제9강 Debugging

© 이대현

학습 목차

- 버그의 이유
- 디버깅 방법
- 예외 발생(Rasing Exception)
- 표명(Assertion)
- Logging
- Pycharm Debugging

© 이대현

버그는 왜 발생하는가?

© 이대현

버그는 왜 발생하는가?

- 예상치 못한 입력
- 논리적 오류

- 스트레스 ㅋㅋ

© 이대현

디버깅 어떤 식으로 합니까?

© 이대현

디버깅 어떤 식으로 합니까?

- 코드 다시 보기 – 코린이
- 프린트 – 초보
- 실시간 디버깅 – 중수
- 테스트 자동화 – 고수
- 초고수는?

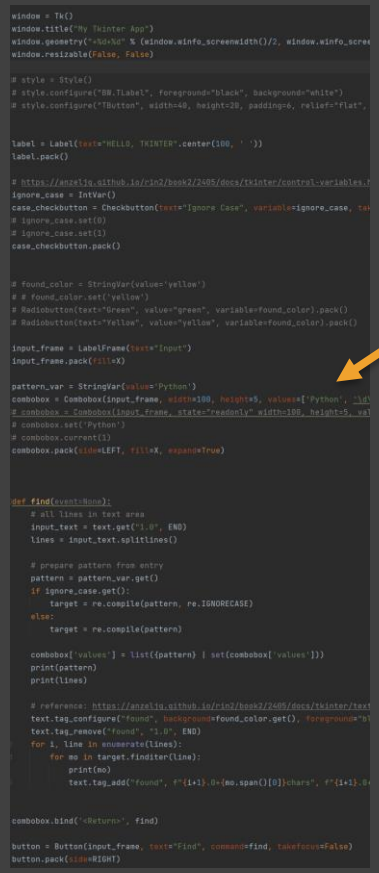
© 이대현

예외 처리 – 잘못된 입력에 대한 대처

```
age = int(input('Enter Age:'))
if age < 18:
    print('Error: Age should be over 18.'))
```

if 를 사용한 예외 처리의 문제점은??

© 이대현



장문의 코드 중간에서 발생하는 예외 상황의 경우 GOTO 가 필요함.

© 이대현

try – except 를 이용한 예외 처리

try:

```
window = Tk()
window.title("My Tkinter App")
window.geometry("+%d+%d" % (window.winfo_screenwidth()/2, window.winfo_scre
window.resizable(False, False)

# style = Style()
# style.configure("BW.TLabel", foreground="black", background="white")
# style.configure("TButton", width=40, height=20, padding=6, relief="flat",


label = Label(text="HELLO, TKINTER".center(100, ' '))
label.pack()

# https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/control-variables.
ignore_case = IntVar()
case_checkbutton = Checkbutton(text="Ignore Case", variable=ignore_case, ta
# ignore_case.set(0)
# ignore_case.set(1)
case_checkbutton.pack()


# found_color = StringVar(value='yellow')
# # found_color.set('yellow')
# Radiobutton(text="Green", value="green", variable=found_color).pack()
# Radiobutton(text="Yellow", value="yellow", variable=found_color).pack()

input_frame = LabelFrame(text="Input")
input_frame.pack(fill=X)

pattern_var = StringVar(value='Python')
combobox = Combobox(input_frame, width=100, height=5, values=['Python', '\d
# combobox = Combobox(input_frame, state="readonly" width=100, height=5, val
# combobox.set('Python')
# combobox.current(1)
combobox.pack(side=LEFT, fill=X, expand=True)



def find(event=None):
    # all lines in text area
    input_text = text.get("1.0", END)
    lines = input_text.splitlines()

    # prepare pattern from entry
    pattern = pattern_var.get()
    if ignore_case.get():
        target = re.compile(pattern, re.IGNORECASE)
    else:
        target = re.compile(pattern)

    combobox['values'] = list({pattern} | set(combobox['values']))
    print(pattern)
    print(lines)

    # reference: https://anzeljg.github.io/rin2/book2/2405/docs/tkinter/text
    text.tag_configure("found", background=found_color.get(), foreground="bl
    text.tag_remove("found", "1.0", END)
    for i, line in enumerate(lines):
        for mo in target.finditer(line):
            print(mo)
            text.tag_add("found", f"{i+1}.0+{mo.span()[0]}chars", f"{i+1}.0+
```

except:

예외 처리를 한곳에 모아 구조화시 킬수 있음.

© 이대현

Raising Exceptions

```
raise Exception

raise Exception('This is Error Message’)

raise ValueError

raise ValueError('Too Young')
```

© 이대현

Exception Handling

```
# 1. 잘못된 입력: input 에 문자열을 입력 - ValueError 발생
# 2. 잘못된 입력: 나이가 적은 경우      - ValueError 발생

try:
    age = int(input('Enter Age:'))
    if age < 18:
        raise ValueError('Too young')

except Exception as err:
    print(f"{err=}, {type(err)=}")
```

```
err=ValueError("invalid literal for int() with base 10: 'asfdaf'"), type(err)=<class 'ValueError'>
```

© 이대현

print box

```
def print_box(symbol, width, height):
    print(symbol*width)
    for i in range(height-2):
        print(symbol+ (' ' * (width-2)) + symbol)
    print(symbol*width)

for sym, w, h in (('*', 4, 4), ('0', 20, 5), ('x', 1, 3), ('ZZ', 3, 3)):
    print_box(sym, w, h)
```

© 이대현

Exception raise 이용

```
def print_box(symbol, width, height):
    if len(symbol) != 1:
      raise Exception('Symbol must be a single character string.')
    if width <= 2:
      raise Exception('Width must be greater than 2.')
    if height <= 2:
      raise Exception('Height must be greater than 2.')

    print(symbol*width)
    for i in range(height-2):
        print(symbol+ (' ' * (width-2)) + symbol)
    print(symbol*width)

for sym, w, h in (('*', 4, 4), ('0', 20, 5), ('x', 1, 3), ('ZZ', 3, 3)):
    try:
        print_box(sym, w, h)
    except Exception as err:
      print('An exception happened: ' + str(err))
```

© 이대현

Traceback 의 이용

```
def f3():
    age = int(input('Enter Age:   '))
    if age < 18:
        print('You cannot enter.')

def f2():
    f3()

def f1():
    for i in range(5):
        f2()

f1()
```

```
Enter Age:   asd
Traceback (most recent call last):
  File "D:₩workCoding₩2024-Script-Language₩src₩LEC09-Debugging₩buggy.py", line 12, in <module>
    f1()
  File "D:₩workCoding₩2024-Script-Language₩src₩LEC09-Debugging₩buggy.py", line 10, in f1
    f2()
  File "D:₩workCoding₩2024-Script-Language₩src₩LEC09-Debugging₩buggy.py", line 6, in f2
    f3()
  File "D:₩workCoding₩2024-Script-Language₩src₩LEC09-Debugging₩buggy.py", line 2, in f3
    age = int(input('Enter Age:   '))
          ^^^^^^^^^^^^^^^^^^^^^^^^^^
ValueError: invalid literal for int() with base 10: 'asd'
```

© 이대현

파일 저장

```
import traceback

def f3():
    age = int(input('Enter Age:   '))
    if age < 18:
        print('You cannot enter.')

def f2():
    f3()

def f1():
    for i in range(5):
        try:
            f2()
        except:
            with open('error_stack.txt', 'a') as ef:
                for line in traceback.format_stack():
                    ef.write(line)

f1()
```

© 이대현

Assertion – 로직에 대한 중간 점검

```
data = 100

assert data == 100

assert data < 100
```

```
age = input('Enter Age: ‘)

# 이 시점에서 확신은? age 는 정수여야 함.
assert type(age) is int, "age should be integer"
```

© 이대현

Exception vs. Assertion

- **Exception**
  - 주로 사용자 입력에 따른 오류
  - 예상하지 못한 상황에 대한 처리를 못한 “버그”
  - 추후 버그 수정을 위해서, 코드 내에 exception 처리는 유지되고 실행됨.

- **Assertion**
  - 심각한, 중대한 버그.
  - 개발 단계에서, 반드시 해결해야 함.
  - 실행 속도를 저하시키기 때문에, release build 에서는 빠져야 함. -O 최적화 옵션.
    - python -O mycode.py

© 이대현

Logging

- **Print 대신 logging !!!!!**

```
import logging
# basic config works only once after logging
logging.basicConfig(
        level=logging.DEBUG,
        format=' %(asctime)s - %(levelname)s - %(message)s'
)
```

© 이대현

Logging levels

Level	Value	When to use
`DEBUG`	10	(주로 문제 해결을 할 때 필요한) 자세한 정보.
`INFO`	20	작업이 정상적으로 작동하고 있다는 확인 메시지.
`WARNING`	30	예상하지 못한 일이 발생하거나, 발생 가능한 문제점을 명시. (e.g. ‘disk space low’) 작업은 정상적으로 진행.
`ERROR`	40	프로그램이 함수를 실행하지 못 할 정도의 심각한 문제.
`CRITICAL`	50	프로그램이 동작할 수 없을 정도의 심각한 문제.

© 이대현

Logging levels

- 로깅 수준을 변경할 수 있음 – 높일 수 있음.
- 낮추는 것은 불가능.
- basicConfig 는 맨 처음 한번 설정이 가능.

```
logging.disable(logging.ERROR)
logging.warning('Warning Log')
logging.error('Error Log')
logging.critical('Critical Log')
```

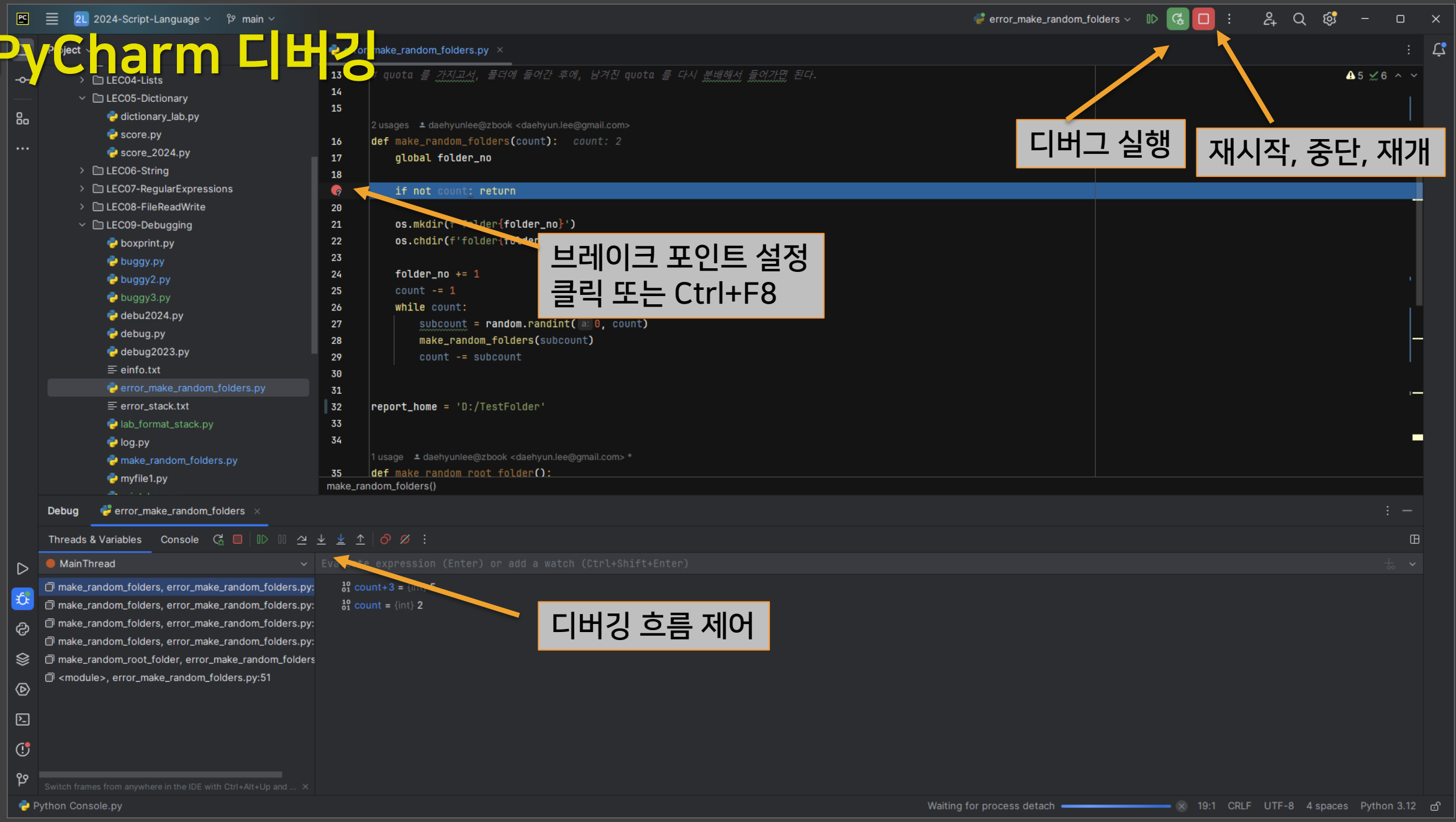
© 이대현

파일 로깅

```
import logging
logging.basicConfig(
        filename='myProgramLog.txt',
        level=logging.DEBUG,
        format='%(asctime)s - %(levelname)s - %(message)s'
)
```

© 이대현

PyCharm 디버깅



© 이대현

디버깅 흐름 제어

- Step Over
  - 한 라인을 실행, 함수를 통째로 실행

- Step Into “my code”
  - 함수 호출 안으로 들어감.

- Step Into
  - 파이썬 내부 라이브러리 함수 안으로도 들어감.

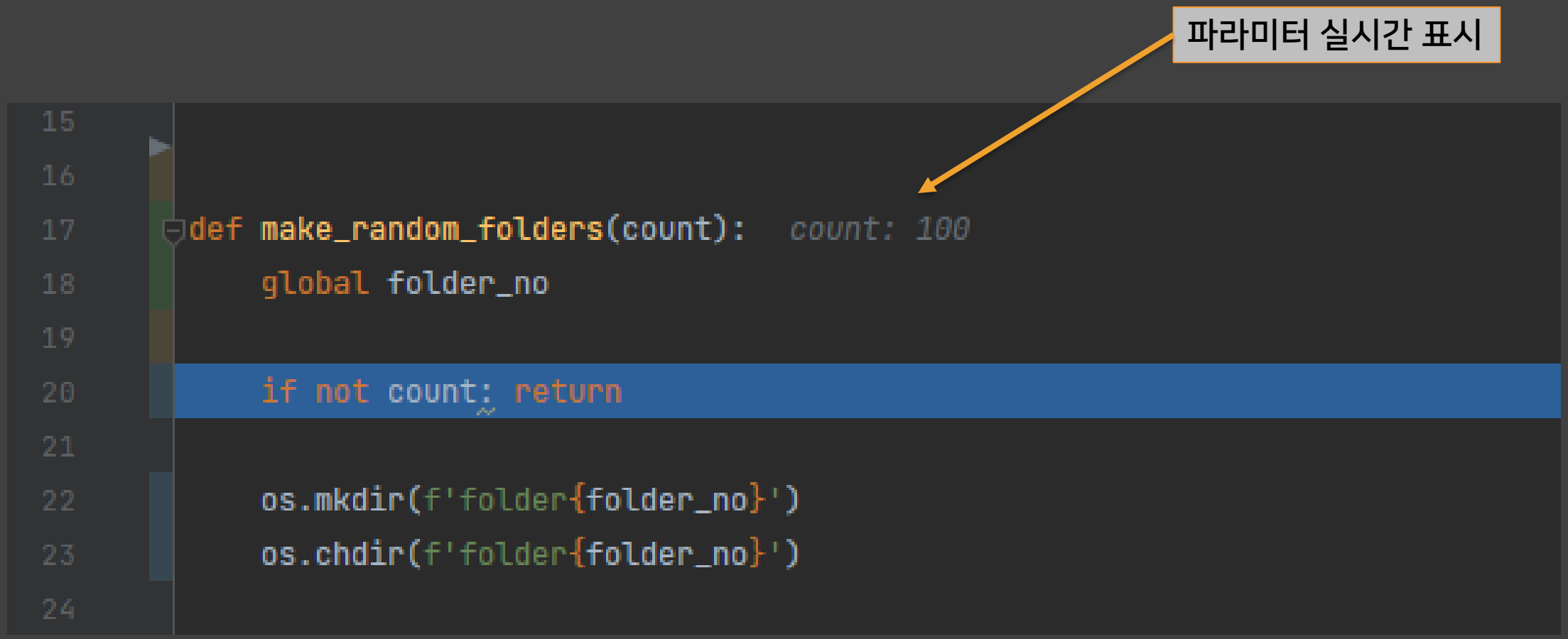
- Step Out : 함수 빠져나오기
- Run to Cursor : 커서 위치까지 실행

© 이대현

```python
import logging
logging.basicConfig(level=logging.DEBUG, format='%(filename)s(%(lineno)d): %(asctime)s - %(levelname)s - %(message)s')
```

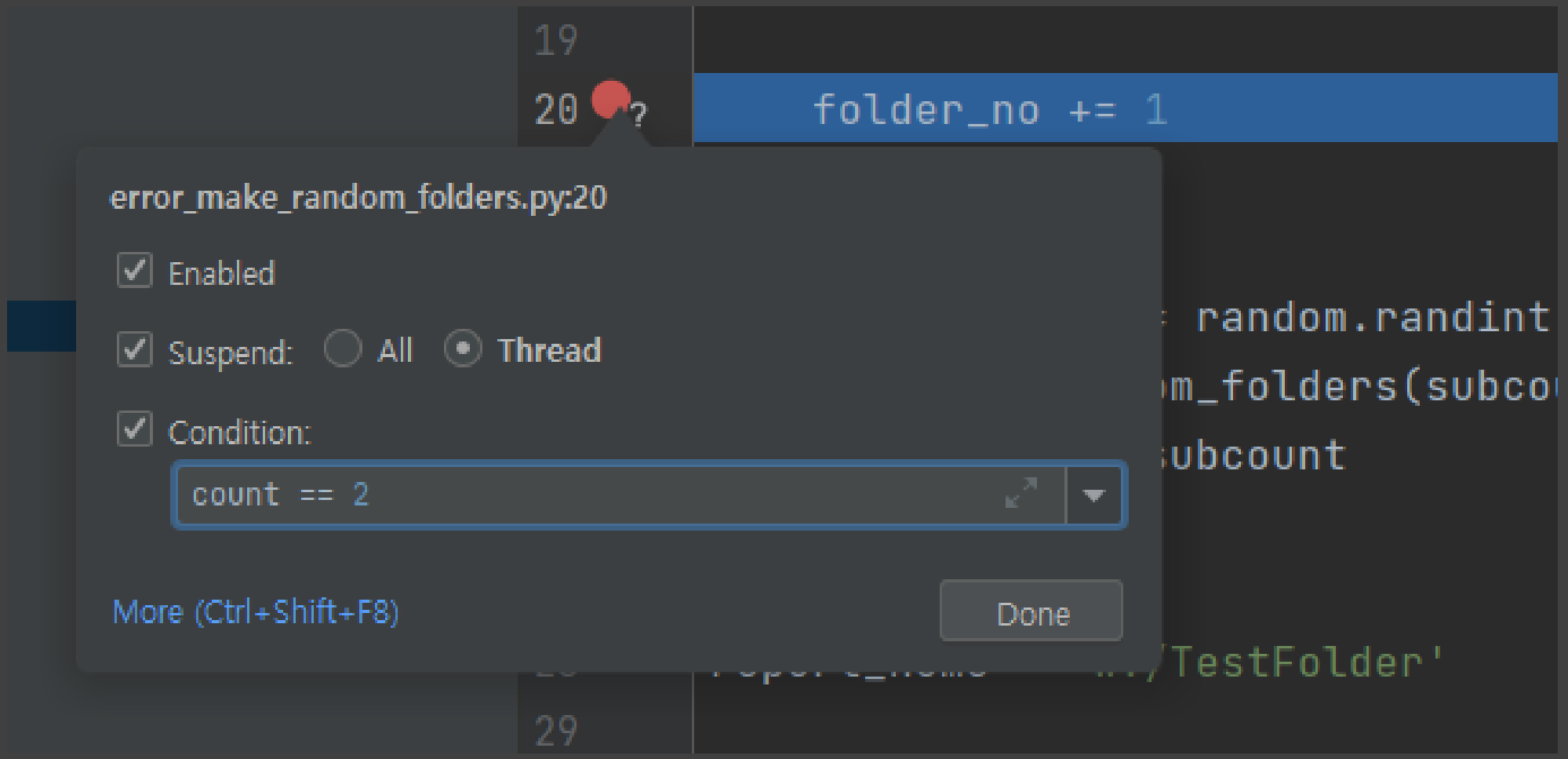
```python
def make_random_root_folder():
    assert not os.path.exists('root_folder'), 'root folder already exists'
    os.mkdir('root_folder')
    os.chdir('root_folder')
    make_random_folders(100)
    os.chdir('..')
```

© 이대현


파라미터 실시간 표시
15
16
17 def make_random_folders(count): count: 100
18 global folder_no
19
20 if not count: return
21
22 os.mkdir(f'folder{folder_no}')
23 os.chdir(f'folder{folder_no}')
24

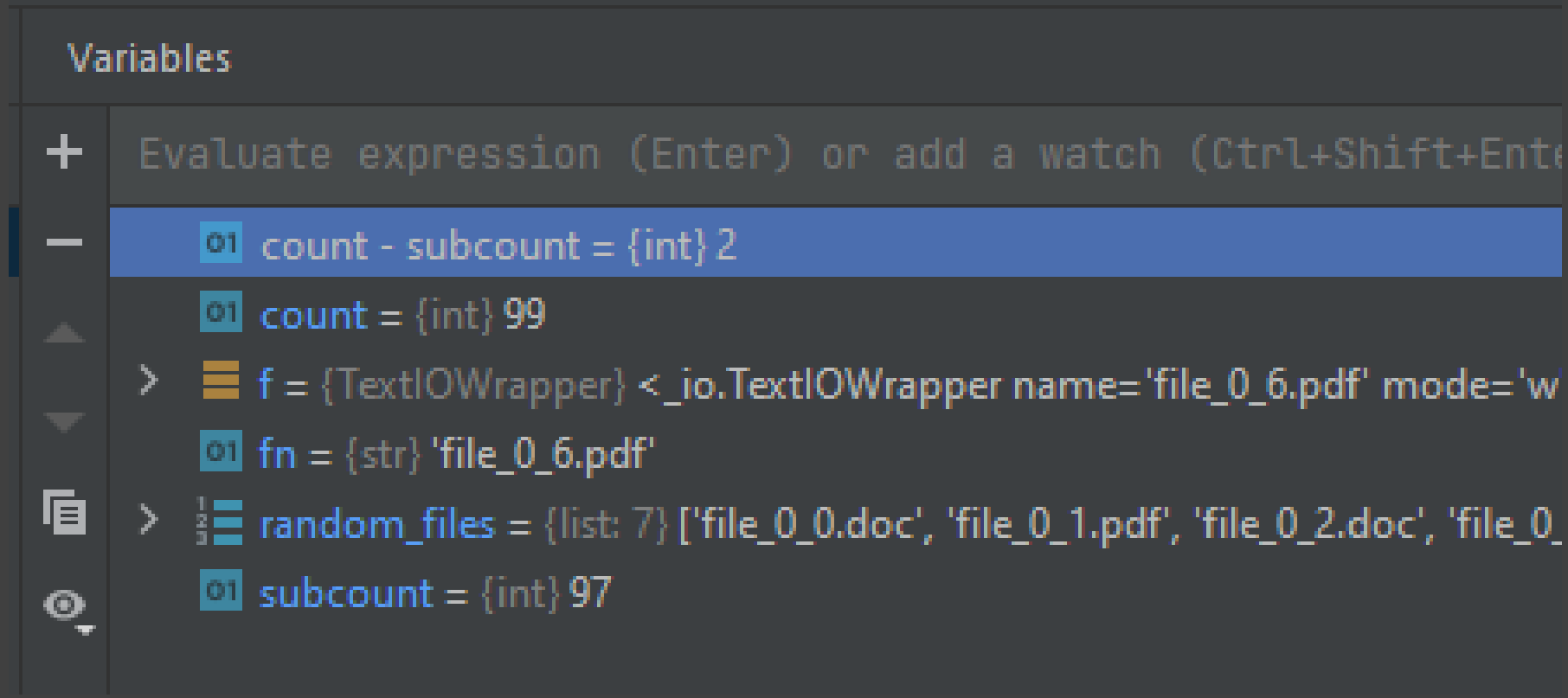
© 이대현

조건에 따른 브레이크 포인트



© 이대현

New Watch



© 이대현

가장 좋은 디버깅은?

- 버그가 없도록 짜는게 최선임.

- 비법?

```
while completed:
    code_a_single_line()
    test_the_line()
    debug_the_line()
    commit_the_line()
```

© 이대현