

14주차 파이썬 기초 프로그래밍

File, 예외 처리



수업 내용

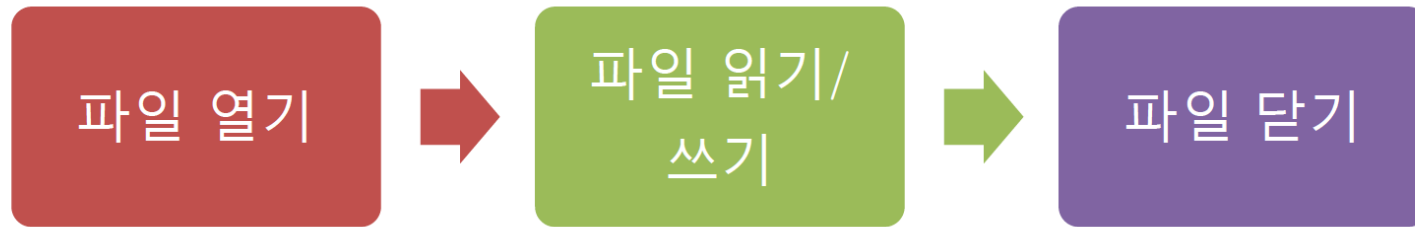
- 파일 기초 : open, read/write, close
- 파일 입출력
 - 텍스트 파일
 - 이진 파일
 - 임의 접근
 - 객체 입출력 : pickle 모듈
- 예외 처리
 - try, except, else, finally

파일 종류

- 텍스트 파일
 - ASCII 코드나 유니코드로 이루어져서 메모장으로 읽을 수 있는 파일.
 - 연속적인 줄(line)로 구성. 각 줄은 줄 바꿈 문자('\n')로 끝남
- 이진 파일
 - 0과 1로 구성된 데이터를 저장한 파일. 특정 프로그램에 의해서만 읽을 수 있음
 - 예 : 실행 파일, 사운드 파일, 이미지파일

파일 열기와 닫기

- 파일 사용 프로세스



- 파일 열기

- `f = open('input.txt', 'r')`

파일 모드 지정
생략하면 기본 값은 'r'

- 파일 닫기

- `f.close()`

파일 모드

| 파일 모드 | 설명 |
|-------|-------------------------------|
| r | 읽기 전용 |
| w | 쓰기 전용. 만약 파일이 존재하면 기존의 내용은 삭제 |
| a | 파일의 끝에 추가. 파일이 없으면 생성한다 |
| r+ | 파일에 읽고 쓸 수 있는 모드. |
| w+ | 읽고 쓸 수 있는 모드. 기존 파일은 삭제 |
| a+ | 파일 끝에 추가. 읽기도 가능 |
| rb | 이진 파일 읽기 전용 |
| wb | 이진 파일 쓰기 전용 |
| ab | 이진 파일 끝에 추가. 쓰기 전용 |
| rb+ | 이진 파일에 읽고 쓰기 |
| wb+ | 이진 파일 읽고 쓰기. 기존 파일은 삭제 |
| ab+ | 이진 파일 끝에 추가. 읽기도 가능 |

텍스트 파일 읽기

- 실습용 데이터 파일 만들기 : phone.txt

```
홍길동 010-1234-5678  
김철수 010-2345-6789  
이영희 010-3456-7890
```

- 파일 읽기

```
>>> infile = open('phone.txt')  
>>> s = infile.read(10)  
>>> s  
'홍길동 010-12'  
>>> infile.close()
```

- 한 줄 씩 읽기 : f.readline()

```
infile = open('phone.txt', 'r')
line = infile.readline()
while line != '':
    print(line)
    line = infile.readline()
infile.close()
```

```
infile = open('phone.txt', 'r')
for line in infile :
    line = line.rstrip()
    print(line)
infile.close()
```

홍길동 010-1234-5678

김철수 010-1234-5679

김영희 010-1234-5680

홍길동 010-1234-5678

김철수 010-1234-5679

김영희 010-1234-5680

텍스트 파일 쓰기

```
outfile = open('phones.txt', 'w')
```

```
outfile.write ('홍길동 010-1234-5678')
```

```
outfile.write ('김철수 010-2345-6789')
```

```
outfile.write ('이영희 010-3456-7890')
```

```
outfile.close()
```

```
import os.path
```

```
if os.path.isfile('phones.txt'):
```

```
    print('동일한 이름의 파일이 존재합니다')
```

```
else :
```

```
    outfile = open('phones.txt', 'w')
```

```
    outfile.write('홍길동 010-1234-5678')
```

```
    outfile.write('김철수 010-2345-6789')
```

```
    outfile.write('이영희 010-3456-7890')
```

```
outfile.close()
```


실습 01

- 텍스트 파일에 상점의 매출이 저장되어 있을 때, 이 파일을 읽어서 평균 매출과 총 매출을 계산한 후에 다른 파일에 출력하는 프로그램을 작성 하시오.

sales.txt

```
1000000  
1000000  
2000000  
500000  
1500000
```

summary.txt

```
총매출 = 6000000  
평균 일매출 = 1200000.0
```

```
# 매출 파일 처리
infilename = input('매출 입력 파일 이름 : ')
outfilename = input('매출 출력 파일 이름 : ')

# 입출력 파일 열기
infile = open(infilename, 'r')
outfile = open(outfilename, 'w')

# 변수 설정
sum = 0
count = 0

# 입력 파일에서 한 줄씩 읽어서 합계 계산
for line in infile :
    dailySale = int(line)
    sum += dailySale
    count += 1

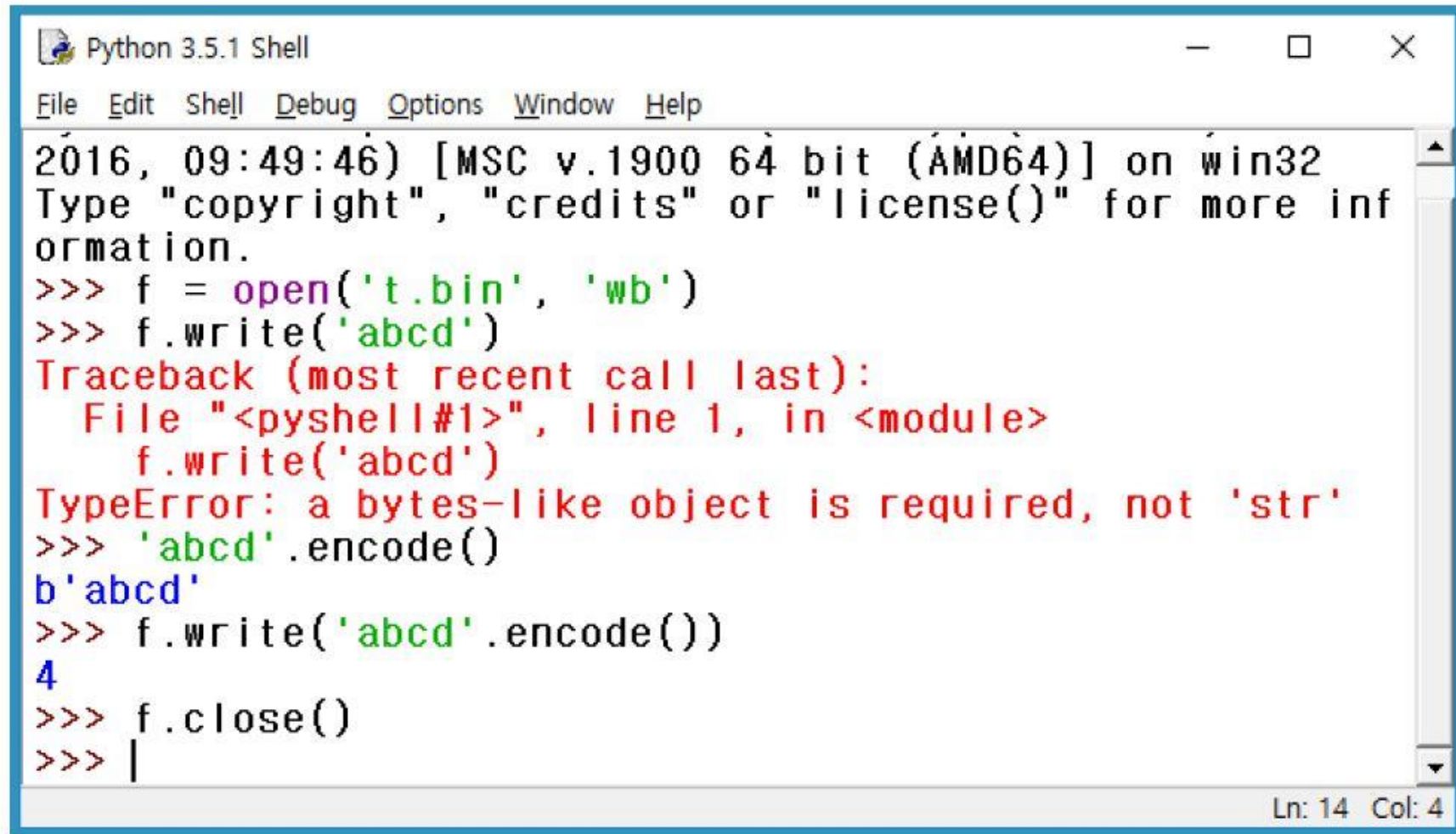
# 총 매출과 일평균 출력
outfile.write('총매출 = ' + str(sum) + '\n')
outfile.write('평균 일매출 = ' + str(sum/count))

infile.close()
outfile.close()
```

이진 파일

- 123을 파일에 저장할 때
 - 텍스트 파일 : '1'. '2'. '3' ASCII 코드 값에 저장
 - 이진 파일 : 123의 이진 수 저장
- 장점
 - 효율성
- 단점
 - 파일 내용을 확인하기 위해서는 데이터 저장 형식을 알고 있어야 함

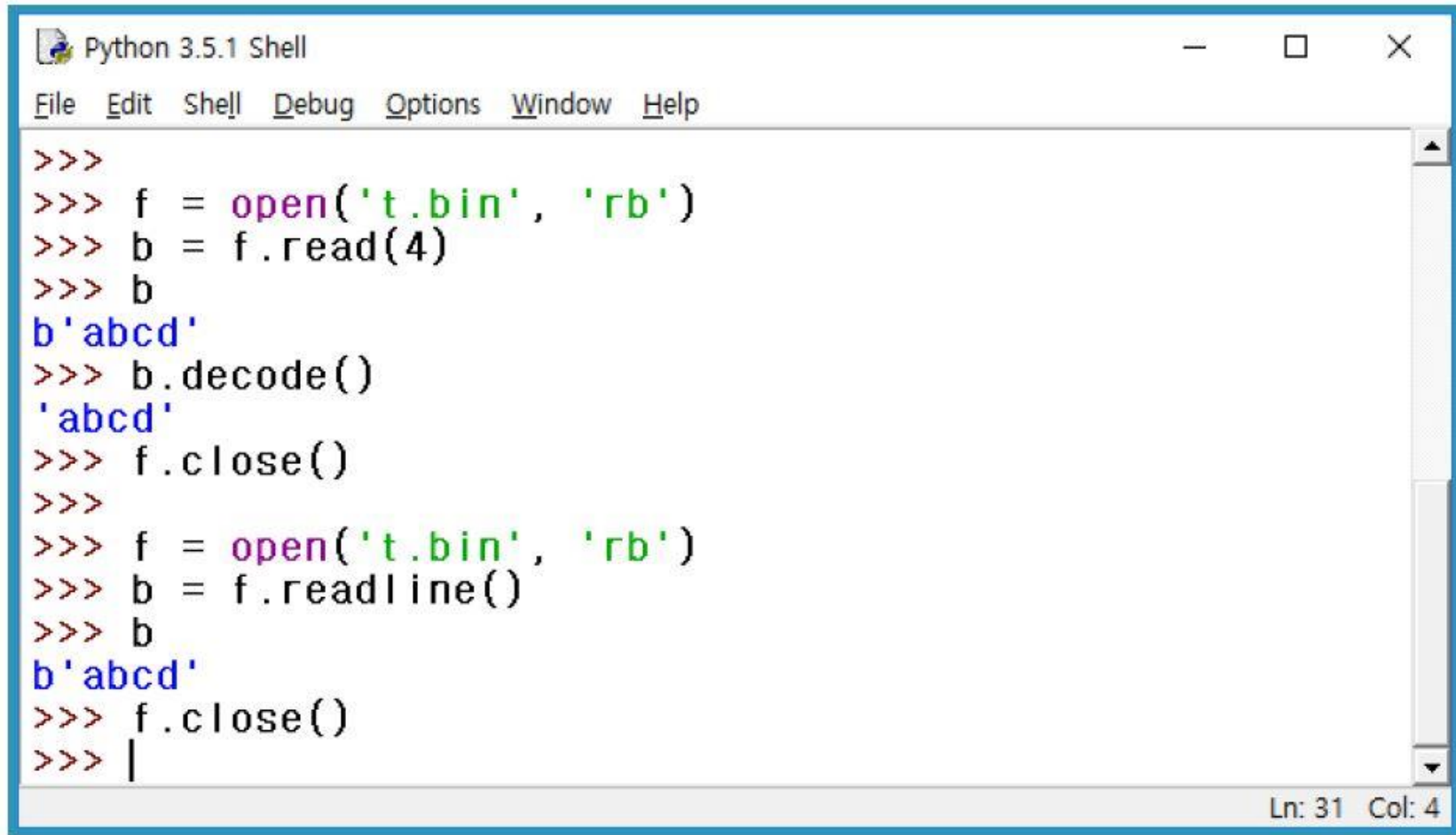
이진 파일 쓰기



```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
2016, 09:49:46) [MSC v.1900 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more inf
ormation.
>>> f = open('t.bin', 'wb')
>>> f.write('abcd')
Traceback (most recent call last):
  File "<pyshell#1>", line 1, in <module>
    f.write('abcd')
TypeError: a bytes-like object is required, not 'str'
>>> 'abcd'.encode()
b'abcd'
>>> f.write('abcd'.encode())
4
>>> f.close()
>>> |
```

Ln: 14 Col: 4

이진 파일 읽기

A screenshot of a Python 3.5.1 Shell window. The window has a title bar with the text 'Python 3.5.1 Shell' and standard window controls (minimize, maximize, close). Below the title bar is a menu bar with options: File, Edit, Shell, Debug, Options, Window, and Help. The main area of the window contains a series of Python commands and their outputs. The commands are: >>>, >>> f = open('t.bin', 'rb'), >>> b = f.read(4), >>> b, >>> b.decode(), >>> f.close(), >>>, >>> f = open('t.bin', 'rb'), >>> b = f.readline(), >>> b, >>> f.close(), and >>> |. The outputs are: b'abcd' (twice) and a vertical bar character. The status bar at the bottom right shows 'Ln: 31 Col: 4'.

```
Python 3.5.1 Shell
File Edit Shell Debug Options Window Help
>>>
>>> f = open('t.bin', 'rb')
>>> b = f.read(4)
>>> b
b'abcd'
>>> b.decode()
'abcd'
>>> f.close()
>>>
>>> f = open('t.bin', 'rb')
>>> b = f.readline()
>>> b
b'abcd'
>>> f.close()
>>> |
Ln: 31 Col: 4
```

실습 02

- 이미지 파일을 읽어서 다른 이미지 파일로 복사하는 프로그램을 작성 하시오.

원본 파일 이름을 입력하시오: *123.png*
복사 파일 이름을 입력하시오 : *456.png*
123.png를 456.png로 복사하였습니다.

파일 명 입력 받기

sourceName = input("원본 파일 이름을 입력하시오 : ")

destName = input('복사 파일 이름을 입력하시오 : ')

파일 열기

infile = open(sourceName, 'rb')

outfile = open(destName, 'wb')

입력 파일에서 1024 바이트씩 읽어서 출력 파일에 쓴다

buffer = infile.read(1024)

while buffer :

 outfile.write(buffer)

 buffer = infile.read(1024)

파일 닫기

infile.close()

outfile.close()

print(sourceName + '를 ' + destName + '로 복사하였습니다.')

파일 임의 접근(Random Access)

- 순차적 접근
 - 파일의 앞부터 순차적으로 읽고 쓰는 것
- 임의 접근
 - 파일의 임의의 위치에 있는 내용에 접근 하는 것
- 메소드
 - seek(n) : 파일의 n 번째 바이트로 이동
 - seek(n, os.SEEK_CUR) : 현재 위치에서 n 바이트 이동. 이진 파일에서만 동작
 - seek(n, os.SEEK_END) : 파일 끝에서 n 바이트 이동. n은 보통 음수. 이진파일에서만 동작
 - tell() : 현재의 파일 포인터 위치를 돌려줌

실습 03

- 실행하여 결과 값을 확인 보시오.

```
>>> f = open('t.txt', 'wb+')          # 읽고 쓰기. 기존 파일 삭제
>>> s = b'0123456789abcdef'          # 이진 파일이므로 바이트 자료형 사용
>>> f.write(s)
16
>>> f.tell()                          # 현재 위치 확인. 파일의 끝
16
>>> f.seek(5)                         # 시작부터 5번째 위치로 이동
5
>>> f.tell()                          # 위치 확인
5
>>> f.read(1)                         # 한 바이트 읽기
b'5'
>>> import os                         # os 모듈 사용
>>> f.seek(2, os.SEEK_CUR)            # 현재 위치에서 2바이트 이동
8
>>> f.seek(-3, os.SEEK_END)           # 파일의 끝에서 3바이트 앞으로 이동
13
>>> f.read(1)                         # 한 바이트 읽기
b'd'
```

객체 입출력 : pickle 모듈

- 객체를 파일로 저장할 때
 - import pickle
 - pickle.dump(출력할 객체, 파일 객체)
- 객체를 파일에서 읽을 때
 - pickle.load(파일 객체)

실습 04 : 객체 입출력 : pickle 사용 예

```
import pickle
f = open('test.pickle', 'wb')
phone = {'윤소정':'010-1234-5678', '김철수':'010-2345-6789',
        '이영희':'010-3456-7890'}

pickle.dump(phone, f) # 파일에 딕셔너리 저장
f.close()
del phone

f = open('test.pickle', 'rb')
phone = pickle.load(f) # 파일에 저장된 딕셔너리 읽기

print(phone)
```

실습 05

- 친구 관리 프로그램을 수정하여 연락처 목록이 프로그램 종료 후에도 계속 유지 되도록 하시오.

1. 연락처 목록 출력 : 친구 이름과 연락처 목록 출력
2. 연락처 추가 : 친구 이름과 연락처 추가
3. 연락처 삭제 : 이름을 입력하면 연락처까지 삭제
4. 연락처 변경 : 이름을 입력하면 수정할 연락처 입력 받음
5. 종료

* 6주차_튜플_딕셔너리 : 실습 02(친구 관리) 참고 및 수정

예외 처리(Exception Handling)

- 예외(Exception) ?
 - 실행 도중 발생하는 오류
- 예외 처리의 목적
 - 오류가 발생 했을 때 사용자에게 알려줌
 - 데이터의 손실이 없도록 처리
 - 중단 없이 계속 실행

예외 처리 방법

try : 예외가 발생할 수 있는 코드
except 오류 종류 : 예외 처리 코드

```
In [5]: while True :  
        try :  
            n = int(input('정수를 입력하세요 : '))  
            break  
        except ValueError :  
            print('정수가 아닙니다. 다시 입력하세요\n')  
  
        print('정수 입력이 성공하였습니다')
```

정수를 입력하세요 : 123
정수가 아닙니다. 다시 입력하세요

정수를 입력하세요 : asdf
정수가 아닙니다. 다시 입력하세요

정수를 입력하세요 : 5
정수 입력이 성공하였습니다

다중 예외 처리

- 예외 종류 별로 다른 처리가 필요할 때

```
try :  
    예외가 발생할 수 있는 코드  
except 오류 종류1 :  
    예외 처리 코드1  
except 오류 종류2 :  
    예외 처리 코드2  
except [ Exception as err ] :  
    예외 처리 코드3  
else :  
    예외가 없는 경우 실행 코드
```

다중 예외 처리

- 몇 가지 예외에 대해 동일한 처리를 할 때

```
try :  
    예외가 발생할 수 있는 코드  
except (오류1 , 오류2, 오류3) :  
    예외 처리 코드1  
except :  
    예외 처리 코드2
```


반드시 실행 : finally

try :
예외가 발생할 수 있는 코드
except 오류 as name :
예외 처리 코드
finally :
반드시 실행할 코드

```
In [2]: try :  
        f = open('test.txt', 'r')  
        f.write('데이터를 파일에 씁니다')  
        # 기타 필요한 파일 연산을 수행합니다  
except IOError as err :  
    print('파일 오류 : ', str(err))  
finally :  
    f.close()
```

파일 오류 : not writable

프로그래밍 과제 09

- 교안의 실습 01, 실습 02, 실습 03, 실습 04, 실습 05를 코딩하여 구글 클래스 룸에 올리기 바랍니다.
 - Lab01.py, Lab02.py, Lab03.py, Lab04.py, Lab05.py
- 누구나 쉽게 이해 할 수 있는 주석문을 잘 다는 것이 중요 합니다.
- 제출 기한 엄수 : 12월 15일(월) 까지