

---

# ***클린코드 – 파이썬 part.1***

---

**순천향대학교  
컴퓨터시스템연구실**

**이인규  
23.01.16**

# 세미나 목표

- ◆ Python이나 Java 등 객체지향언어의 이해
- ◆ 깨끗한 소스코드를 작성하는 방법을 배운다.
- ◆ 객체지향언어의 디자인패턴 공부

디자인패턴이란?

코드를 재사용하기 좋은 형태로 특정 규약을 만들어서 정리한 것

# 레퍼런스

- ◆ 제목: 클린코드 이제는 파이썬이다.
- ◆ 지은이: 알 스웨이가트

## ◆ 전체 내용 요약

1장. 에러 메시지 파악, 환경 설정

2장. 클린 코드의 모범 사례, 도구, 기법

3장. 파이썬의 객체지향 프로그래밍

# 에러 메시지 파악

- ◆ 파이썬 프로그램은 코드가 try except 문으로 처리할 수 없는 예외가 발생하면 충돌(Crash)을 발생시킨다.
- ◆ 이때 파이썬은 에러 메시지와 함께 스택 추적(Stack trace)을 보여준다.

# 스택 추적의 예시

```
1  def a():
2      print('Start of a()')
3      b()
4
5  def b():
6      print('Start of b()')
7      c()
8
9  def c():
10     print('Start of c()')
11     42 / 0 # division error
12
13  a()
```

- ◆ a 함수 실행
- ◆ b 함수 실행
- ◆ c 함수 실행
- ◆ 42 / 0 에서 오류를 발생 시킨다.

# 스택 추적의 예시

```
C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15>python 2023-01-15-1.py
Start of a()
Start of b()
Start of c()
```

```
Traceback (most recent call last):
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 13, in <module>
    a()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 3, in a
    b()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 7, in b
    c()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 11, in c
    42 / 0 # division error
    ~~~~
ZeroDivisionError: division by zero
```

- ◆ 위와 파이썬 프로그램에서 오류가 발생했을 때, ZeroDivisionError와 함께 Traceback이 출력되는 것을 확인할 수 있다.

# 스택 추적의 예시

```
C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15>python 2023-01-15-1.py
```

```
Start of a()
```

```
Start of b()
```

```
Start of c()
```

```
Traceback (most recent call last):
```

```
File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 13, in <module>  
    a()
```

```
File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 3, in a  
    b()
```

```
File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 7, in b  
    c()
```

```
File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 11, in c  
    42 / 0 # division error
```

```
~~~~~
```

```
ZeroDivisionError: division by zero
```

- ◆ 처음으로 a()라는 함수가 호출되었다.
- ◆ <module>을 통해 이 행이 전역(global)에 위치해 있다는 것을 알려준다.

# 스택 추적의 예시

```
C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15>python 2023-01-15-1.py
Start of a()
Start of b()
Start of c()
Traceback (most recent call last):
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 13, in <module>
    a()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 3, in a
    b()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 7, in b
    c()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 11, in c
    42 / 0 # division error
    ~~~~
ZeroDivisionError: division by zero
```

- ◆ 그 다음 a 함수 내에서 b()가 호출되었고, b 함수 내에서 c 함수가 호출 되었다.



# 스택 추적의 예시

```
C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15>python 2023-01-15-1.py
Start of a()
Start of b()
Start of c()
Traceback (most recent call last):
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 13, in <module>
    a()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 3, in a
    b()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 7, in b
    c()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 11, in c
    42 / 0 # division error
      ~~~~
ZeroDivisionError: division by zero
```

- ◆ 마지막으로 c 함수에서  $42 / 0$  코드가 실행되었는데, 이때 오류가 발생했음을 알 수 있다.
- ◆ 소프트웨어 설계상 나누기 0은 계산할 수 없다.

# 스택 추적시 주의 할 점

```
C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15>python 2023-01-15-1.py
Start of a()
Start of b()
Start of c()
Traceback (most recent call last):
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 13, in <module>
    a()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 3, in a
    b()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 7, in b
    c()
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-1.py", line 11, in c
    42 / 0 # division error
    ~~~~
ZeroDivisionError: division by zero
```

- ◆ 이때 추적 정보가 제공한 행 번호는 파이썬이 에러를 최종적으로 감지한 곳이며, 진짜 원인은 이 앞에 있을 수 있다.

# 스택 추적시 주의 할 점

```
1  def spam(number1, number2):  
2      |   return number1 / (number2 - 42)  
3  
4      spam(101, 42)
```

# 스택 추적시 주의 할 점

```
1  def spam(number1, number2):  
2      |   return number1 / (number2 - 42)  
3  
4      spam(101, 42)
```

- ◆ 매개변수1과 2를 계산한 결과를 return하는 함수

# 스택 추적시 주의 할 점

```
C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15>python 2023-01-15-2.py
Traceback (most recent call last):
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-2.py", line 4, in <module>
    spam(101, 42)
  File "C:\Users\dldls\OneDrive\바탕 화면\CSLab Semina\세미나 준비용 파일\2023-01-15\2023-01-15-2.py", line 2, in spam
    return number1 / (number2 - 42)
           ~~~~~^~~~~~
ZeroDivisionError: division by zero
```

- ◆ 사실 위의 코드에서 오류가 감지된 행은 문제가 없지만, number2에 42라는 값이 들어왔을 때만 문제가 발생 된다.
- ◆ 결론적으로 문제는 스택 추적에서 알려준 행에서가 아닌 전역의 함수 호출에서 문제인 것이다.

```
1  def spam(number1, number2):
2      return number1 / (number2 - 42)
3
4  spam(101, 42)
```

# 린터(Linter)

- ◆ 소스 코드를 실행하지 않고, 분석해 잠재적인 에러를 경고하는 프로그램
- ◆ 정적 분석[실행하지 않고 분석하는 것]을 사용하기 때문에 모든 에러를 잡아내지는 못한다.
- ◆ 세탁기에 있는 섬유와 찌꺼기를 거르는 보풀 거름망(Lint trap)에서 유래되었다.

# VSCode에서의 린터 플러그인

**Pylance**

v2023.1.20

Microsoft

47,329,344

★★★★☆ (175)

A performant, feature-rich language server for Python in VS Code

Disable

Uninstall

Switch to Pre-Release Version



This extension is enabled globally.

Details

Feature Contributions

Changelog

Dependencies

Runtime Status

## Pylance

Fast, feature-rich language support for Python

Pylance is an extension that works alongside Python in Visual Studio Code to provide performant language support. Under the hood, Pylance is powered by [Pylint](#), Microsoft's static type checking tool. Using Pylint, Pylance has the ability to supercharge your Python IntelliSense experience with rich type information, helping you write better code faster.

Pylance is the default language support for [Python in Visual Studio Code](#) and is shipped as part of that extension as an optional dependency.

The Pylance name is a small ode to Monty Python's Lancelot who was the first knight to answer the bridgekeeper's questions in the Holy Grail.

세미나 준비용 파일 > 2023-01-15-1.py

```
1 print('Hello')
2 print('World')
```

PROBLEMS

1

OUTPUT

DEBUG CONSOLE

TERMINAL



2023-01-15-1.py 세미나 준비용 파일

1

⊗ "(" was not closed Pylance [Ln 1, Col 6]

# Question?



Please contact :

**이인규**  
**순천향대학교 컴퓨터학부**  
**멀티미디어관 M606**

Email : [dldlsrb1414@naver.com](mailto:dldlsrb1414@naver.com)