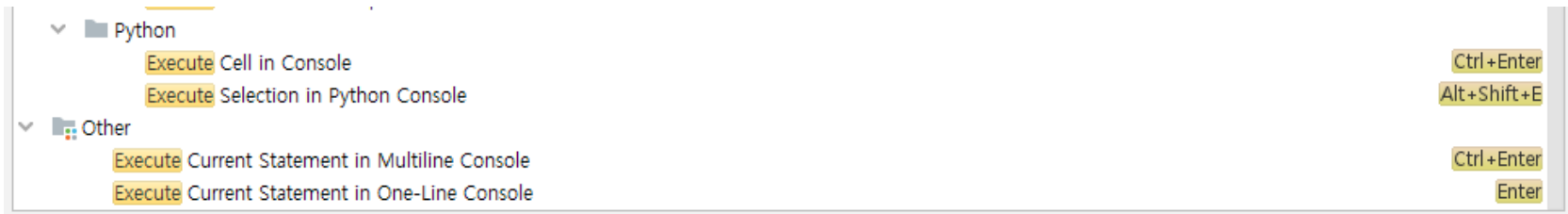


# 파이참 콘솔 활용 팁

# 콘솔 활용

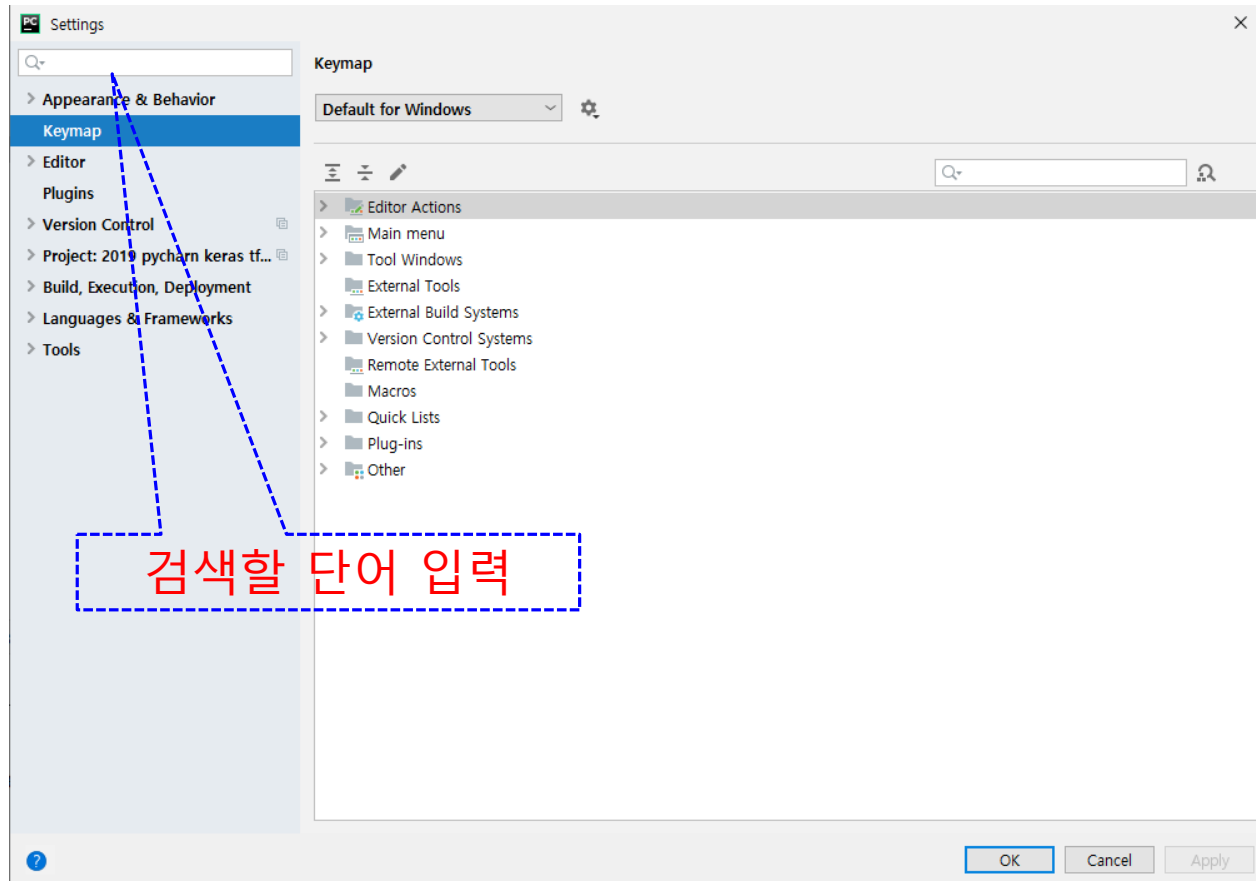
- 소스의 셀을 콘솔에서 실행
  - Alt + Shift + e
- 콘솔 내부에서 코딩 후 실행 시
  - Ctrl + Enter
    - 바로 실행이 안되고 다음 줄도 코딩 가능
    - 콘솔에서 여러 줄 코딩 후 실행



# 주요 파이참 환경 설정 팁

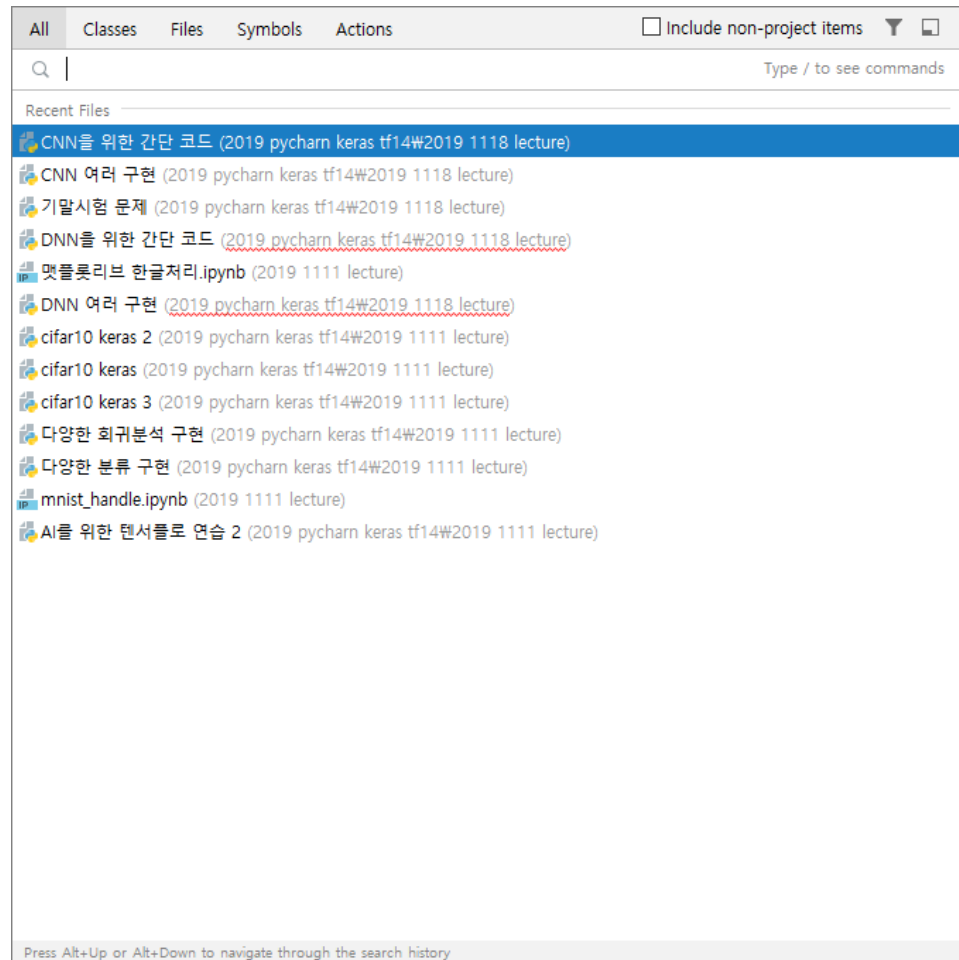
# 환경설정 메뉴, 파일 | settings

- Ctrl + Alt + s



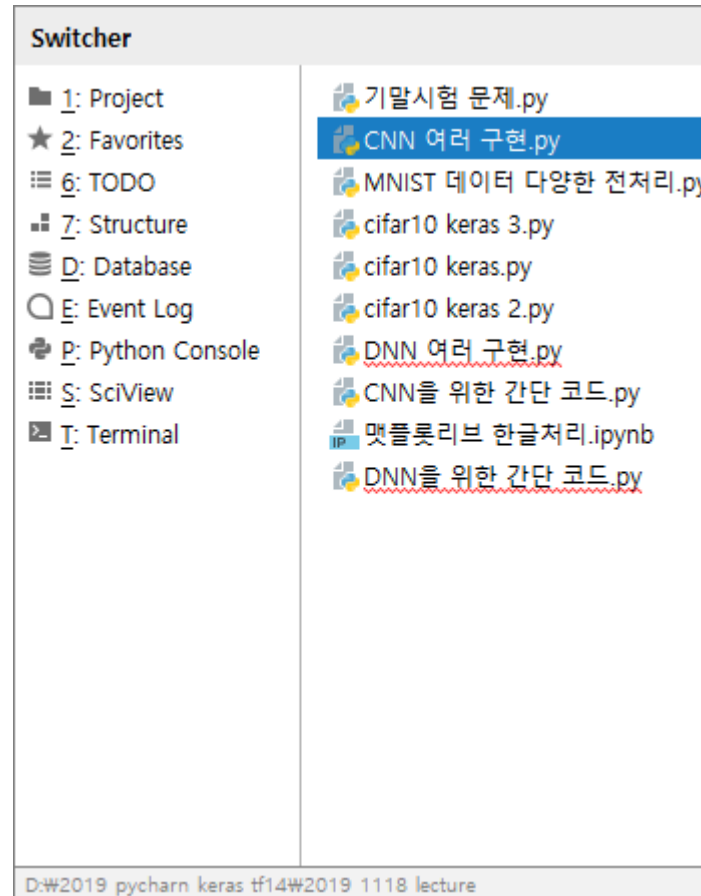
# 전체 검색

- **Shift + shift**
  - 검색의 대상을 탭으로 이동



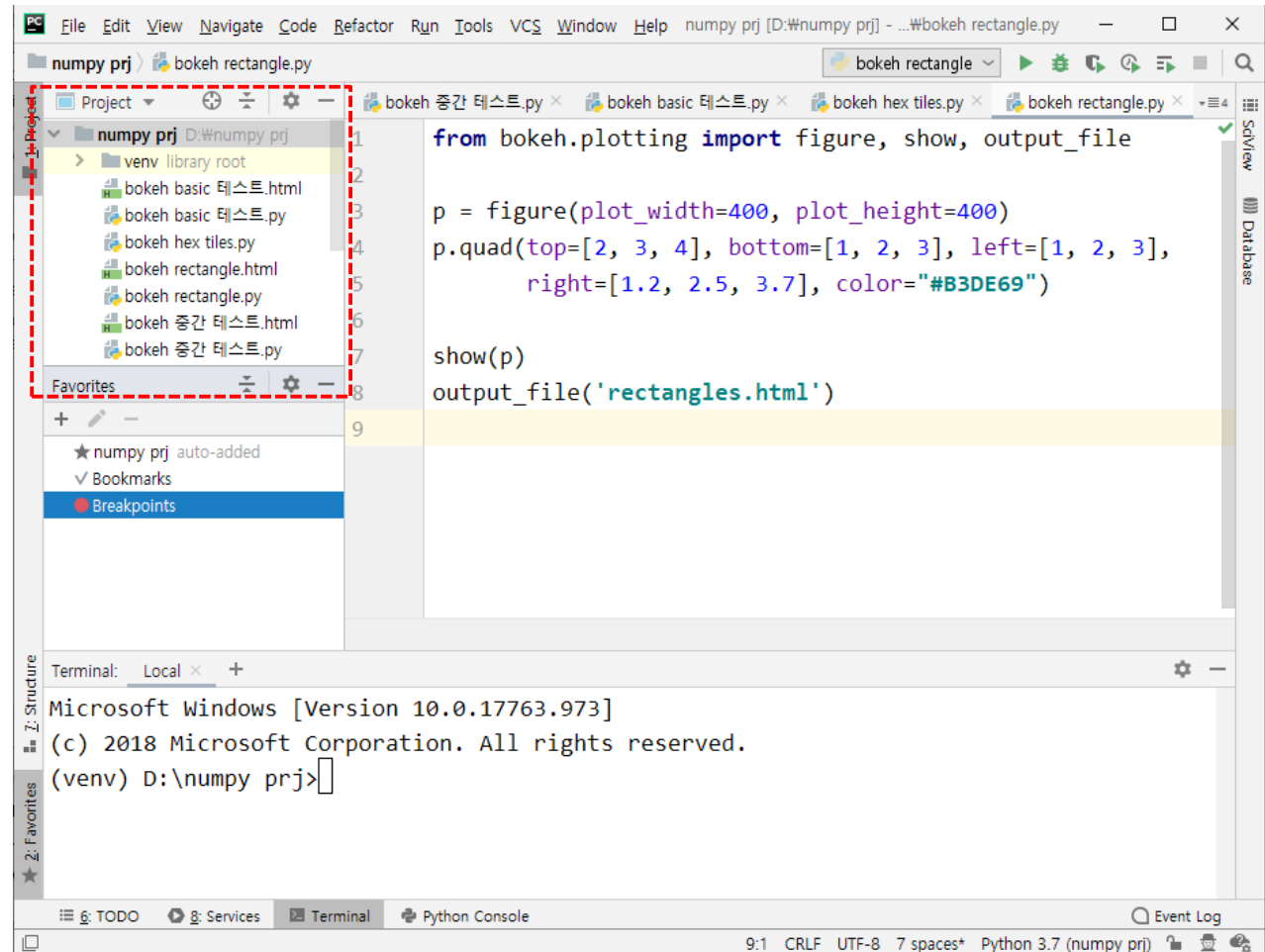
# 윈도 이동

- **Switcher: Ctrl + tab**



# 프로젝트 창 보이기 / 감추기

- Alt + 1



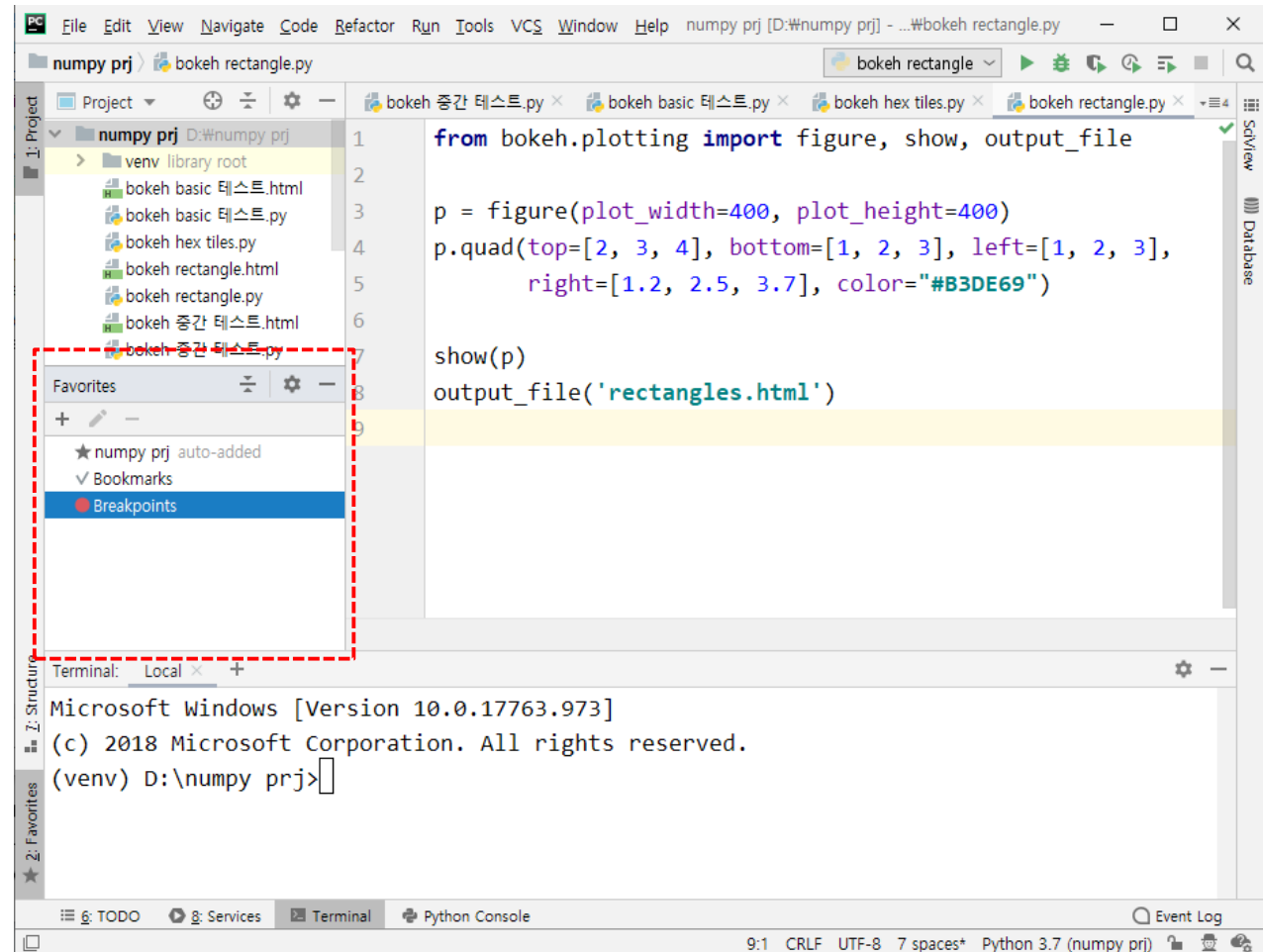
# 즐거찾기 창 보이기 / 감추기

## • Alt + 2

- Favorites 창
- 내부 \*
  - 즐거찾기 폴더
  - + 로 추가
  - - 로 삭제

## • 삽입할 파일

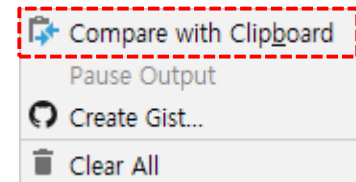
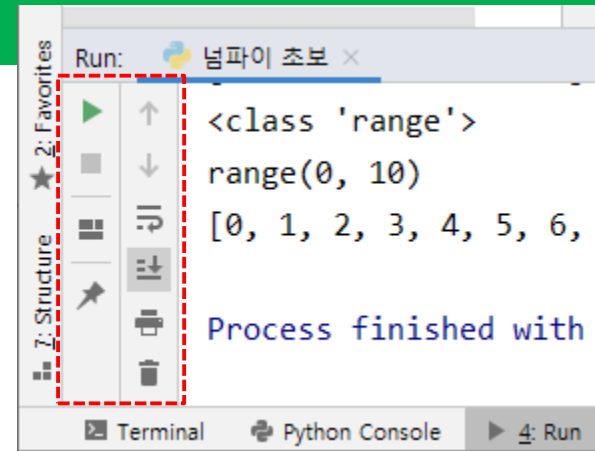
- 드래그 드롭
- 팝업 메뉴
  - Add to New Favorites List





# 실행 창 보이기 / 감추기

- **Run 창 (Alt + 4)**
- **왼쪽 버튼**
  - Rerun(마지막 실행 파일 재실행)
  - Stop(현재 실행 중인 파일 실행 중단)
  - Restore Layout(레이아웃 초기화)
  - Pin Tab(현재 실행 탭 고정)
  - Up/Down to Stack Trace(trace 상에서 상위 또는 하위 단계로 이동)
  - Soft-Wrap(토글 키)
    - 출력 내용이 한 줄을 넘기면 아래 줄에 출력하거나 윈도우 크기에 상관없이 한 줄에 출력하여 스크롤을 사용하도록
  - Scroll to the end(제일 아래쪽으로 스크롤)
  - Print(출력 결과를 정말 프린터에서 출력)
  - Clear All(현재 출력 결과를 모두 지우기)
- **Run 창 팝업 메뉴 Compare with Clipboard 항목**
  - 현재 클립보드에 있는(즉, Ctrl + C 등으로 복사한) 내용과 출력 결과를 비교하는 창을 띄움
    - 정답 출력 결과를 복사해 봤다면 유용하게 사용

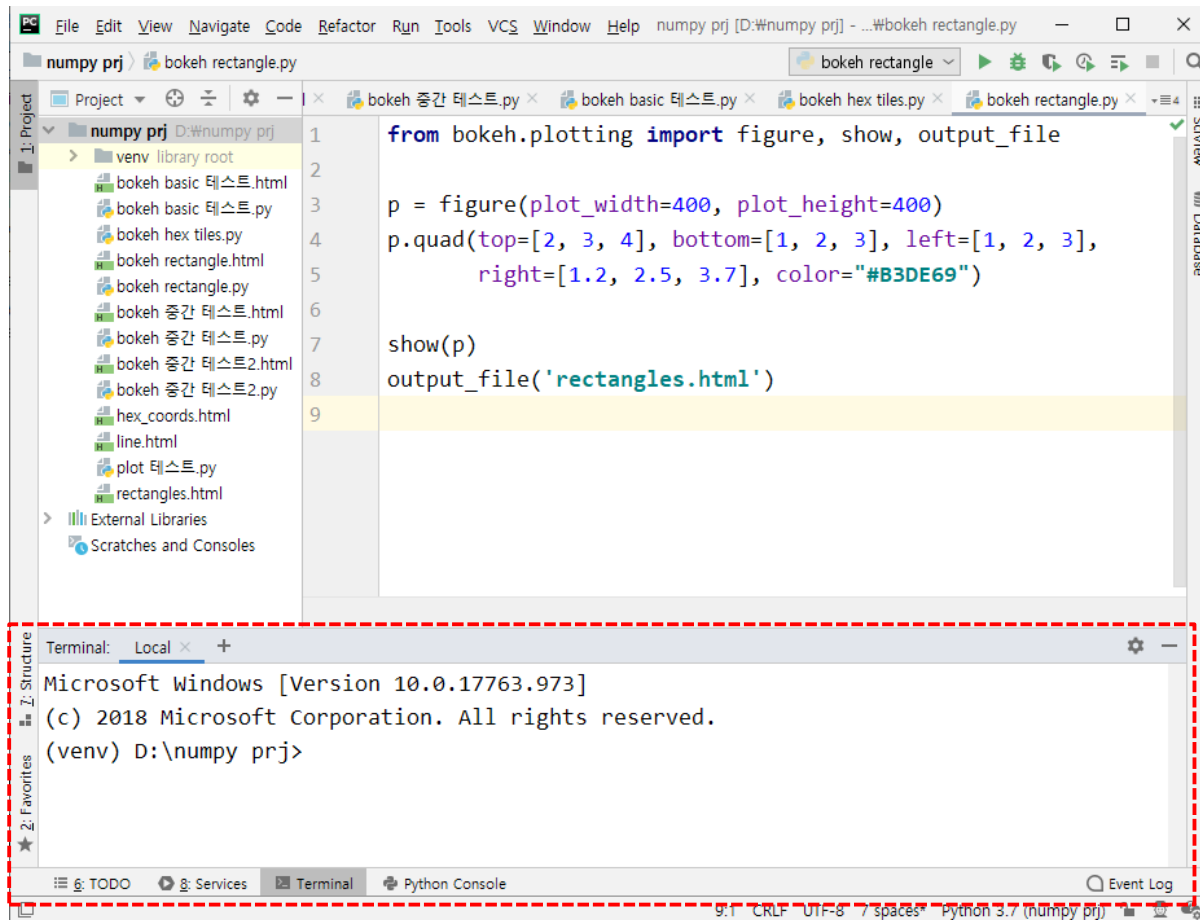


# TODO 창 보이기 / 감추기

- **Alt + 6**
  - 소스에서 주석(#)을 달면서 앞에 TODO:라는 문구를 적으면 해당 주석은 특별히 눈에 띄는 연두색으로 바뀜
- **TODO 창**
  - 앞으로 해야 할 것을 모아 놓은 것
  - 이를 나중에 찾아보려면
    - **TODO 창을 클릭하거나 Alt + 6**

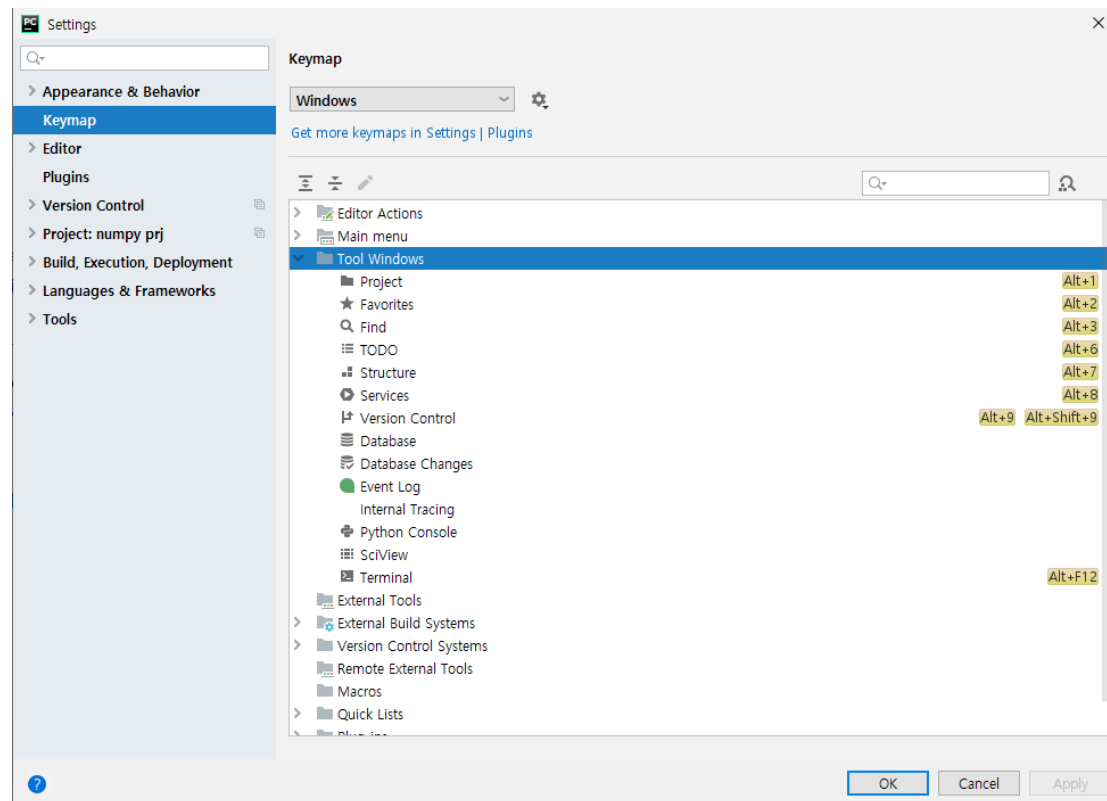
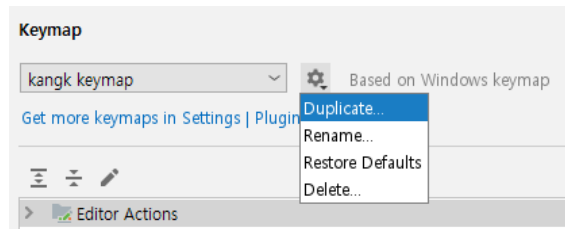
# 터미널(프롬프트) 창 보이기 / 감추기

- alt + F12



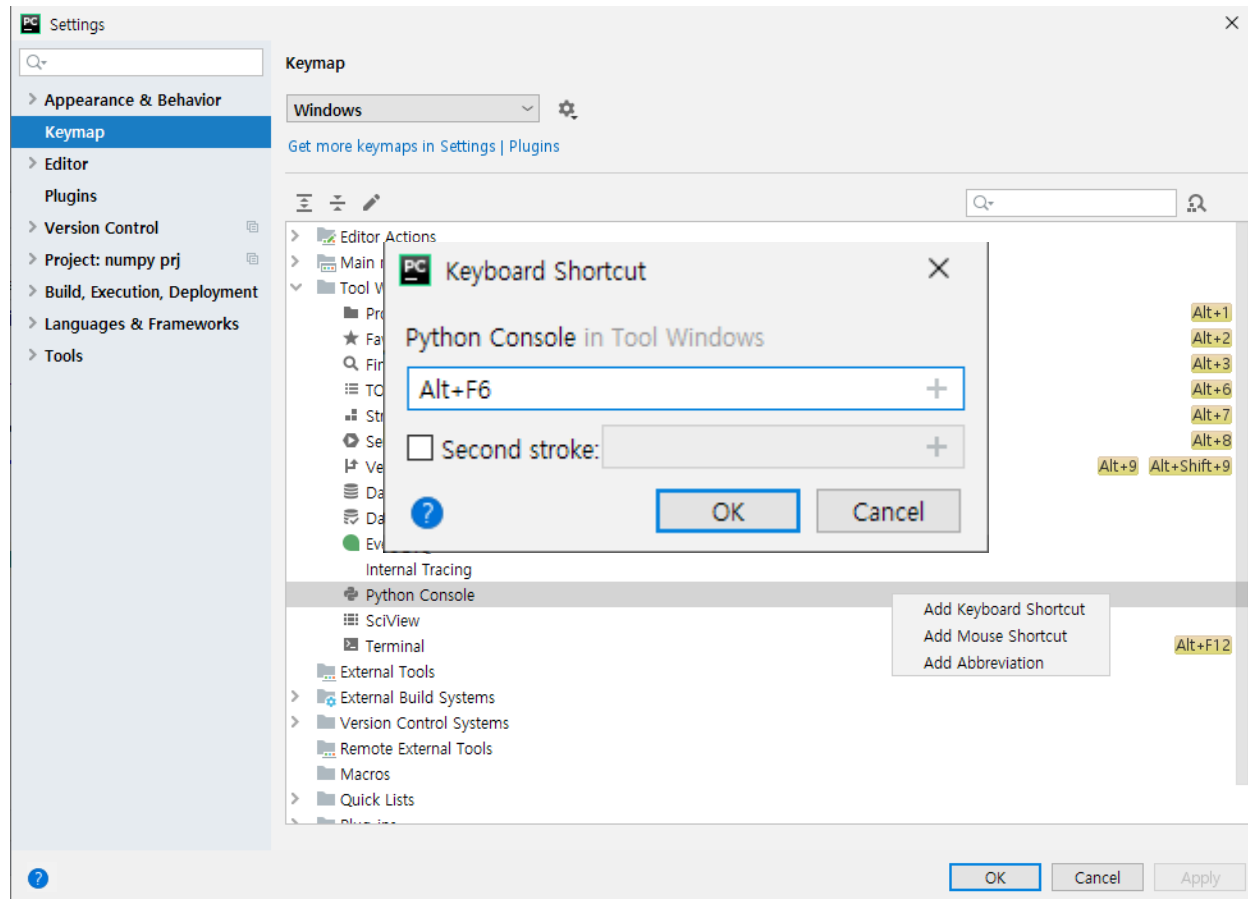
# Key map

- Settings의 keymap
  - 필요하면 복사가 가능



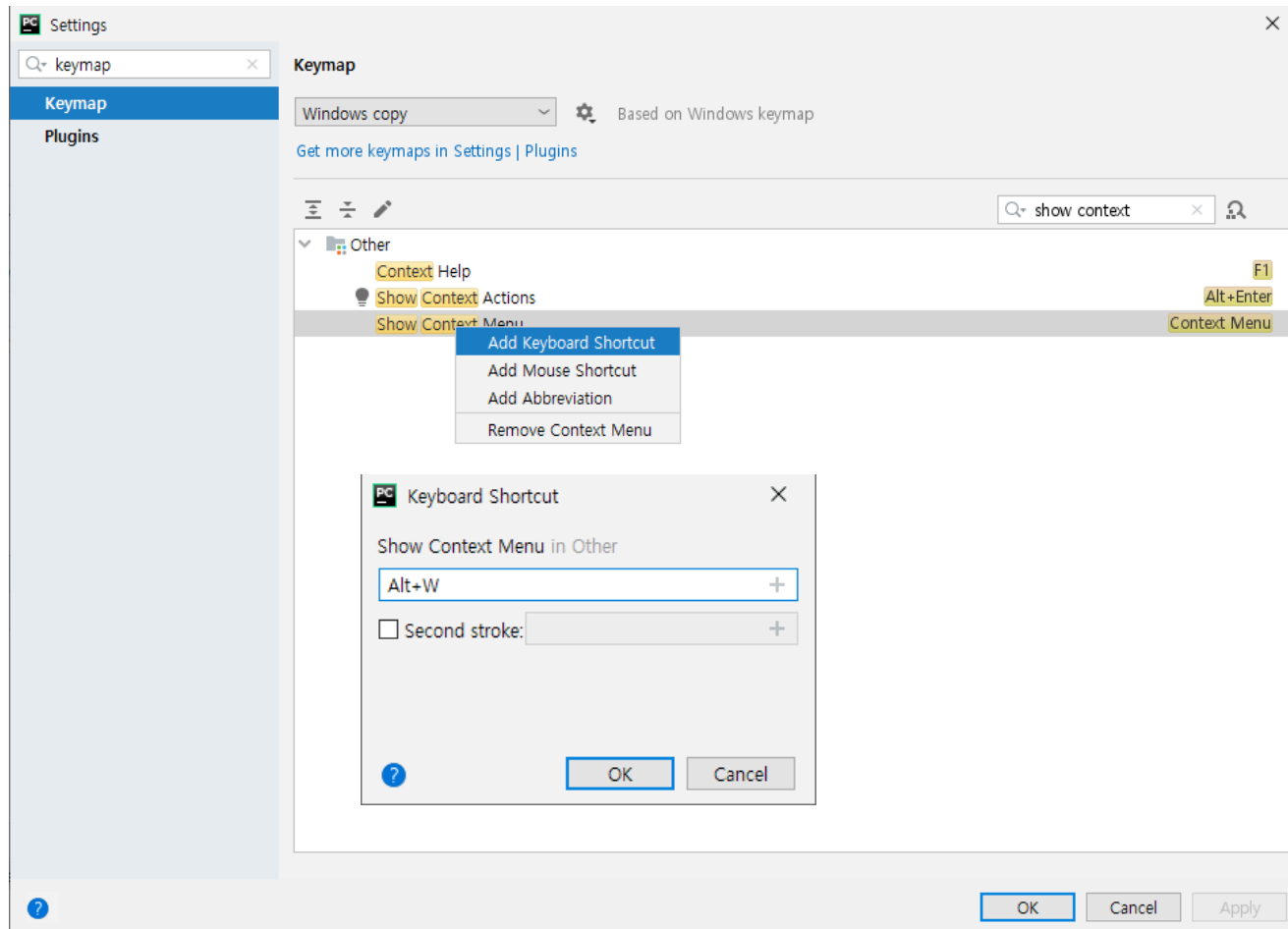
# Key map 추가

- Python console: Alt + F6로 추가



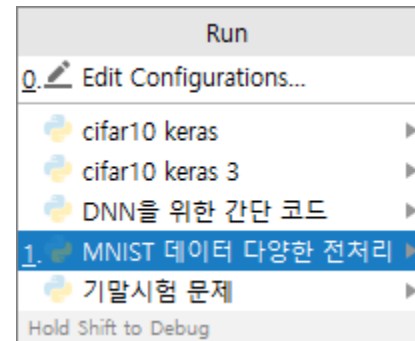
# Key map 추가

- 바로가기 메뉴(context show menu): Alt + W로 추가



# 실행

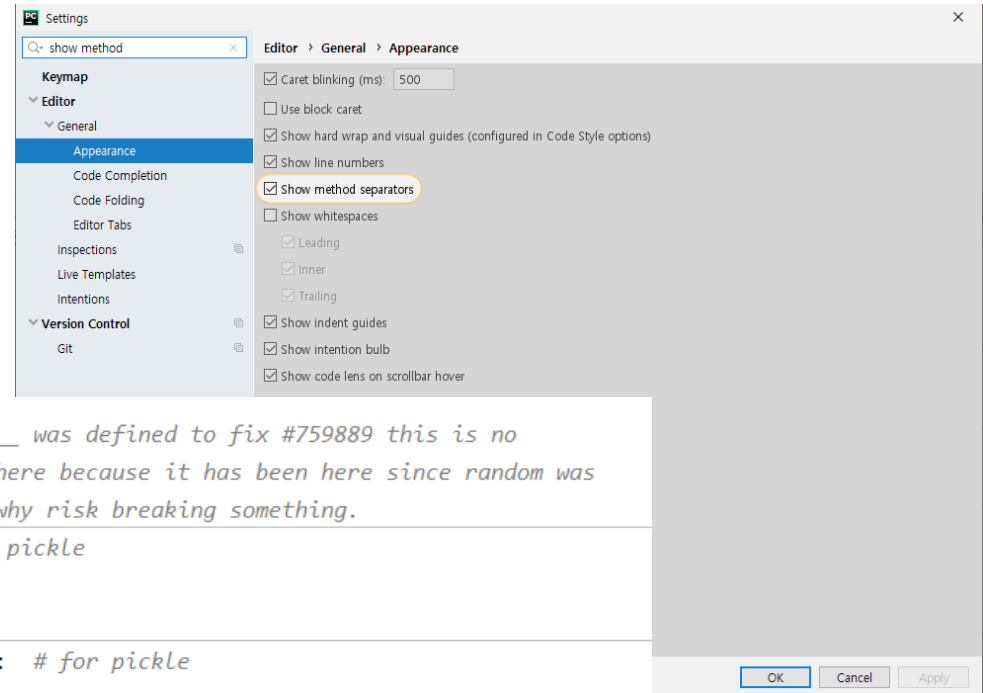
- **Ctrl + shift + F10**
  - 현재 소스 실행
- **Shift + F10**
  - 최근 소스 재실행
- **Alt + shift + F10**
  - 실행할 소스 보여 선택할 수 있도록



- **Alt + shift + E**
  - 현재 줄이나 선택한 블록을 파이썬 Console에서 바로 실행

# 소스 보기 설정

- Show method separators



```
# Issue 17489: Since __reduce__ was defined to fix #759889 this is no
# longer called; we leave it here because it has been here since random was
# rewritten back in 2001 and why risk breaking something.

def __getstate__(self): # for pickle
    return self.getstate()

def __setstate__(self, state): # for pickle
    self.setstate(state)

def __reduce__(self):
    return self.__class__, (), self.getstate()

## ----- integer methods -----

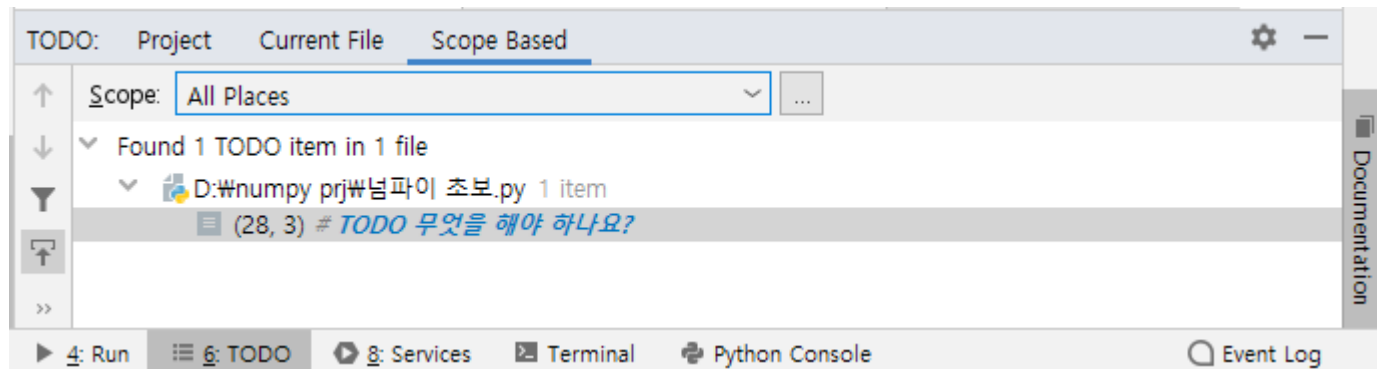
def randrange(self, start, stop=None, step=1, _int=int):
    """Choose a random item from range(start, stop[, step]).
```



파이참 수스 편집 단축키

# 에디터 주요 기능(1)

- 코드를 입력하다가 **ctrl + space** 누르기
  - 소스 코딩 도움
- **control + /**
  - 주석 달기
- **나중에 할 일 기록**
  - 소스에 할 일을 기록 # TODO로 시작
    - **# TODO 무엇을 해야 하나요?**
  - Alt+6 로 TODO 창을 띄우면 할일 목록을 일괄적으로 관리 가능



## 에디터 주요 기능(2)

- 라인 번호 이동
  - Ctrl + G
- 커서 있는 한 줄(또는 드래그한 선택 범위)을 복사해 아래에 붙여 넣기
  - Ctrl + D(Duplicate):
- 커서 있는 한 줄(또는 드래그한 선택 범위)을 잘라내기/복사
  - Ctrl + X / Ctrl + C
- 현재 선택 범위의 한 단계 위 범위를 전체 선택(블록이나 괄호 등)
  - Ctrl + W
- 코드 내에 import들을 파일 맨 위로 모아 잘 정리
  - Ctrl + Alt + O(Import Optimization)
- 코드의 빈 줄, indentation 등을 한 번에 정리
  - Ctrl + Alt + L
- 정리 대화 상자
  - Ctrl + Alt + Shift + L
- 한 행 이동
  - Alt + shift + 위/아래 화살표

## 에디터 주요 기능(3)

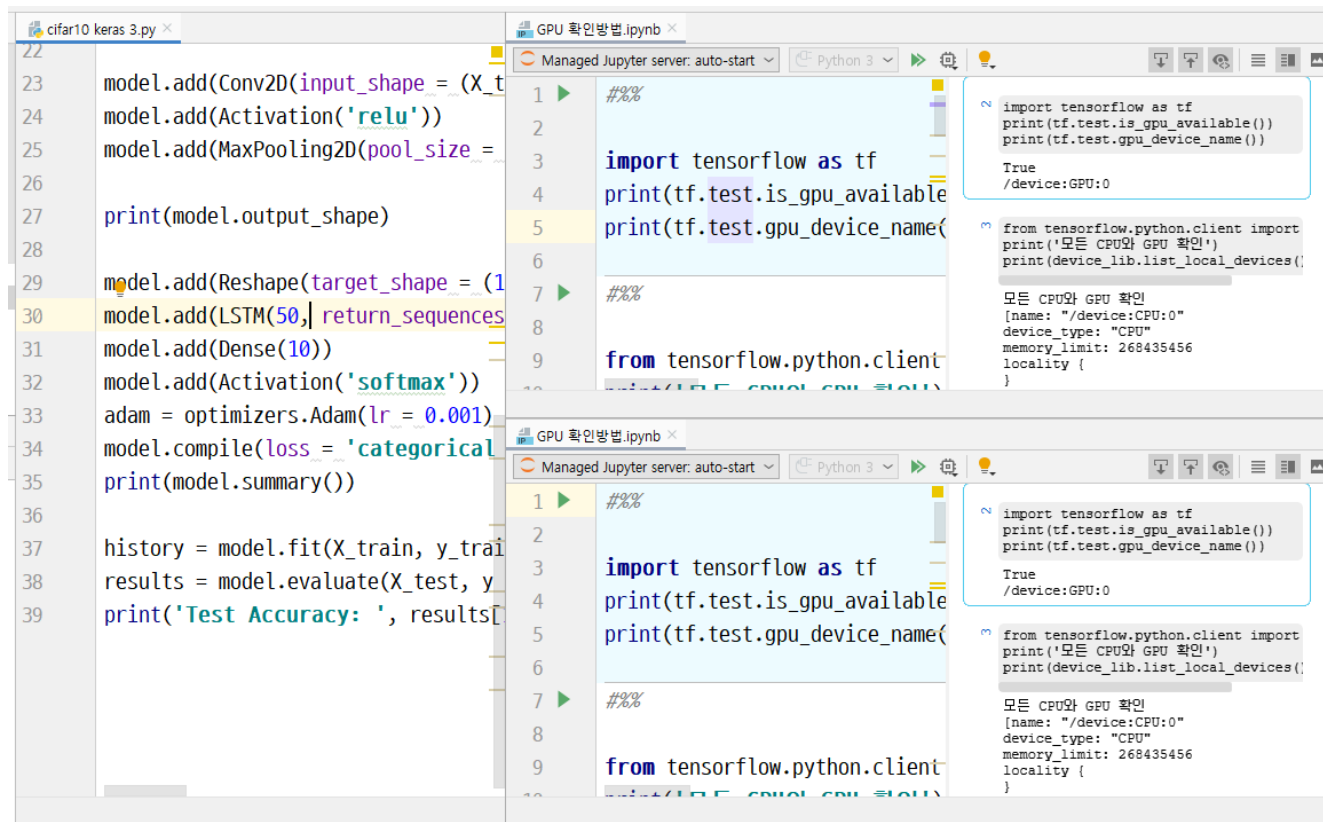
- 해당 변수/함수가 선언된 위치로 화면/커서가 이동
  - Ctrl + 마우스
    - 풍선 도움말
  - Ctrl + 클릭
    - 해당 구현 소스 파일로 이동
- Ctrl + R
  - 찾기 및 바꾸기의 기본 단축키
- Alt + enter
  - 행의 인텐트 바로 가기
- Alt + 화살표 >, <
  - 소스 간 이동
- 함수의 소스로 이동
  - Ctrl + 마우스 클릭, Ctrl + B
- 자동 들여쓰기
  - Ctrl + alt + i

```
singer = ('BTS', '불빨간사춘기', 'BTS', '블랙핑크', '태연')
credit = ([2020, 1, 18], [2020, 2, 17])
space = '밤', '낮', '해', '달'
```

Convert list to set ▶  
Convert list to tuple ▶

# 소스 여러 창 열기

- 편집 창(메인 화면)의 탭을 우클릭
  - Split Vertically를 클릭
  - Split Horizontally

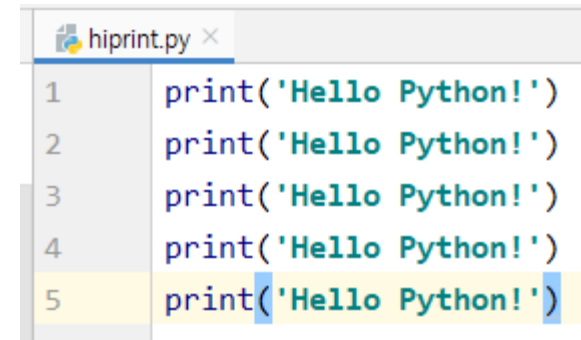


# 여러 부분을 동시에 편집

- **Alt + 클릭**
  - 커서를 원하는 곳에 일단 놓고
    - 같은 것을 입력하고 싶은 곳에 Alt를 누른 채로 새로 클릭
      - 커서가 여러 개가 되는 것을 확인 가능
    - 이 상태에서 키보드로 입력을 시작하면 여러 곳에서 한번에 입력이 가능
- **Ctrl을 연속 두 번 누르는데,**
  - 두번째는 누른 채로 방향키로 커서를 위 아래 좌우
- - [Wheel Mouse Button]으로 드래그한다.
- - [Alt] + [Shift] + [Insert] 를 눌러서 옵션을 On/Off할 수 있다. On 상태에서는 [Shift]를 누르고 세로방향으로도 영역을 지정할 수 있다.
- 출처: <https://forgiveall.tistory.com/454> [하하하하하]

# 일련의 작업 단축키 활용

- **Alt + 1: 프로젝트 창 이동**
- **Alt + insert: 파일 생성**
  - Python file
  - 파일 이름 입력, hiprint.py, 코딩
    - `print 'Hello Python!'`
- **Ctrl + D 4번: 줄 복사**
  - `Print 'Hello Python!'`
  - `Print 'Hello Python!'`
  - `Print 'Hello Python!'`
  - `Print 'Hello Python!'`
  - `Print 'Hello Python!'`
- **Ctrl + shift + F10: 현재 파일 실행**
  - 오류 발생
- **Ctrl + ctrl + 화살표 아래로: 세로 편집**
  - `print` 이후, 코딩 완성
    - `print('Hello Python!')`
- **Shift + F10: 이전 실행 파일 재실행**
  - 실행 성공



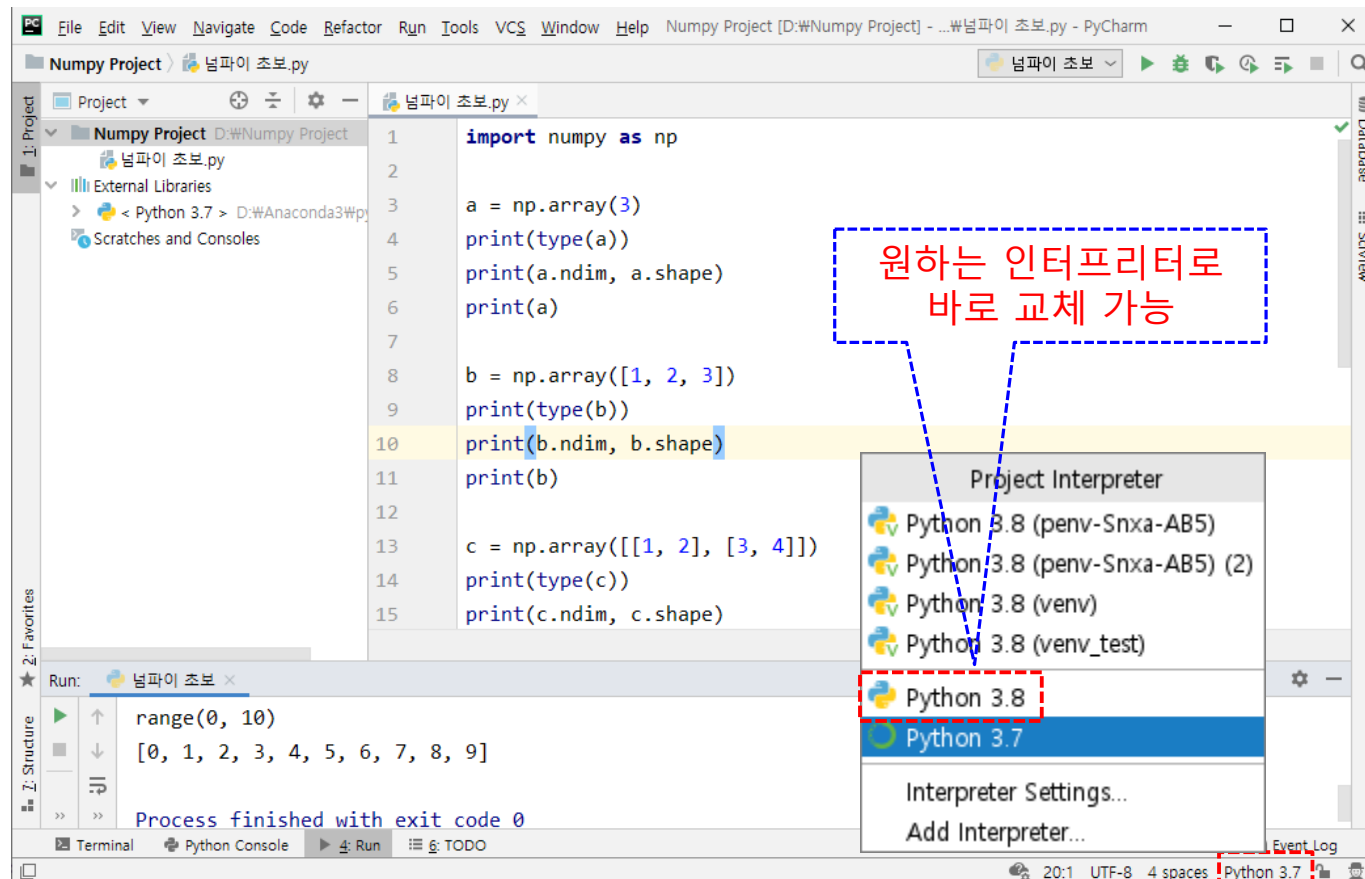
```
hiprint.py x
1 print('Hello Python!')
2 print('Hello Python!')
3 print('Hello Python!')
4 print('Hello Python!')
5 print('Hello Python!')
```

현재 인터프리터를  
상태 바에서 바로 변경



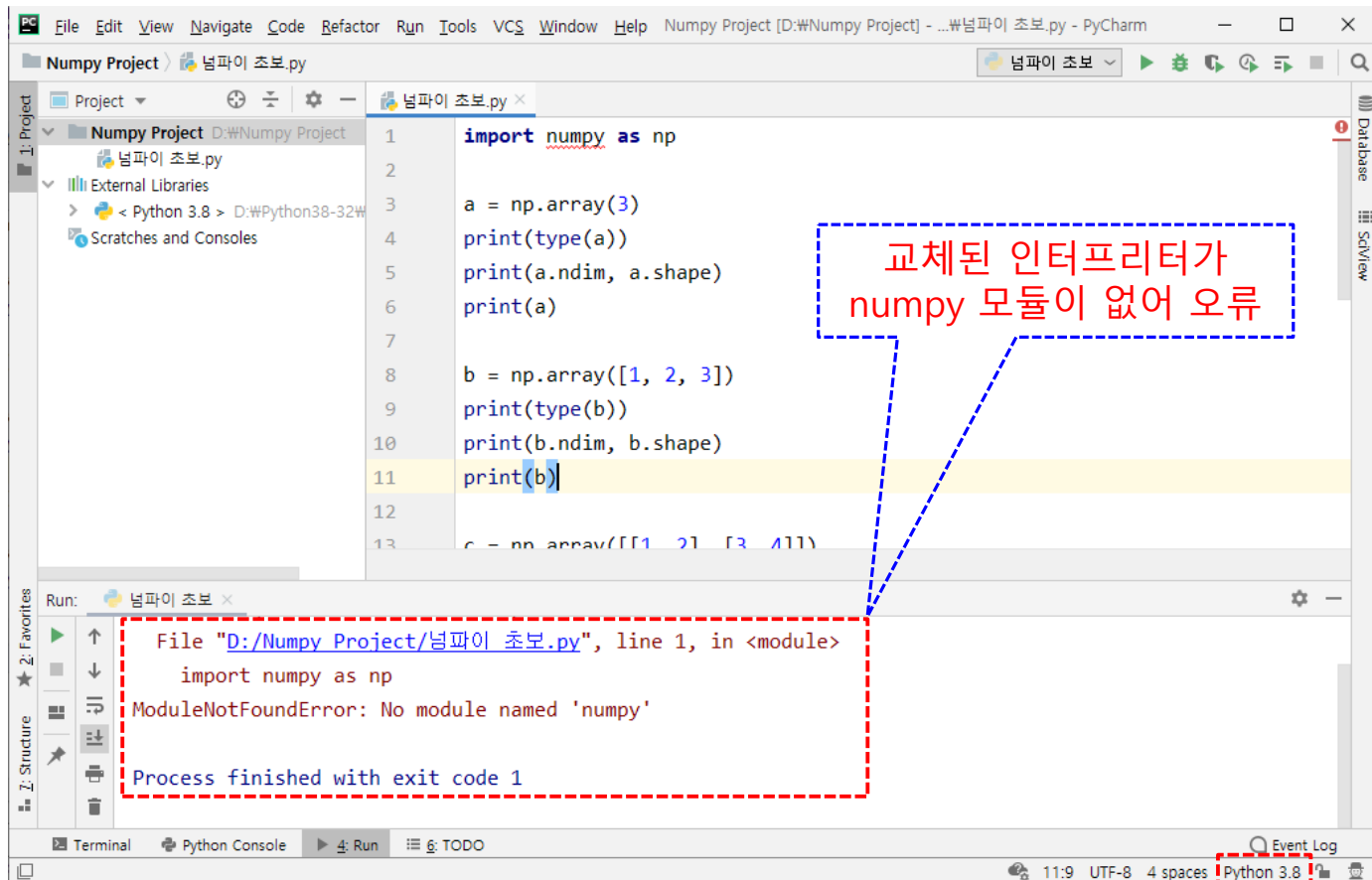
# 현재 인터프리터

- 아나콘다의 Python 3.7
  - Numpy 의 코드 실행 가능



# 인터프리터 교체 후

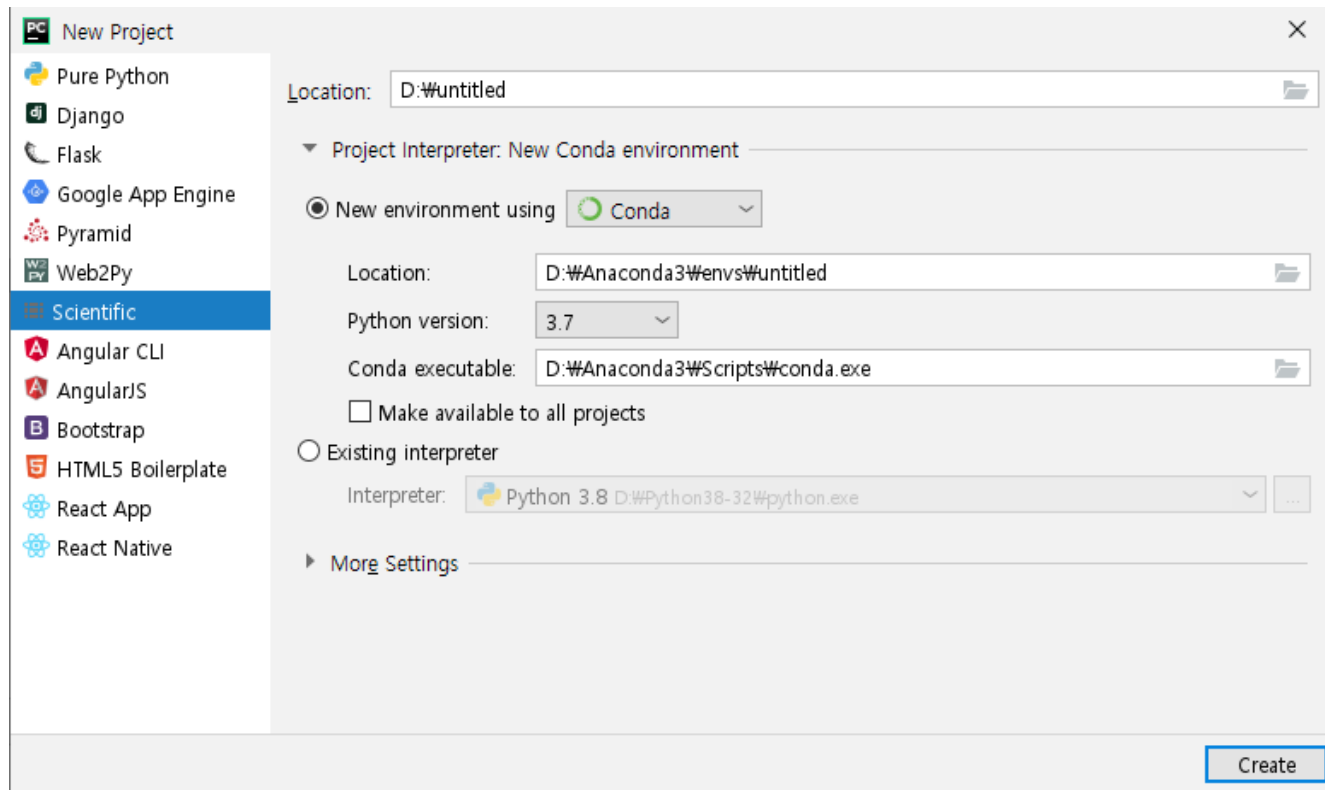
- 기본 파이썬의 Python 3.8로 교체
  - Numpy 의 코드 실행 불가능



Scientific 프로젝트 생성:  
프로페셔널 버전 기능

# 과학용 프로젝트

- 프로젝트 생성 시 scientific으로
  - 주로 Conda로 생성



# 셀마다 실행 활용

## • 소스 파일

– #%% 단위로 콘솔에서 실행됨

• **Ctrl + Enter**

