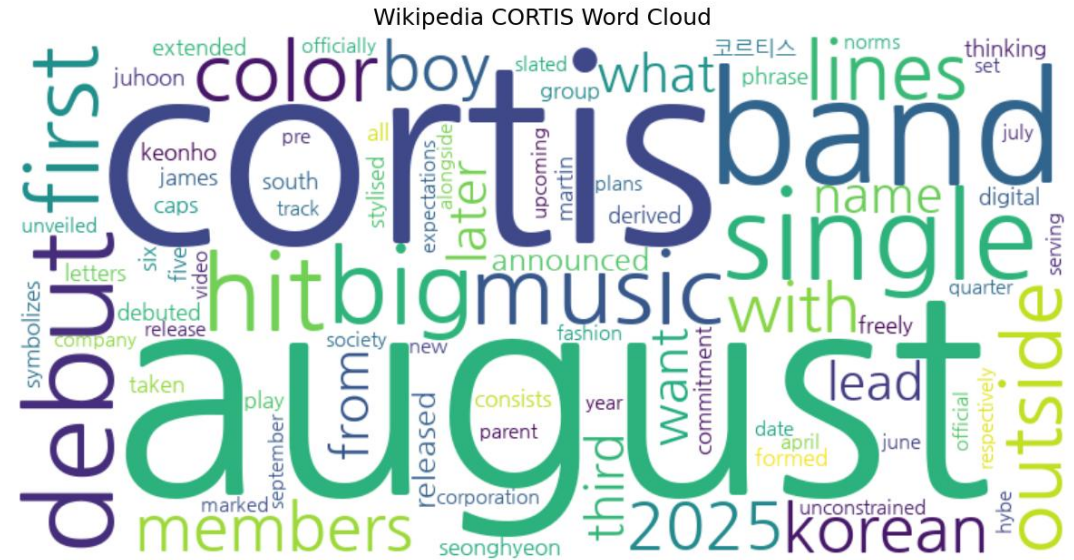
```
# 불용어(Stopwords) 제거 (예: 'the', 'and' 등 흔한 단어, 간단히 하드코딩)
stopwords = {'the', 'and', 'or', 'but', 'in', 'on', 'at', 'to', 'for', 'of', 'a', 'an', 'is', 'are', 'was', 'were', 'be', 'been', 'have', 'has', 'had', 'do', 'does', 'did', 'will', 'would', 'could', 'should',
'may', 'might', 'must', 'can', 'this', 'that', 'these', 'those', 'i', 'you', 'he', 'she', 'it', 'we', 'they', 'me', 'him', 'her', 'us', 'them', 'my', 'your', 'his', 'its', 'our', 'their'}
filtered_freq = {word: freq for word, freq in word_freq.items() if word not in stopwords and len(word) > 2}
```

```
# 워드 클라우드 생성
```

```
wc = WordCloud(width=800, height=400, background_color='white', max_words=100, font_path=font_path).generate_from_frequencies(filtered_freq)
```

```
# Matplotlib으로 표시
```

```
plt.figure(figsize=(10, 5))
plt.imshow(wc, interpolation='bilinear')
plt.axis('off')
plt.title('Wikipedia CORTIS Word Cloud')
plt.show()
```





Questions?