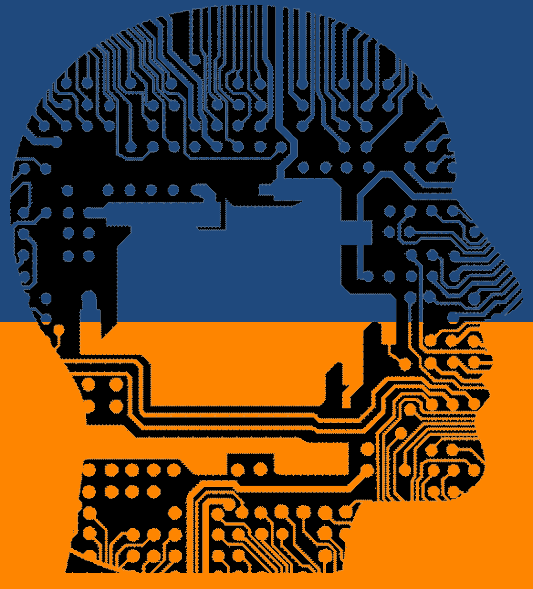
한국IT진흥부설
정보보호교육학원 아이섹

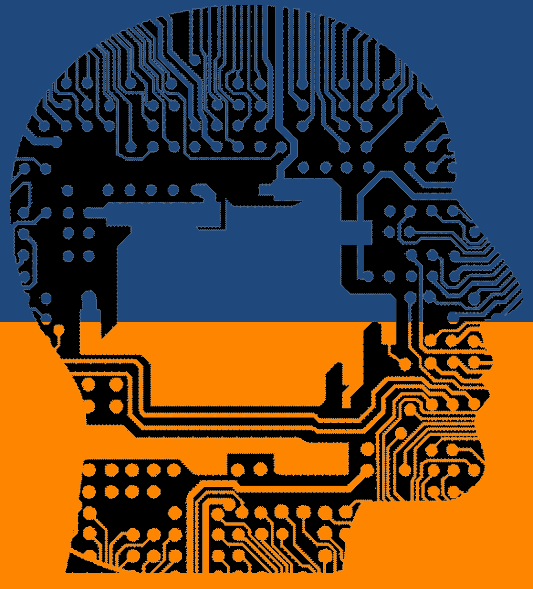


파이썬 기초

예외 처리

By SoonGu Hong

한국IT진흥부설
정보보호교육학원 아이섹



파이썬 기초

1. 예외란?

1-1. 예외(Exception)란?

- 예외(Exception) : 실행 중에 발견 된 오류

- 예외 처리(Exception Handling) : 오류 발생에 대한 대처 방법 중의 하나

- 예외 처리는 시스템 스스로 오류를 복구 하는 것이 아니고 오류발생 가능성이 있는 부분에 대한 처리를 미리 프로그래밍 하는 것

- 다음과 같은 상황들은 예외 처리를 해야 할 필요가 있음
    1. 파일을 다룰 때 파일이 없거나 쓰기 금지로 인한 오류
    2. 데이터베이스 프로그래밍 시 제약조건 등에 의한 데이터베이스 서버 오류
    3. 네트워크 프로그래밍 시 네트워크 연결 실패로 인한 오류
    4. 리스트 또는 튜플의 인덱스를 벗어난 참조로 인한 오류

1-2. 예외발생 예시

```
1  f = open('없는파일.txt', 'r')
```

```
---------------------------------------------------------------------------
FileNotFoundError                         Traceback (most recent call last)
<ipython-input-1-c0c01e0cc22c> in <module>()
----> 1 f = open('없는파일.txt', 'r')

FileNotFoundError: [Errno 2] No such file or directory: '없는파일.txt'
```

파일을 다룰 때 파일이 없거나 쓰기 금지로 인한 오류

```
1  4 / 0
```

```
---------------------------------------------------------------------------
ZeroDivisionError                         Traceback (most recent call last)
<ipython-input-2-87ac9c94a8c2> in <module>()
----> 1 4 / 0

ZeroDivisionError: division by zero
```

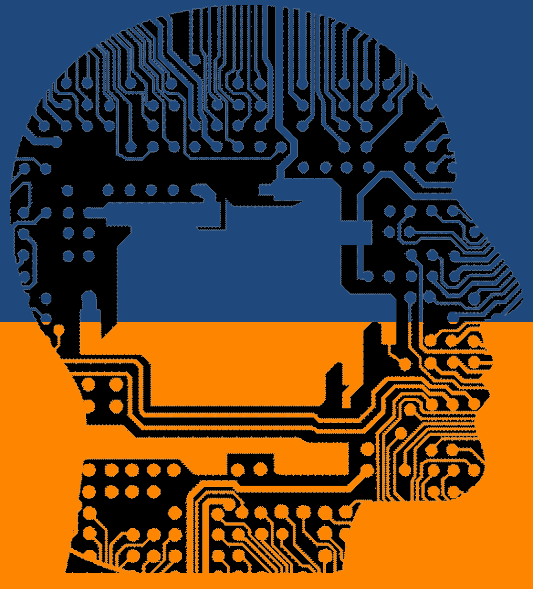
0으로 나누려는 오류

```
1  a = [1,2,3]
2  a[3]
```

```
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-3-47894dcd86df> in <module>()
      1 a = [1,2,3]
----> 2 a[3]

IndexError: list index out of range
```

리스트의 인덱스를 벗어난 참조 오류



파이썬 기초

2. try ~ except절

2-1. try ~ except

- try는 예외를 처리할 때 예외가 발생할 가능성이 있는 문장을 작성

* try 절(또는 블록)에서 예외가 발생하면 except 절을 찾음

- except 절은 예외가 발생했을 경우 실행되어야 할 코드를 작성

```
try :
    # 예외가 발생할 가능성이 있는 문장
except :
    # 예외가 발생했을 경우 실행할 문장
```

2-2. try ~ except 예시

- 유효한 정수가 입력되면 try 블록의 나머지 문장을 실행
- 입력한 값이 정수가 아니면 정수형으로 변환하는 도중 예외가 발생

```
while True:
    try:
        x = int(input('정수를 입력하세요: '))
        print('입력한 정수는 {}입니다.'.format(x))
        break
    except:
        print('유효한 정수가 아닙니다. 다시 시도하세요.')
```

```
정수를 입력하세요: hello
유효한 정수가 아닙니다. 다시 시도하세요.
정수를 입력하세요: 12345
입력한 정수는 12345입니다.
```

2-3. 여러 예외 처리하기

- try 블록에서 발생할 수 있는 예외가 여러 개일 경우 각각의 예외를 처리하도록 catch 블록을 정의해 놓을 수 있음

```
try :
    # 예외가 발생할 가능성이 있는 문장
except Error1 :
    # Error1이 발생했을 경우 실행할 문장
except Error2 :
    # Error2가 발생했을 경우 실행할 문장
```

2-4. 다중 예외처리 예시1

```
while True:
    try:
        x = int(input('정수를 입력하세요: '))
        print('입력한 정수는 {}입니다.'.format(x))
        print('100을 입력한 수로 나누면 {}입니다.'.format(100/x))
        break
    except ValueError:
        print('유효한 정수가 아닙니다. 다시 시도하세요.')
    except ZeroDivisionError:
        print('0으로 나눌 수 없습니다.')
```

```
정수를 입력하세요: 0
입력한 정수는 0입니다.
0으로 나눌 수 없습니다.
정수를 입력하세요: hello
유효한 정수가 아닙니다. 다시 시도하세요.
정수를 입력하세요: 20
입력한 정수는 20입니다.
100을 입력한 수로 나누면 5.0입니다.
```

2-5. 다중 예외처리 예시2

- except 절은 여러 예외를 괄호로 묶은 튜플로 지정 할 수 있음

```
except (Error1, Error2, ...):
    pass
```

```
while True:
    try:
        x = int(input('정수를 입력하세요: '))
        print('입력한 정수는 {}입니다.'.format(x))
        print('100을 입력한 수로 나누면 {}입니다.'.format(100/x))
        break
    except (ValueError, ZeroDivisionError):
        print('유효한 수가 아닙니다. 다시 시도하세요.')
```

2-6. 정리작업 finally

- 자원을 반납하는 코드를 작성
- 프로그램을 더 안정적으로 만듬
- 예외처리 시 사용하는 finally 블록을 이용
- 사전 정의된 정리작업을 위한 with를 이용

```
try:
    f = open('myfile.txt', 'r')
except FileNotFoundError as e:
    print(str(e))
else:
    data = f.read()
    print(data)
finally:
    f.close()
```

```
hello myfile
```

2-7. 정리작업 with

```
for line in open("myfile.txt"):
    print(line, end="")
```

print 구문을 실행한 후 불확실한 시간 동안 파일이 열려 있음

```
hello myfile
```

```
with open("myfile.txt") as f:
    for line in f:
        print(line, end="")
```

파일 f가 항상 닫힘

```
hello myfile
```

감사합니다
THANK YOU