예외 처리

# try ~ except

## 형식1

|  | try:  실행할 코드  except:  예외가 발생했을 때 처리하는 코드 |
| --- | --- |

#### except에서 모든 예외처리를 하는 형식

## 형식2

|  | try:  실행할 코드  except 예외명:  예외가 발생했을 때 처리하는 코드 |
| --- | --- |

#### 특정예외만 처리하는 형식

* + ZeroDivisionError : 숫자를 0으로 나눠서 발생하는 에러
  + IndexError : 범위를 벗어난 인덱스에 접근하여 발생하는 에러
  + AttributeError : 모듈, 클래스의 잘못된 속성 사용시 발생하는 에러
  + NameError : 변수선언 없이 변수를 사용하여 발생하는 에러

## 형식3

|  | try:  실행할 코드  except 예외명 as 변수:  예외가 발생했을 때 처리하는 코드 |
| --- | --- |

#### 예외의 에러 메시지를 변수로 받아오는 형식

* 변수명은 주로 except의 첫글자인 e를 사용한다.

# else와 finally

## 형식

|  | try:  실행할 코드  except:  예외가 발생했을 때 처리하는 코드  else:  예외가 발생하지 않았을 때 실행할 코드  finally:  예외 발생 여부와 상관없이 항상 실행할 코드 |
| --- | --- |

#### else는 except 바로 다음에 와야 하며, except를 생략할 수 없다.

* finally는 예외 발생 여부와 상관없이 항상 코드를 실행하며, except와 else를 생략할 수 있다.

# 임의로 예외 발생시키기

숫자를 0으로 나눴을때나 리스트의 인덱스를 벗어난 경우와 같이 파이썬에서 이미 정의된 예외 이외에 개발자가 직접 예외를 발생시킬 수 있다.

## 형식

|  | try:  if 예외조건: # if문으로 예외 판단  raise Exception('예외메세지') # 예외를 발생시킴  except Exception as e:  print('예외가 발생했습니다.', e) # 위에서 정한 예외메세지가 출력됨 |
| --- | --- |

#### 개발자가 직접 예외를 발생시킬 때는 raise에 예외를 지정하고 에러 메시지를 정의한다.

# 예외클래스 만들기(사용자 정의 예외)

파이썬에서 이미 정의된 예외이외에 개발자가 직접 예외클래스를 정의할 수 있다.

## 형식

|  | class 예외이름(Exception):  def \_\_init\_\_(self):  super().\_\_init\_\_('에러메시지') |
| --- | --- |

#### Exception을 상속받아서 새로운 클래스를 만든다.

* \_\_init\_\_ 메서드에서 부모클래스의 \_\_init\_\_ 메서드를 호출하면서 에러 메시지를 정의한다.

### 예제] 16tryexcept.py









