12. 예외처리

12.1 오류 이해하기

오류란?

프로그래밍 올바르게 동작하지 않는 현상

오류의 종류

- 구문 오류 (Syntax Error)
- 예외 (Exception) 와 런타임 오류(Runtime Error)
- 논리 오류 (logical Error)

구문 오류

프로그램 실행 전에 발생하는 오류

괄호 개수 안 맞음, 들여쓰기, 오타 등 문법 상의 오류

```
print("해결안됩니다.)
'''
SyntaxError: EOL while scanning string literal
'''
```

예외 혹은 런타임오류

프로그램을 실행하는 과정에서 발생하는 오류

실행 중 비정상적인 값이 발생되어 더 이상 진행이 불가능할 경우 발생

이 부분은 나중에 자세히 다룸

논리 오류

구문오류나 예외, 런타임 오류가 발생하지는 않았지만, 그릇된 결과가 발생한 것.

```
# 짝수를 찾는 함수
def function(n):
    if n /2 ==0:
        print("짝수입니다.")
    else:
        print("홀수입니다.")

function(4) #홀수입니다?
function(3) #홀수입니다
```

위의 예시는 실행과정에서 아무런 문제가 없다. 하지만 4가 홀수라는 결과를 도출한다.

이는 `function()` 함수에서 나머지를 찾는 연산자인 %가 아닌 / 연산자를 사용했기 때문이다.

고치면 아래와 같이 정상적으로 나옴.

```
# 짝수를 찾는 함수
def function(n):
    if n % 2 ==0:
        print("짝수입니다.")
    else:
        print("홀수입니다.")

function(4) #짝수입니다.
function(3) #홀수입니다
```

12.2 예외 처리하기

예외는 주로 (1) 조건문 사용 하거나, (2) try~except 구문을 사용한다.

조건문 사용해서 예외 처리하기

```
import math

a = input("정수를 입력하세요")
if a.isdecimal() == True:
    a = int(a)
    print("원의 반지름은 {}".format(a))
    print("원의 둘레는 {}".format(math.pi*2*a))
    print("원의 넓이는 {}".format(math.pi*a**2))
else:
    print("정수가 아닙니다.")
```

하지만 어느 상황에서 예외가 발생할지 알 수 없기 때문에 조건문으로는 모든 예외를 처리할 수 없다.

따라서 try~except구문을 사용한다.

try~except 구문 사용

```
import math
try:
    n = int(input("정수를 입력하세요"))
    n = int(n)
    print("원의 반지름은 {}".format(n))
    print("원의 둘레는 {}".format(math.pi*2*n))
    print("원의 넓이는 {}".format(math.pi*n**2))
except:
    print("정수를 입력하지 않으셨습니다.")
```

try~except~else 구문 사용

예외가 발생할 가능성이 있는 구문 만을 남기고 나머지는 except나 else 구문에 넣는다.

```
import math
try:
    n = int(input("숫자를 입력하세요"))
except:
    print("정수를 입력하지 않았습니다.")
else:
    n = int(n)
    print("원의 반지름은 {}".format(n))
    print("원의 둘레는 {}".format(math.pi*2*n))
    print("원의 넓이는 {}".format(math.pi*n**2))
```

finally 구문 사용

예외가 발생하든 발생하지 않든 무조건 실행이 되어야 하는 구문

```
import math
try:
    n = int(input("숫자를 입력하세요"))
except:
    print("정수를 입력하지 않았습니다.")
else:
    n = int(n)
    print("원의 반지름은 {}".format(n))
    print("원의 둘레는 {}".format(math.pi*2*n))
    print("원의 넓이는 {}".format(math.pi*n**2))
finally:
    print("프로그래밍 종료됩니다.")

'''
숫자를 입력하세요2
원의 반지름은 2
원의 둘레는 12.566370614359172
원의 넓이는 12.566370614359172
프로그래밍 종료됩니다.
'''

'''
숫자를 입력하세요글자
정수를 입력하지 않았습니다.
프로그래밍 종료됩니다.
'''
```

12.3 예외의 고급 처리하기

예외 객체란?

예외 상황 발생시 관련 정보가 객체에 저장된다.

객체 명	상황
IndexError	list의 index를 넘어갔을 때 list1 = [1,2,3,4] print(list1[100])
ValueError	1. 값을 변환할 수 없을 때 a = int("안녕하세요") 2. 참조값이 없을 때 arr = ['a','b','c','d'] print(arr.index('z'))
ZeroDivisionError	0으로 나누었을 때 print(10/0)
FileNotFoundError	파일을 찾을 수 없을 때
NameError	선언하지 않은 변수를 호출할 때 print(var)
TypeError	잘못된 타입을 전달했을 때 a = 1+"abc" # 더할 수 없는 잘못된 타입을 전달했을 때

```
# 리스트 인덱스 범위를 벗어난 경우
list1 = [1,2,3]

try:
    i = int(input())
    print(list1[i])
except IndexError as e:
    print("에러 발생했습니다.")
    print(e)

'''
6
에러 발생했습니다.
list index out of range
'''
```

```
# 예외 구문을 다르게 해서 예외 발생시키기
list1 = [1,2,3]
try:
    n = input()
    int_n = int(n)
    print(list1[int_n])
except IndexError as e:
    print("리스트 범위를 벗어났습니다.")
    print(e)
```

```
except ValueError as e:
    print("정수를 입력해야 합니다.")
    print(e)

'''
5
리스트 범위를 벗어났습니다.
list index out of range
'''

'''
v
정수를 입력해야 합니다.
invalid literal for int() with base 10: 'v'
'''
```

예외 강제로 실행시키기

```
while 1:
    n = input("정수를 입력하세요")
    for digit in n:
        if not digit in "0123456789":
            raise ValueError("정수가 아닙니다.")
    print("정수로 변환한 결과입니다. {}".format(int(n)))

'''
정수를 입력하세요2
정수로 변환한 결과입니다. 2
정수를 입력하세요글자
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
Input In [240], in <cell line: 2>()
      3 for digit in n:
      4     if not digit in "0123456789":
----> 5         raise ValueError("정수가 아닙니다.")
      6 n = int(n)
      7 print("정수로 변환한 결과입니다. {}".format(n))

ValueError: 정수가 아닙니다.
'''
```