7장. 예외처리, 파일 입출력

에러(Error)와 예외(Exception)

에러(Error)

- **구문(syntex) 오류** : 문법에 맞지 않거나 오타가 났을 경우 발생하는 오류
  IDE에서 실행 전에 알 수 있음

예외(Exception)

- **실행(runtime) 오류** : 문법적인 오류는 없지만 실행(run) 될 때 에러가 발생하는 것을 말한다.

  예) 파일을 읽어 사용하려는데 파일이 없는 경우,
  리스트 값을 출력하려는데 리스트 요소가 없는 경우 등..

  에러가 발생되면 프로그램의 동작이 중지 또는 종료된다

예외(Exceptions)

❖ python.org > Tutorial(자습서)

8.2. Exceptions

Even if a statement or expression is syntactically correct, it may cause an error when an attempt is made to execute it. Errors detected during execution are called *exceptions* and are not unconditionally fatal: you will soon learn how to handle them in Python programs. Most exceptions are not handled by programs, however, and result in error messages as shown here:

Copy

```
>>> 10 * (1/0)
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    10 * (1/0)
         ~^~
ZeroDivisionError: division by zero
>>> 4 + spam*3
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    4 + spam*3
        ^^^^
NameError: name 'spam' is not defined
>>> '2' + 2
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
    '2' + 2
    ~~~~^~~
TypeError: can only concatenate str (not "int") to str
```

예외(Exceptions)

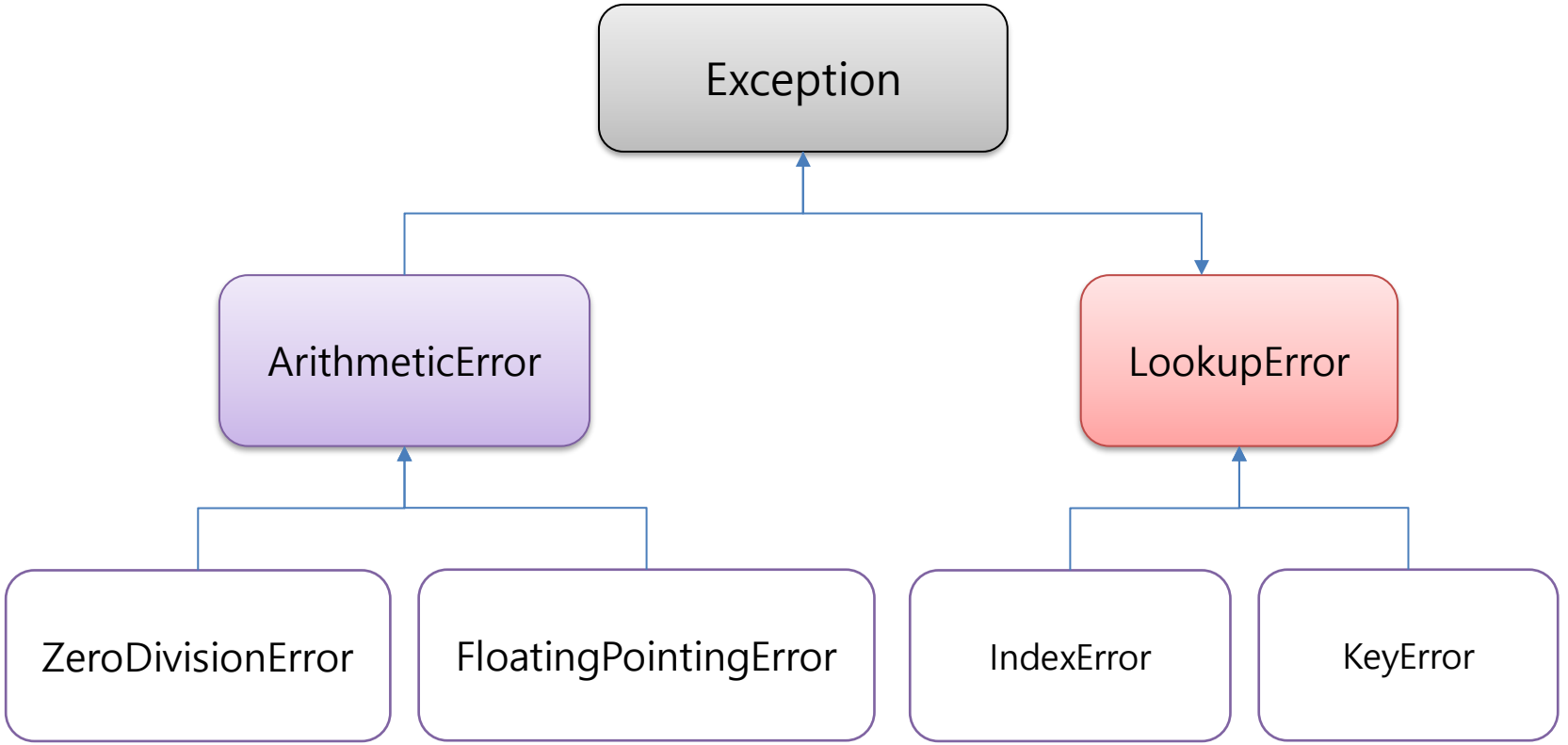
❖ 여러가지 예외 발생 코드

```
print('2' + 3) # TypeError 발생


# print(4 / 0)  # ZeroDivisionError 발생


# calculation = 10 + number  # NameError 발생
```

Exception 계층도



예외(Exceptions)

❖ **예외 처리 방법** : **try ~ except 구문**

try:

예외가 발생할 가능성이 있는 코드

except 예외 클래스:

예외가 발생했을 경우 실행 코드

예외(Exceptions)

- 예외 처리 코드

```
try:
    print('2' + 3)  # TypeError 발생
except TypeError:
    print("TypeError 예외가 발생했습니다.")

try:
    print(4 / 0)  # ZeroDivisionError 발생
except ZeroDivisionError:
    print("0으로 나눌 수 없습니다.")

try:
    calculation = 10 + number  # NameError 발생
except NameError:
    print("변수가 정의되지 않았습니다.")
```

예외(Exceptions)

- 숫자를 입력할 곳에 문자를 입력하여 예외 발생

```
숫자를 입력하세요 : ␣
Traceback (most recent call last):
  File "C:/pyworks/cho8/try_except/value_error.py", line 3,
    x = int(input("숫자를 입력하세요 : "))
ValueError: invalid literal for int() with base 10: '␣'
```

```
x = int(input("숫자를 입력하세요: "))
try:
    print(f"입력한 숫자는 {x}입니다.")
except ValueError:
    print("유효한 숫자를 입력하세요.")
```

예외(Exceptions)

숫자 추측 게임

- 예외 처리

```
import random

com = random.randint(1, 30) #컴퓨터의 난수
# print(com)

while True:
    x = input("맞혀보세요(입력: 1 ~ 30): ")
    guess = int(x) # 사용자가 추측한 수

    if guess < 1 or guess > 30:
        print("범위를 초과했어요. 다시 입력하세요")
    elif guess == com:
        print("정답!")
        break
    elif guess > com:
        print("너무 커요")
    else:
        print("너무 작아요")
```

다중 예외(Exceptions)

- 다중 try ~ except 구문

```
try:
    실행 코드
except 오류
    문제 발생시 실행코드
except 오류
    문제 발생시 실행코드
```

다중 예외(Exceptions)

- 다중 try ~ except 구문

```
try:
    data = [20, 10, 30, 50]
    print(data[4])  # IndexError 발생
    print(data[3] / 0)  # ZeroDivisionError 발생
except IndexError:
    print("리스트 인덱스가 범위를 벗어났습니다.")
except ZeroDivisionError:
    print("0으로 나눌 수 없습니다.")
```

예외 처리 미루기

➢ raise 구문

- raise 예외클래스("예외 메시지")
- 예외를 의도적으로 발생시킴

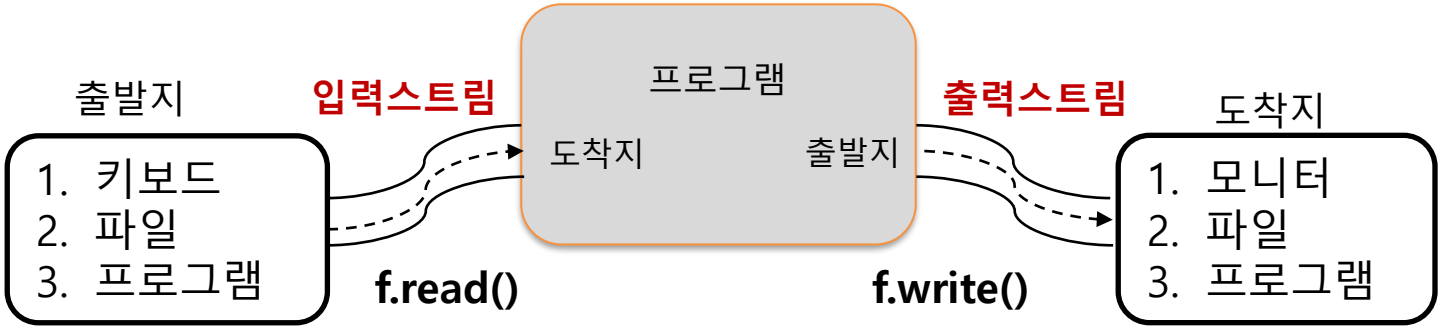
```
def check_id(id):
    if len(id) < 4:
        raise ValueError("ID는 4글자 이상이어야 합니다.")
    return id
try:
    userid = check_id("sky")
    print(f"ID: {userid}")
except ValueError as e:
    print(f"예외 발생: {e}")
```

입, 출력 스트림

- **스트림(stream)**

  자료흐름이 물의 흐름과 같다는 뜻이다. 입출력 장치는 매우 다양하기 때문에 프로그램 호환성이 떨어짐

  - 입력 스트림 – 동영상을 재생하기 위해 동영상 파일에서 자료를 읽을때 사용함
  - 출력 스트림 – 사용자가 쓴 글을 파일에 저장할 때는 출력 스트림 사용함

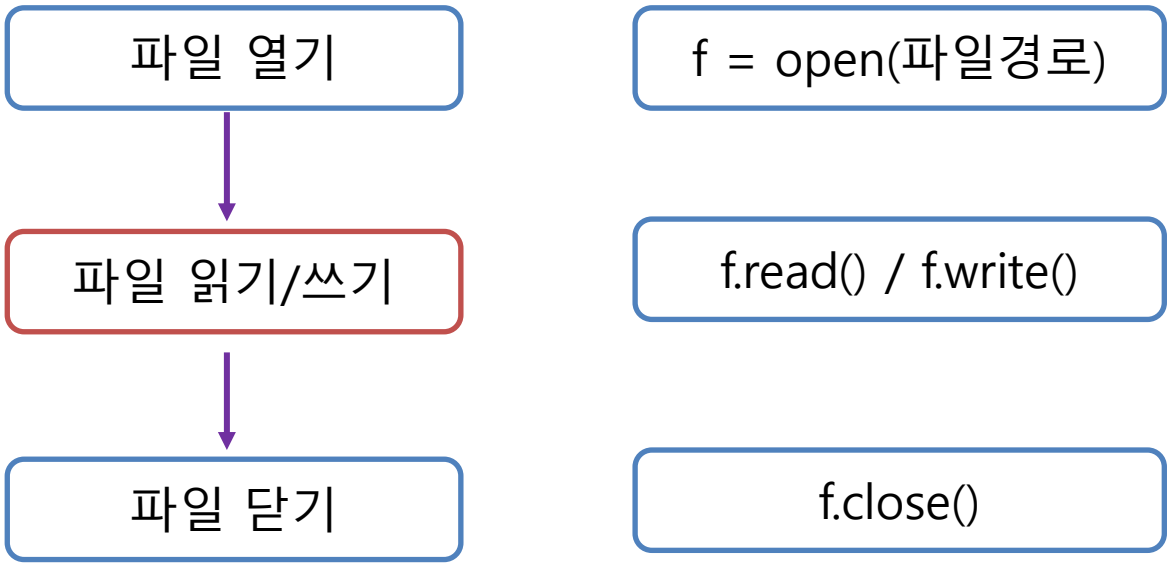


파일 입출력

- **파일 입출력의 필요성**

  프로그램 실행 중에 메모리에 저장된 데이터는 프로그램이 종료되면 사라진다.
  데이터를 프로그램이 종료된 후에도 계속해서 사용하려면 파일에 저장하고
  필요할 때 파일을 읽어서 데이터를 사용할 수 있다.

- **파일 입출력 프로세스**



파일 쓰기

- **파일 관련 주요 메서드**

메서드	모드	기능
open(파일, "w")	w	파일 열기(쓰기)
open(파일, "r")	r	파일 열기(읽기)
open(파일, "a")	a	파일 열기(추가 쓰기)
write()		파일 내용 쓰기
read()		파일 내용 읽기
close()		파일 닫기

파일 쓰기

- 파일 쓰기

```
# 파일 열기
f = open('c:/pyfile/file1.txt', 'w')

# 파일에 내용 쓰기
f.write("하늘\n")
f.write("첫 번째 줄입니다.\n")
f.write("cloud\n")
f.write("學生\n")
f.write("30\n")
f.write("3.14\n")

# 숫자 쓰기 불가
# f.write(100)  # TypeError 발생

# 파일 닫기
f.close()
```

file.txt

```
하늘
cloud
學生
30
3.14
```

파일 읽기

- 파일 읽기

```
# 파일 열기
# f = open('c:/pyfile/file1.txt', 'r')

# 파일 열기 예외 처리
try:
    f = open('c:/pyfile/file1.txt', 'r')
except FileNotFoundError:
    print("파일을 찾을 수 없습니다.")

# 파일 내용 읽기
content = f.read()
print("파일 내용:")
print(content)

# 파일 닫기
f.close()
```

파일 쓰기(추가모드)

- 파일 쓰기(추가 모드)

```
# 파일 열기 (추가 모드)
try:
    f = open('c:/pyfile/file1.txt', 'a')
except FileNotFoundError:
    print("파일을 찾을 수 없습니다.")

# 파일에 내용 추가하기
f.write("추가된 내용입니다.\n")
f.write("Hello, World!\n")

# 파일 닫기
f.close()
```

with ~ as 구문

- 자원누수 방지를 돕는 with ~ as 구문

  f.close()를 사용하지 않음

with open(파일이름) as 파일 객체:
코드 블럭

with ~ as 구문 예제

- 구구단 파일 쓰기

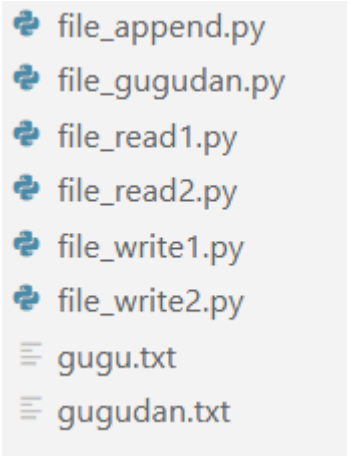
```
with open("gugudan.txt", 'w') as f:
    # 한 개의 단 출력
    """
    dan = 5
    for i in range(1, 10):
        f.write(f"{dan} x {i} = {dan * i}\n")
    """

    # 구구단 전체
    for i in range(2, 10):
        for j in range(1, 10):
            gugudan = f"{i} x {j} = {i * j}\n"
            f.write(gugudan)
        f.write("\n") #단별 줄바꿈
```

with ~ as 구문 예제

- 구구단 파일 읽기

```
with open("gugudan.txt", "r") as f:
    gugudan = f.read()
    print(gugudan)
```




```
2 x 1 = 2
2 x 2 = 4
2 x 3 = 6
2 x 4 = 8
2 x 5 = 10
2 x 6 = 12
2 x 7 = 14
2 x 8 = 16
2 x 9 = 18

3 x 1 = 3
3 x 2 = 6
3 x 3 = 9
3 x 4 = 12
3 x 5 = 15
3 x 6 = 18
3 x 7 = 21
3 x 8 = 24
3 x 9 = 27
```

with ~ as 구문 예제

- 파일에 리스트 자료 쓰고 읽기

```
# 파일에 장바구니 항목 쓰기
# 리스트 생성
carts = ["라면", "달걀", "우유", "치즈"]

# 파일에 쓰기
with open("carts.txt", "w") as file:
    for item in carts:
        file.write(item + "\n")

# 파일 닫기 (with 구문 사용으로 자동 닫힘)
print("carts.txt 파일에 작성되었습니다.")

# 파일에서 장바구니 항목 읽기
with open("carts.txt", "r") as file:
    print("장바구니 항목:")
    # content = file.read()

    for line in file:
        print(line.strip()) # 줄바꿈 문자 제거하여 출력
```

```
장바구니 항목:
라면
달걀
우유
치즈
```

영어 타자 연습 프로그램

● 영어 타자 게임

게임 방법

- 파일 쓰기를 이용하여 word.txt 파일을 생성한다.
- 게임이 시작되면 영어 단어가 화면에 표시된다.
- 사용자는 최대한 빠르고 정확하게 입력해야 한다.
- 바르게 입력했으면 다음 문제로 넘어가고 "통과"를 출력한다.
- 오타가 있으면 '오타! 다시 도전!'이 출력되고 같은 단어가 한 번 더 나온다.
- 타자 게임 시간을 측정한다.

영어 타자 연습 프로그램

- **영어 타자 게임**

```
[타자 게임]준비되면 엔터!

-문제 1
grape
grape
통과!
-문제 2
potato
potata
오타! 다시 도전!
-문제 2
potato
potato
통과!
-문제 3
grape
```

```
-문제 9
tree
tree
통과!
-문제 10
garlic
garlic
통과!
타자 시간 : 34.46초
```

리스트 랜덤 출력

- **word.txt 파일 만들고, 랜덤 추출하기**

```
import random

with open("word.txt", 'w') as f:
    word = ['sky', 'earth', 'moon', 'flower', 'tree',
            'strawberry', 'grape', 'garlic','onion', 'potato']
    for i in word:
        f.write(i + ' ')

# 단어를 랜덤 추출
with open("word.txt", 'r') as f:
    data = f.read().split()
    word = random.choice(data)
    print(word)
```

영어 타자 연습 프로그램

- 영어 타자 게임

```
import random
import time

try:
    # 파일 읽기
    with open("word.txt", "r") as f:
        word = f.read().split()
        #print(word)

    n = 1 #문제 번호

    print("[타자 게임] 준비되면 엔터")
    input()

    start = time.time()
```

영어 타자 연습 프로그램

- 영어 타자 게임

```
    while n < 11:
        print("\n문제", n)
        q = random.choice(word)
        print(q)  #문제 출제

        u = input() #사용자 입력
        if q == u:
            print("통과!")
            n += 1
        else:
            print("오타! 다시 도전!")

    end = time.time()
    et = end - start
    print(f"게임 소요 시간: {et:.2f}")

except FileNotFoundError:
    print("파일을 찾을 수 업습니다.")
```

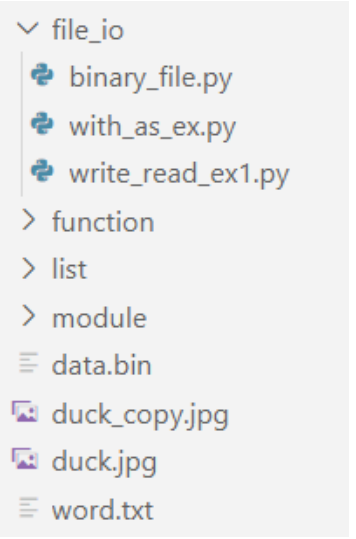
바이너리 파일 읽고 쓰기

◆ 바이너리 파일

바이너리 파일이란 화상, 음성 등의 대부분의 파일로 0과 1로 이루어진 파일이다.

open("파일 위치", 'wb')

모드	설명
wb	쓰기
rb	읽기



바이너리 파일 읽고 쓰기

◆ 바이너리 파일

```
# 바이너리 파일 읽기/쓰기 예제
with open("data.bin", "wb") as f:
    # 정수 데이터를 바이너리 형식으로 쓰기
    for i in range(1, 6):
        # 4바이트 정수로 변환하여 쓰기
        f.write(i.to_bytes(4, byteorder='little'))

with open("data.bin", "rb") as f:
    print("바이너리 파일에서 읽은 정수들:")
    while True:
        bytes_read = f.read(4)
        if not bytes_read:
            break
        # 4바이트 정수를 다시 정수로 변환
        number = int.from_bytes(bytes_read, byteorder='little')
        print(number)
```

```
바이너리 파일에서 읽은 정수들:
1
2
3
4
5
```

바이너리 파일 읽고 쓰기

◆ 이미지 복사하기

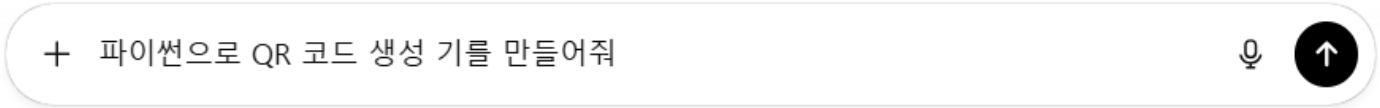
이미지 파일 읽어와서 다른 이름으로 쓰기

```
# 이미지 파일 읽기
with open("duck.jpg", "rb") as f1:
    data = f1.read()

# 이미지 파일 쓰기
with open(".duck_copy.jpg", "wb") as f2:
    f2.write(data)
```

QR 코드 생성기

- **챗GPT 프롬프트 입력**



- qrcode 모듈 설치하기

  pip install grcode

- qr 코드 스캔

  ☞ https://github.com/kiyongee2

QR 코드 생성기

```python
import qrcode

# QR 코드에 담을 데이터
# data = "Hello, this is a QR code!"
data = "https://github.com/kiyongee2"

# QR 코드 생성
img = qrcode.make(data)

# 생성된 QR 코드 이미지 저장
img.save("qrcode.png")
print("QR 코드가 'qrcode.png' 파일로 저장되었습니다.")

# QR 코드 표시
img.show()
```

pickle 모듈

◎ **pickle** 모듈

- 객체의 형태를 그대로 유지하면서 파일에 저장하고 불러올 수 있는 모듈이다.
- 이때 객체란, 리스트나 딕셔너리등의 자료구조도 포함한다.

모드	설명
pickle.dump	쓰기
pickle.load	읽기

pickle 모듈

```
import pickle

try:
    with open("object.dat", "wb") as f:
        lis = ['강아지', '고양이', '닭']
        dic = {1: '강아지', 2: '고양이', 3: '닭'}
        pickle.dump(lis, f)
        pickle.dump(dic, f)
except FileNotFoundError:
    print("파일을 찾을 수 없습니다.")


try:
    with open("object.dat", "rb") as f:
        lis = pickle.load(f)
        dic = pickle.load(f)
        print(lis)
        print(dic)
except FileNotFoundError:
    print("파일을 찾을 수 없습니다.")
```

```
['강아지', '고양이', '닭']
{1: '강아지', 2: '고양이', 3: '닭'}
```

실습 문제 – 파일 입출력

실행 결과가 표시되도록 코드 작성란을 완성하세요.

☞ 실행 결과

```
봄
여름
가을
겨울
```

```
seasons = ["봄", "여름", "가을", "겨울"]

with open("season.txt", "w", encoding='utf-8') as f1:

    # 코드 작성
```

```
with open("season.txt", "r", encoding='utf-8') as f2:

    #코드 작성
```