제10강 Debugging

학습 목차

- Exception
- Assertion
- Logging
- Pycharm Debugging

Raising Exceptions

```
raise ValueError

raise ValueError('Too young')

raise Exception

raise Exception('This is error message')
```

Exception Handling

```
try:
    age = int(input('Enter Age:'))
    if age < 18:
        raise Exception('Too Young’)

except Exception as err:
    print(err)
```

Traceback

myfile.py

```
def bacon():
    raise Exception('My Error')


def spam():
    bacon()


spam()
```

```
C:\Python\python.exe D:/workCoding/ScriptLanguage2021/src/LEC10-Debugging/myfile.py
Traceback (most recent call last):
  File "myfile.py", line 9, in <module>
    spam()
  File "myfile.py", line 6, in spam
    bacon()
  File "myfile.py", line 2, in bacon
    raise Exception('My Error')
Exception: My Error

Process finished with exit code 1
```

Traceback 결과를 저장

```
import traceback


def bacon():
    try:
        raise Exception('My Error')
    except:
        ef = open('einfo.txt', 'w')
        for line in traceback.format_stack():
            ef.write(line)
        ef.close()
```

Assertion

```
data = 100

assert data == 100

assert data < 100
```

```
age = input('Enter Age: ')
assert type(age) is int, "age should be integer"
```

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

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'
)
```

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.

Logging levels

- 로깅 수준을 변경할 수 있음.

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

파일 로깅

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

PyCharm 디버깅