예외 처리

- description : 예외 처리
- author : 도봉산핵주먹
- email : hylee@repia.com
- lastupdate : 2020-06-25

예외 처리

예제 코드

```
# Section10
 # 파이썬 예외처리의 이해

 # 예외 종류
# SyntaxError, TypeError, NameError, IndexError, ValueError, KeyError..
 # 문법적으로 에러가 없지만 코드 실행 프로세스에서 발생하는 예외 처리 중요

# linter : 코드 스타일, 문법 체크

################################
### SyntaxError : 잘못된 문법
################################
# print('test)
# print('Hello'))
# if True
#    pass
# a = 20; b = 30; a+ = b
# x => y


################################
### NameError : 참조 변수 없음
################################

a = 10
b = 15

# print(c)

################################
### ZeroDivisionError : 0 나누기 에러
################################

# print(10 / 0)
```

```
##############################
### IndexError : 인덱스 범위 오버
##############################

x = [10, 20, 30]

# print(x[1])
# print(x[3]) # 예외 발생
# print(x.pop())
# print(x.pop())
# print(x.pop())
# print(x.pop()) # 예외 발생
# print(x.pop(50)) # 예외 발생


##############################
### KeyError
##############################

dic = {'name': 'Kim', 'Age': 33, 'City': 'Seoul'}

# print(dic['hobby'])     # 키가 존재하지 않으면 예외
# print(dic.get('hobby')) # 안전


# 항상 예외가 발생하지 않을 것으로 가정하고 먼저 코딩
# 그 후 런타임 예외 발생 시 예외처리 권장(EAFP 코딩 스타일)


##############################
### AttributeError : 모듈, 클래스에 있는 잘못된 속성 사용시에 예외
##############################

import time

# print(time.time())
# print(time.month()) # 예외 발생

x = [1, 2, 3]
# print(x.append(4))
# print(x.add(10))

##############################
### ValueError : 참조 값이 없을 때 예외
##############################

x = [1, 5, 9]
```

```
# x.remove(5)
# print(x)

# x.remove(100)
# print(x) # 예외 발생

t = (10, 100, 1000)

print(t.index(100))
# print(t.index(7)) # 예외 발생

##############################
### FileNotFoundError
##############################

# f = open('test.txt') # 애외 발생

##############################
### TypeError : 자료형에 맞지 않는 연산을 수행 할 경우
##############################

x = [1, 2]
y = (1, 2)
z = 'test'

# print(x + y) # 예외 발생
# print(x + z) # 예외 발생
# print(y + z) # 예외 발생
# print(sum([1,2,3],10,1)) # 예외 발생

# print(x + list(y))
# print(x + list(z)

############################################################
# 예외 처리 기본
# try                 에러가 발생 할 가능성이 있는 코드 실행
# except 에러명1:   여러 개 가능(에러 처리)
# except 에러명2:
# else:               try 블록의 에러가 없을 경우 실행
# finally:            항상 실행
############################################################


# 예제1

name = ['Kim', 'Lee', 'Park']

try:
    z = 'Kim'  # 'Cho' 예외 발생
    x = name.index(z)
    print('{} Found it! {} in name'.format(z, x + 1))
```

```
except ValueError:
    print('Not found it! - Occurred ValueError!')
else:
    print('ok! else!')

print()

# 예제2

try:
    z = 'Kim'  # 'Cho' 예외 발생
    x = name.index(z)
    print('{} Found it! {} in name'.format(z, x + 1))
except:  # 모든 에러를 처리(Exception)
    print('Not found it! - Occurred ValueError!')
else:
    print('ok! else!')

print()

# 예제3

try:
    z = 'Kim'  # 'Cho' 예외 발생
    x = name.index(z)
    print('{} Found it! {} in name'.format(z, x + 1))
except Exception as e:
    print(e)  # 에러 내용 출력
    # pass # 임시로 에러 해결 시 예외 처리
else:
    print('ok! else!')
finally:
    print('ok! finally!')  # 무조건 수행 됨

print()

# 예제4
# 예외처리는 하지 않지만, 무조건 수행 되는 코딩 패턴

try:
    print('try')
finally:
    print('finally')

print()

# 예제5
# 예외 발생 : raise
# raise 키워드로 예외 직접 발생
```

```
try:
    a = 'Park'
    if a == 'Kim':
        print('Ok! pass')
    else:
        raise ValueError
except ValueError:
    print('Raise! Occurred ValueError')
except Exception:
    print('Occurred Exception')
else:
    print('ok! else!')

############################################################
# 예외 처리 기본
# try                 에러가 발생 할 가능성이 있는 코드 실행
# except 에러명1:   여러 개 가능(에러 처리)
# except 에러명2:
# else:               try 블록의 에러가 없을 경우 실행
# finally:            항상 실행
############################################################
```

실행 콘솔

```
1
Kim Found it! 1 in name
ok! else!

Kim Found it! 1 in name
ok! else!

Kim Found it! 1 in name
ok! else!
ok! finally!

try
finally

Raise! Occurred ValueError
```

Tip

도봉산핵주먹, python, 예외처리

From:
http://rwiki.repia.com/ - **문제를 잘 정의하는 것은 문제를 절반 해결한 것이다. - 2023.12**

Permanent link:
http://rwiki.repia.com/doku.php?id=wiki:ai:python:%EC%98%88%EC%99%B8%EC%B2%98%EB%A6%AC&rev=1593067843

Last update: **2022/03/10 19:52**