Exception Handling

Created by 이 윤 준 (yoonjoon.lee@gmail.com)

May, 2019

When exception occurs?

```
>>> f = open("나없는파일", 'r')
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
FileNotFoundError: [Errno 2] No such file or directory: 'ㄴ
```

```
>>> 4 / 0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

```
>>> a = [1,2,3]
>>> a[4]
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: list index out of range
```

How to do with exceptions

try ... except

```
try:
    ...
except [발생 오류[as 오류 메시지 변수]]:
    ...
```

1. try, except만 쓰는 방법

```
try:
    ...
except:
    ...
```

2. 발생 오류만 포함한 except문

```
try:
    ...
except 발생 오류:
    ...
```

3. 발생 오류와 오류 메시지 변수까지 포함한 except문

```
try:
    ...
except 발생 오류 as 오류 메시지 변수:
    ...
```

```
try:
    4 / 0
except ZeroDivisionError as e:
    print(e)
```

try .. finally

try문에는 finally절을 사용할 수 있다. finally절은 try문 수행 도중 예외 발생 여부에 상관없이 항상 수행.

```
f = open('foo.txt', 'w')
try:
    # 무언가를 수행한다.
finally:
    f.close()
```

Multiple error handling

```
try:
    ...
except 발생 오류1:
   ...
except 발생 오류2:
   ...
```

```
try:
    a = [1,2]
    print(a[3])
    4/0
except ZeroDivisionError:
    print("0으로 나눌 수 없습니다.")
except IndexError:
    print("인덱싱 할 수 없습니다.")
```

```
try:
    a = [1,2]
    print(a[3])
    4/0
except ZeroDivisionError as e:
    print(e)
except IndexError as e:
    print(e)
```

```
try:
    a = [1,2]
    print(a[3])
    4/0
except (ZeroDivisionError, IndexError) as e:
    print(e)
```

Errors Avoidance

```
try:
    f = open("나없는파일", 'r')
except FileNotFoundError:
    pass
```

Exception generaton

```
class Bird:
    def fly(self):
        raise NotImplementedError
```

> `NotImplementedError`는 파이썬 내장 오류로, 꼭 작성해야 하는 부분이 구현되지 않았을 경우 일부러 오류를 발생시키고자 사용합니다.

Error definition

```
class MyError(Exception):
    pass
```

```
def say_nick(nick):
    if nick == '바보':
        raise MyError()
    print(nick)
```

```
say_nick("천사")
say_nick("바보")
```

```
천사
Traceback (most recent call last):
  File "...", line 11, in <module>
    say_nick("바보")
  File "...", line 7, in say_nick
    raise MyError()
__main__.MyError
```

```
try:
    say_nick("천사")
    say_nick("바보")
except MyError:
    print("허용되지 않는 별명입니다.")
```

```
try:
    say_nick("천사")
    say_nick("바보")
except MyError as e:
    print(e)
```

`print(e)`는 오류메시지가 아무것도 출력되지 않음

```
class MyError(Exception):
    def __str__(self):
        return "허용되지 않는 별명입니다."
```