LESSON

예외 처리(Exception Handling)

예외 처리(Exception Handling)란?

Python

- 프로그램을 작성할 때 다양한 오류를 만나게 된다.
- 오류의 종류
  - ① 프로그램 실행 전에 발행하는 오류 => **구문 오류(Syntax error)**
    ex)들여쓰기, 괄호의 짝,…
  - ② 프로그램 실행 중에 발행하는 오류 => **예외(Exception) 또는 런타임 오류 (Runtime error)**

예외 종류 알아보기

예외 종류(클래스)	의미
BaseException	최상위 예외 클래스
Exception	사용자 정의 예외 클래스의 슈퍼 클래스
EOFError	파일에서 더 이상 읽을 데이터가 없을 때
ModuleNotFoundError	import할 모듈이 없을 때
FileNotFoundError	파일이 존재하지 않을 때
IndexError	잘못된 인덱스 사용할 때
NameError	잘못된 이름(변수) 사용할 때
SyntaxError	문법 오류일 때
TypeError	계산하려는 데이터의 유형이 잘못되었을 때
ValueError	함수의 매개변수에 잘못된 값을 넘길 때
ZeroDivisionError	0으로 나눌 때

예외 클래스의 계층 구조

Python 

BaseException	모든 예외의 최상위 예외
├─ SystemExit	프로그램을 종료하는 명령이 실행되었을 때
├─ KeyboardInterrupt	Control-C 키가 입력되었을 때
└─ Exception	대부분의 예외의 상위 예외
├─ ArithmeticError	수의 연산과 관련된 문제
│ └─ ZeroDivisionError	수를 0으로 나누려 할 때
├─ AssertionError	assert 문에 의해 발생
├─ AttributeError	(모듈·클래스·인스턴스에서) 잘못된 속성을 가리킬 때
├─ EOFError	(파일에서) 읽어들일 데이터가 더이상 없을 때
├─ ImportError	모듈을 임포트할 수 없을 때
│ └─ ModuleNotFoundError	임포트할 모듈을 찾을 수 없을 때
├─ LookupError	잘못된 인덱스.키로 인덱싱할 때
│ ├─ IndexError	(시퀀스에서) 잘못된 인덱스로 인덱싱할 때
│ └─ KeyError	(매핑에서) 잘못된 키로 인덱싱할 때
├─ NameError	잘못된 이름(변수)을 가리킬 때
├─ OSError	운영 체제의 동작과 관련된 다양한 문제
│ ├─ ChildProcessError	하위 프로세스(프로그램이 실행한 외부 프로그램)에서 오류 발생
│ ├─ FileExistsError	이미 존재하는 파일.디렉터리를 새로 생성하려 할 때
│ ├─ FileNotFoundError	존재하지 않는 파일.디렉터리에 접근하려 할 때
│ ├─ IsADirectoryError	파일을 위한 명령을 디렉터리에 실행할 때
│ ├─ NotADirectoryError	디렉터리를 위한 명령을 파일에 실행할 때
│ ├─ PermissionError	명령을 실행할 권한이 없을 때
│ └─ TimeoutError	명령의 수행 시간이 기준을 초과했을 때
├─ RuntimeError	다른 분류에 속하지 않는 실행시간 오류
│ ├─ NotImplementedError	내용 없는 메서드가 호출되었을 때
│ └─ RecursionError	함수의 재귀 호출 단계가 허용한 한계를 초과했을 때
├─ SyntaxError	구문 오류
│ └─ IndentationError	들여쓰기가 잘못되었을 때
│ └─ TabError	들여쓰기에 탭과 스페이스를 번갈아가며 사용했을 때
├─ TypeError	연산·함수가 계산할 데이터의 유형이 잘못되었을 때
├─ ValueError	연산·함수가 계산할 데이터의 값이 잘못되었을 때
│ └─ UnicodeError	유니코드와 관련된 오류
└─ Warning	심각한 오류는 아니나 주의가 필요한 사항에 관한 경고

try, except문 / try, except, else, finally문

Python

사용법:

```
try:
    실행할 코드
except 예외_종류1:
    오류시 실행할 코드
except 예외_종류2:
    오류시 실행할 코드
```

사용법:

```
try:
    실행할 코드
except 예외_종류:
    오류시 실행할 코드
except:
    오류시 실행할 코드
else:
    오류가 아닐 때 실행할 코드
finally:
    무조건 실행되는 코드
```

예외 처리


```
# 0으로 나누는 오류를 예외 처리하기
try:
    x = int(input("number: "))
    y = 10/x
    print(y)
except:
    print("예외 발생. 0으로 나눔")
```

<실행결과>
number: 0
예외 발생. 0으로 나눔

```
# 오류 회피
try:
    x = int(input("number: "))
    y = 10/x
    print(y)
except:
    pass
```

<실행결과>
number: 0

Process finished with exit code 0

예외 처리


```
# 오류에 따라 다르게 처리하기
num1 = input("숫자1: ")
num2 = input("숫자2: ")

try:
    num1 = int(num1)
    num2 = int(num2)
    result = num1/num2
    print(result)
except ValueError:
    print("정수를 입력하세요.")
except ZeroDivisionError:
    print("0으로 나눌 수 없습니다.")
```

```
<실행결과>
숫자1: dk
숫자2: 9
정수를 입력하세요.
숫자1: 10
숫자2: 0
0으로 나눌 수 없습니다.
```

예외 처리

오픈할 파일을 입력받아서 있으면 내용을 출력하고 "프로그램을 종료합니다."를 출력.
파일이 없으면 "파일이 존재하지 않습니다"와 "프로그램을 종료합니다."를 출력

```
try:
    filename = input("오픈할 파일명 입력: ")
    file = open(filename, "r")
except FileNotFoundError:
    print(f"{filename} 파일이 존재하지 않습니다.")
else:
    print(file.read())
    file.close()
finally:
    print("프로그램을 종료합니다.")
```

[결과]
오픈할 파일명 입력: abc.txt
abc.txt 파일이 존재하지 않습니다.
프로그램을 종료합니다.