



13장. 예외 처리

파이썬 정복



Contents

❖ 목차

- 1. 예외 처리
- 2. 자원 정리



1. 예외 처리

❖ 예외 (Exception)

- 프로그램 코드는 이상이 없으나 실행 중에 불가피하게 발생하는 문제

exception

```
str = "89점"  
score = int(str)  
print(score)  
print("작업완료")
```

실행결과

```
Traceback (most recent call last):  
  File "C:/Python/test.py", line 2, in <module>  
    score = int(str)  
ValueError: invalid literal for int() with base 10: '89점'
```

- 에러 발생 직후 프로그램 종료되어 이후 명령 무시



1. 예외 처리

❖ 예외 처리

```
try:  
    실행할 명령  
except 예외 as 변수:  
    오류 처리문  
else:  
    예외가 발생하지 않을 때의 처리
```

- 간단하게 **try: except:** 까지만 사용하는 것도 가능

tryexcept

```
str = "89점"  
try:  
    score = int(str)  
    print(score)  
except:  
    print("예외가 발생했습니다.")  
print("작업완료")
```

실행결과

예외가 발생했습니다.
작업 완료

1. 예외 처리

- 예외 발생하지 않으면 `except` 블록 무시하고 다음 문장 실행
- 상황 자체를 해결하지는 않으나 예외 인식 / 처리 기회 제공
- 예외 처리 구문을 루프로 감싸면 무엇이 잘못되었는지 알려줌
 - 전체를 무한 루프로 감싸 예외 발생 시 다시 루프 선두로 돌아감



1. 예외 처리

❖ 예외의 종류

예외	설명
NameError	명칭이 발견되지 않는다. 초기화하지 않은 변수를 사용할 때 발생한다.
ValueError	타입은 맞지만 값의 형식이 잘못되었다.
ZeroDivisionError	0으로 나누었다.
IndexError	첨자가 범위를 벗어났다.
TypeError	타입이 맞지 않다. 숫자가 필요한 곳에 문자열을 사용한 경우 등이다.



1. 예외 처리

excepts

```
str = "89"  
try:  
    score = int(str)  
    print(score)  
    a = str[5]  
except ValueError:  
    print("점수의 형식이 잘못되었습니다.")  
except IndexError:  
    print("첨자 범위를 벗어났습니다.")  
print("작업완료")
```

실행결과

89

첨자 범위를 벗어났습니다.
작업완료

- `str[5]` → `IndexError` 예외 발생
- `except` 블록 아무리 많아도 먼저 발생한 예외에 맞는 하나만 선택됨
- 두 개 이상 괄호로 묶어 `except` 블록에서 동시에 받기 가능

```
except (ValueError, IndexError):  
    print("점수의 형식이나 첨자가 잘못되었습니다.")
```



1. 예외 처리

❖ raise 명령

- 고의적으로 예외 발생시킴
- 작업 위한 선결조건 맞지 않거나 문제 발생할 경우 호출원으로 사용

raise

```
def calcsun(n):  
    if (n < 0):  
        raise ValueError  
    sum = 0  
    for i in range(n+1):  
        sum = sum + i  
    return sum  
  
try:  
    print("~10 =", calcsun(10))  
    print("~-5 =", calcsun(-5))  
except ValueError:  
    print("입력값이 잘못되었습니다.")
```

실행결과

```
~10 = 55  
입력값이 잘못되었습니다.
```



1. 예외 처리

- 특이값 리턴하는 방식도 가능
 - 리턴값 점검에 철저할 것

errorret

```
def calcsun(n):  
    if (n < 0):  
        return -1  
    sum = 0  
    for i in range(n+1):  
        sum = sum + i  
    return sum  
  
result = calcsun(10)  
if result == -1:  
    print("입력값이 잘못되었습니다.")  
else:  
    print("~10 =", result)  
  
result = calcsun(-5)  
if result == -1:  
    print("입력값이 잘못되었습니다.")  
else:  
    print("~10 =", result)
```

실행결과

~10 = 55
입력값이 잘못되었습니다.



1. 예외 처리

- 예외 블록은 평이하게 작성하되 예외 발생 시에만 `except` 블록 점프
- 함수별로 비정상 상황 처리 방법 다름
- 예외 던지는 구문은 반드시 `try` 블록으로 감쌀 것



2. 자원 정리

❖ finally 블록

- 예외 발생 여부와 무관하게 반드시 실행해야 할 명령 지정

finally

```
try:
    print("네트워크 접속")
    a = 2 / 0
    print("네트워크 통신 수행")
finally:
    print("접속 해제")
print("작업 완료")
```

실행결과

```
네트워크 접속
접속 해제
Traceback (most recent call last):
  File "C:/Python/test.py", line 3, in <module>
    a = 2 / 0
ZeroDivisionError: division by zero
```



2. 자원 정리

❖ assert 문장

- 프로그램의 현재 상태가 맞는지 확인
- assert 조건, 메시지

assert

```
score = 128  
assert score <= 100, "점수는 100 이하여야 합니다"  
print(score)
```





Thank You !

파이썬 정보

