ILE3-007 예외처리

프로그램 수행 시 발생하는 에러에 대하여 대응하는 방법

예외처리

```
>>> wrong_val = 'ten'
>>> integer_val = int(wrong_val)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ValueError: invalid literal for int() with base 10: 'ten'
```

```
>>> ret = 'Success'
>>> ret_msg = '결과값은 {} {} 입니다.'.format(ret)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
IndexError: Replacement index 1 out of range for positional args tuple
```

try / except

예외가 발생 하였을 경우 지정한 코드를 호출하는 구문

```
try:
    # do something
except:
    # ????
```

Example

```
wrong_val = 'ten'

try:
    integer_val = int(wrong_val)
except:
    print('정상적인 숫자가 아닙니다.')
```

무슨 에러인지 알고 싶을 때

except 뒤에 에러 종류를 넣어주면 됨 (여러개 사용 가능)
해당 에러 외를 처리하고 싶으면 마지막에 except를 한번 더 추가

```
try:
    #do something
except ErrorType:
    # ????
except:
    # ????
```

Example

```
wrong_val = 'ten'

try:
```

```
    ret_msg = '결과값은 {} {} 입니다.'.format(int(wrong_val))
except ValueError:
    print('정상적인 숫자를 입력해 주십시오.')
except IndexError:
    print('코드 작성을 제대로 해 주십시오')
except:
    print('%&#$^$%@$#%#@$?')
```

에러 내용까지 알고 싶을 때

에러 종류를 지정한 에러 종별에 `as [변수명]`을 추가

> 에러 종별을 지정하지 않은 경우는 사용 불가 -> Exception 사용

```
try:
    #do something
except ErrorType as e:
    # ????
    # print(e)
except Exception e:
    # ????
    # print(e)
```

Example

```
wrong_val = 'ten'

try:
    ret_msg = '결과값은 {} {} 입니다.'.format(int(wrong_val))
except ValueError as e:
    print('정상적인 숫자를 입력해 주십시오.')
    print(e)
except IndexError as e:
    print('코드 작성을 제대로 해 주십시오')
    print(e)
except Exception as e:
    print('%&#$^$%@$#%#@$?')
    print(e)
```

try / except / else

에러 안 났을때는??

try 내의 구문이 모두 예외없이 완료 되었을 때 else 구문 호출

> try - except - else 순서는 지켜야 합니다.

```
try:
    # do something
except:
    # ????
else:
    # good job
```

Example

```
wrong_val = '1'

try:
    ret_msg = '결과값은 {} {} 입니다.'.format(int(wrong_val), 2)
except ValueError as e:
    print('정상적인 숫자를 입력해 주십시오.')
    print(e)
except IndexError as e:
    print('코드 작성을 제대로 해 주십시오')
    print(e)
except Exception as e:
    print('%&#$^$%@$#%#@$?')
    print(e)
else:
    print(ret_msg)
    print('참 잘했어요')
```

try / (except) / (else) / finally

뭐가 어찌 됐든 다 끝났을 때는?

try -> except or else -> finally 순서대로 호출 됨

```
try:
    # do something
except:
    # ????
else:
    # good job
finally:
    # say goodbye
```

Example

```
wrong_val = '1'

try:
    ret_msg = '결과값은 {} {} 입니다.'.format(int(wrong_val), 2)

except ValueError as e:
    print('정상적인 숫자를 입력해 주십시오.')
    print(e)
except IndexError as e:
    print('배열 범위를 벗어났습니다.')
    print(e)
except Exception as e:
    print('%&#$^$%@$#%#@$?')
    print(e)
else:
    print('참 잘했어요')
    print(ret_msg)
finally:
    print('대충 빗자루 타고 손 흔들면서 날라가는 짤')
```

과제

간단한 나누기 프로그램을 만들어 봅시다.
첫번째 값과 두번째 값을 나눈 결과를 출력을 합니다.

```
첫번째 값: 15
두번째 값: 3
계산된 값: 5
```

그리고 입력값으로 아래의 값을 사용할 경우, 아래와 같이 예외가 처리되어야 합니다.

```
첫번째 값: 15
두번째 값: 0
0으로 나눌 수 없습니다.
```

```
첫번째 값: 15
두번째 값: Zero
숫자를 입력하여 주십시오
```