파이썬 프로그래밍

예외 처리

한국기술교육대학교
온라인평생교육원

▣ 예외 발생

1. Raise로 예외 발생하기

- 예외를 특정 상황 조건에서 raise 키워드를 통해 발생시킬 수 있다.
- 아래 예는 시퀀스 형 클래스를 설계할 때 인덱싱을 구현하는 __getitem__ 메소드에서 인덱스가 범위를 넘을 때 IndexError를 발생시킨다.

▪raise 키워드 → 예외를 임의로 발생시킬 수 있는 키워드
▪self.n = n → 인스턴스 객체가 내부적으로 n 값을 지님
▪__getitem__ → 인덱싱 연산에 대응되는 연산
▪ raise IndexError → 이미 존재하는 Error → 이것을 발생시키겠다
▪10이 되는 순간 if 절 조건을 만족하여 IndexError 발생
▪for~in 구문은 IndexError가 발생할 때까지만 수행
▪IndexError를 잡고 싶으면 try, except절 사용

▣ 예외 발생

1. Raise로 예외 발생하기

```
class SquareSeq:
    def __init__(self, n):
        self.n = n
    def __getitem__(self, k):
        if k >= self.n or k < 0 :
            raise IndexError # 첨자 범위를 벗어나면 IndexError 예외를 발생시킴
        return k * k
    def __len__(self):
        return self.n

s = SquareSeq(10)
print s[2], s[4]
for x in s:  # IndexError가 발생하는 시점까지 반복한다
    print x,
print s[20]  # 첨자 범위가 넘었다
```

```
4 16
0 1 4 9 16 25 36 49 64 81
---------------------------------------------------------------------------
IndexError                                Traceback (most recent call last)
<ipython-input-6-f362c0e14e3a> in <module>()
     13 for x in s:  # IndexError가 발생하는 시점까지 반복한다
     14     print x,
---> 15 print s[20]  # 첨자 범위가 넘었다

<ipython-input-6-f362c0e14e3a> in __getitem__(self, k)
      4     def __getitem__(self, k):
      5         if k >= self.n or k < 0 :
----> 6             raise IndexError # 첨자 범위를 벗어나면 IndexError 예외를 발생시킴
      7         return k * k
      8     def __len__(self):

IndexError:
```

▣ 예외 발생

2. 사용자 클래스 예외 정의 및 발생시키기

- 사용자 정의 예외 클래스를 구현하는 일반적인 방법은 Exception 클래스를 상속 받아 구현한다.
  - Exception 클래스의 서브 클래스 중 하나를 상속 받아도 된다.
- 사용자 정의 예외 발생 방법
  - 내장 예외 발생 방법과 동일하게 raise [클래스의 인스턴스] 와 같이 해당 예외 클래스의 인스턴스를 던진다.
- 사용자 정의 예외를 잡는 방법
  - except [클래스 이름] 과 같이 해당 예외 클래스 이름을 사용한다.
- 아래 예에서 except Big이 잡는 예외는 Big과 Small 이다.
  - 이유: Small은 Big의 하위 클래스이기 때문

▪사용자 정의 예외 클래스 → 파이썬이 내장하고 있는 것 이외의 클래스

▣ 예외 발생

2. 사용자 클래스 예외 정의 및 발생시키기

```
class Big(Exception):
    pass

class Small(Big):
    pass

def dosomething1():
    x = Big()
    raise x

def dosomething2():
    raise Small()

for f in (dosomething1, dosomething2):
    try:
        f()
    except Big:
        print "Exception occurs!"
```

```
Exception occurs!
Exception occurs!
```

▪슈퍼클래스가 Exception으로 예외 클래스 정의한 것 사용 가능
▪예외를 발생시키는 방법 = 예외를 정의하는 방법
▪내장예외 발생시키는 방법 = 내장예외 처리하는 방법
▪Big, Small은 학습자가 스스로 정의하는 예외 클래스
▪함수 호출 후 보니, 함수 안 raise 있으면 예외 발생 → 예외 catch

▣ 예외 발생

3. 예외값 전달하기

- raise 키워드 뒤에 예외와 함께, 추가 메시지를 함께 던질 수 있다.

```
def f():
    raise Exception, 'message!!!'

try:
    f()
except Exception, a:
    print a
```

```
message!!!
```

▪raise란 키워드 사용 방법 → 예외 클래스, 메세지

- 생성자 안에 넣어준 에러 메시지는 except 키워드 사용시에 두 번째 인자로 해당 메시지를 받을 수 있다.

```
a = 10
b = 0
try:
    if b == 0:
        raise ArithmeticError('0으로 나누고 있습니다.')
    a / b
except ArithmeticError, v:
    print v
```

```
0으로 나누고 있습니다.
```

▪ArithmeticError는 파이썬에 조재하는 에러
▪예외(클래스)를 통해서 생성자를 보는 것
▪생성자를 부르면서 문자열로 넣어주는 인자는 , 뒤로 받아낼 수 있음