



# <sup>9</sup>~~10~~. 파일 입출력

대량 데이터 입출력 방법

## ◆ 왜 파일을 이용하는가

- 중요한 데이터는 영구적으로 남겨야 한다.
- python 프로그램이 종료되면 데이터가 소멸된다.
- 컴퓨터는 전원이 꺼지면 데이터가 소멸된다.
- 파일은 데이터를 영구적으로 저장할 수 있는 방법이다.
- 파일은 대량의 데이터를 이동할 때 쓸 수 있는 유용한 방법이다.
- 파일에 데이터를 저장하는 방법과 파일에서 데이터를 읽어오는 방법을 알아야 한다.

# 파일 읽기/쓰기 절차

파일에 데이터를 기록하려면	파일에 있는 데이터를 가져오려면
① 파일을 열기(생성) ② 데이터를 기록하기 ③ 파일 작업을 종료하기	① 파일을 열기(선택) ② 데이터를 읽기 ③ 파일 작업을 종료하기

## ◆ 파일 생성/선택하기

`f = open('파일이름.txt', 'w')`  
*(w: write의 의미)*  
*f: 파일을 가리키는 변수 → 변수로 남아있어야  
후속작업 가능!*

파일객체 = open(파일이름, 파일모드)

`f` → `파일이름.txt`

## ◆ 파일 종료하기

- 파일 읽기/쓰기 작업을 마치고 프로그램에서의 파일 사용을 마친다.
- ✓ 파일 종료하기를 해야 파일 작업이 마무리된다. (안 할 수도 있다)

`f.close()` → ③에 공동적으로 사용

# 파일 선택

## ◆ 파일 선택하기

파일객체 = open(파일이름, 파일모드)

- 읽을 데이터가 저장된 파일을 선택

f = open('data.txt', 'r') ← read의bi  
← 파일 모드

## ◆ 파일 모드

모드	설명
r	파일에서 데이터를 읽을 때
w	파일에 데이터를 기록할 때
a	파일의 끝에 추가된 내용을 기록할 때

append

파일에서 데이터 읽을 때에는

```
f = open('data.txt', 'r')
```

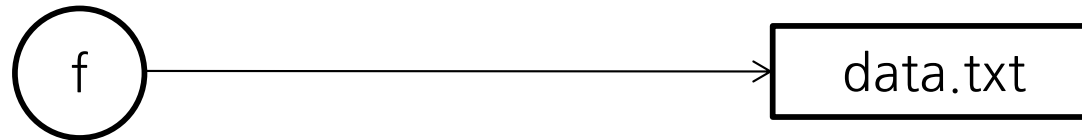
파일에 데이터를 기록할 때에는

```
f = open('data.txt', 'w')
```

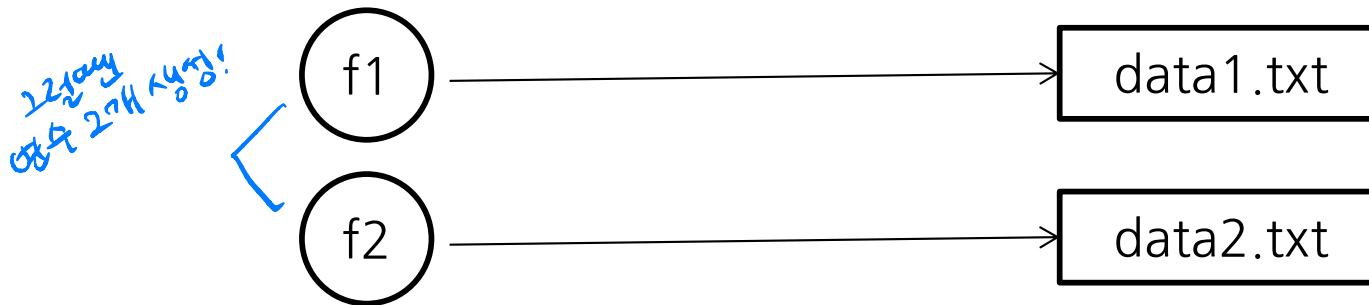
# 파일 선택

## ◆ `f = open('data.txt', 'r')` 의 의미

- `f`는 파일을 가리키는 식별자



- 프로그램에서 데이터를 읽고 쓸 때에는 변수 `f`를 이용한다.
- 한 번에 2개 이상의 파일을 읽고 쓸 수도 있다.



```
f1 = open('data1.txt', 'r')  
f2 = open('data2.txt', 'w')  
# 읽기 또는 쓰기 작업
```

data1.txt에서 읽어서  
data2.txt에 기록하기 위한 준비

# 파일에 데이터 기록하기

```
f = open('data.txt', 'w')
```

```
for i in range(1, 11):
```

```
    data = "%d번째 줄입니다.\n" % i
```

```
    f.write(data)
```

```
f.close()
```

결과를 확인해 보자. 파일은 어디에 있을까? 파일 내용은 어떻게 볼까?

내가 원래 파일 넣은  
곳에다가!

```
① for i in range(10):  
    f = open("data%d.txt" % i, "w")  
    f.write("memory")  
    f.close
```

→ 아예 파일 10개 생성!

파일객체.write(기록할 내용)

```
② f=open("2dan.txt", "w")  
for i in range(1, 10):  
    f.write("%d * %d = %d" % (2, i, 2*i))  
f.close()
```

(주) 기존 파일을 'w'로 열면 기존의 데이터가 없어진다.

→ 자칫하면 파일은 휴지통에도 갔음!  
완전히 없어지는 것!

# 파일에서 데이터 읽기

## ◆ readline( ) - 문자열 한 줄을 읽는다

```
f = open('data.txt', 'r')
```

```
line = f.readlines()
```

```
print(line)
```

```
f.close()
```

↳ 여러줄을 읽을때  
(이렇게 모든행)

```
ret = 파일객체.readline( )
```

## ◆ 모든 내용을 읽어 화면에 출력

```
f = open('data.txt', 'r')
```

```
while True:
```

```
line = f.readline()
```

```
if not line:break
```

```
print(line)
```

```
f.close()
```

↳ 한줄씩 읽어오면서  
다 출력하는 방법

즉, 이 방법은 화면에 파일의  
모든 내용을 표시!

↳ 한 줄씩 읽어와서 출력하려면  
readline쓰는게 better!

# readlines(), read()

◆ readlines() - 파일을 모두 읽어 리스트로 반환한다.

```
f = open('data.txt', 'r')
```

```
mlines = f.readlines()
```

```
[ for line in mlines :  
    print(line) ]
```

→ 한줄씩  
출력해서  
화면에 표시해라!

```
f.close()
```

◆ read() - 파일을 모두 읽어 문자열로 반환한다.

```
f = open('data.txt', 'r')
```

```
data = f.read()
```

```
print(data)
```

```
f.close()
```

숫자는 나중에 따로  
숫자로 변환해줘야지!

① readline() ② readlines() ③ read()

어느 것이 더 좋을까?



# 파일 끝에 추가하기

- ◆ 'w' 모드는 기존 데이터를 삭제하고 새 데이터를 기록한다.
- ◆ 'a' 모드는 기존 데이터의 끝에 새 데이터를 **추가**한다.

```
f = open('data.txt', 'a')
for i in range(11, 20):
    data = "%d번째 줄입니다.\n" % i
    f.write(data)
f.close()
```

↳ 참고만!  
이런식으로 잘 안됨

# 실습 (1)

## ◆ 사용자가 입력한 문장을 파일에 저장하는 프로그램을 작성하라.

- 사용자가 “end” 이라고 입력하면 입력이 종료되고 저장된다.
- 파일 이름은 input.txt 로 한다.
- 실행 예)
  - > Hello World
  - > nice to meet you
  - > good morning vietnam
  - > end

파일을 저장하였습니다.

## ◆ 이 파일을 읽어 화면에 표시하고 몇 행으로 구성되었는지 나타내는 프로그램을 작성하라.

# with의 사용

## ◆ 잠시 파일을 열어 읽기 처리를 한 후 파일을 바로 닫을 때

- 좀 더 간단하게 파일을 열고 닫을 수 없을까?

```
with open("data.txt", "w") as f :
```

```
    f.write("Life is short, you need python")
```

간단하게 1줄만 읽을때  
but 이렇게 쓸 필요 없음

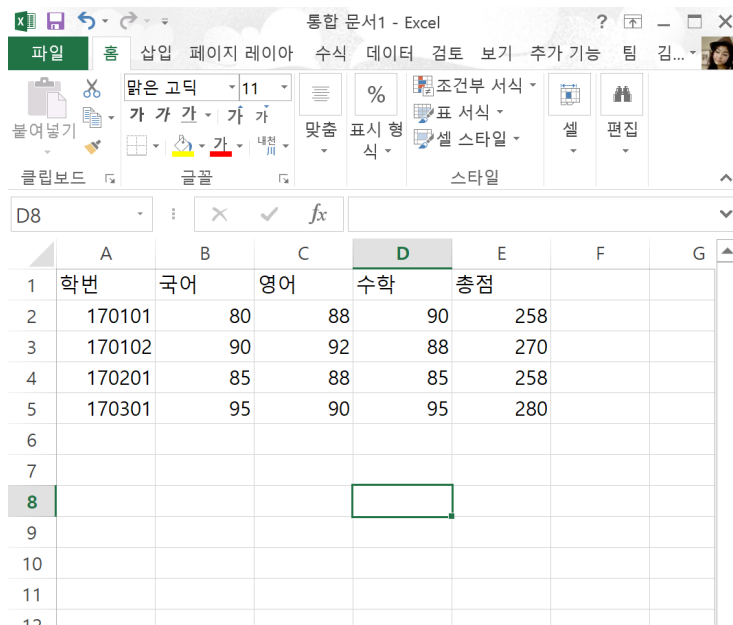
- 문장 수행과 함께 파일은 close 된다.(f.close가 필요 없다)

## ◆ 엑셀은 데이터를 다루는 매우 범용적인 도구이다.

- 엑셀에서 만들어지거나 엑셀에서 처리되는 데이터도 많다.

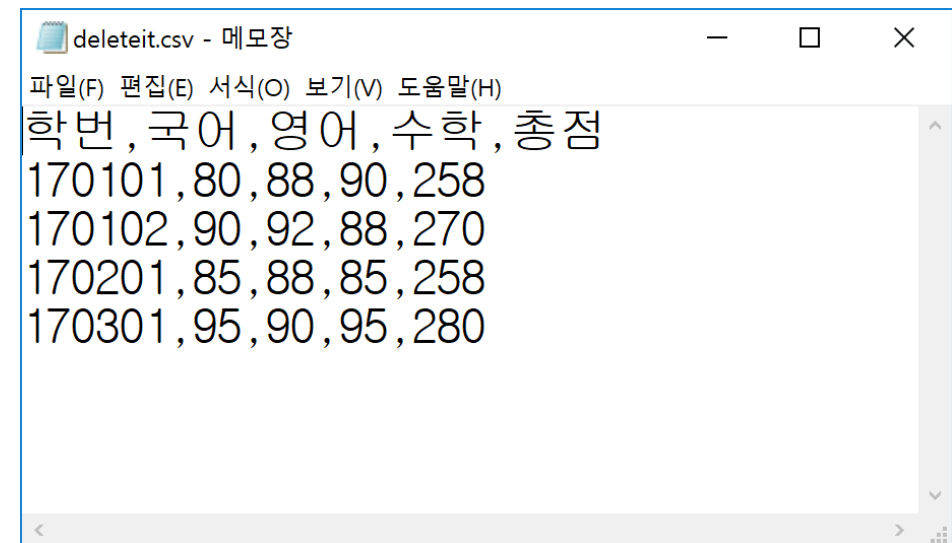
## ◆ 엑셀로 만들어진 데이터를 파이썬 프로그램에 가져오려면?

- CSV(comma separated value) 파일을 이용 → 저장시 파일형태 CSV로 저장!



통합 문서1 - Excel

	A	B	C	D	E	F	G
1	학번	국어	영어	수학	총점		
2	170101	80	88	90	258		
3	170102	90	92	88	270		
4	170201	85	88	85	258		
5	170301	95	90	95	280		
6							
7							
8							
9							
10							
11							
12							



deleteit.csv - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

학번,국어,영어,수학,총점  
170101,80,88,90,258  
170102,90,92,88,270  
170201,85,88,85,258  
170301,95,90,95,280

# csv 파일 처리

```
f = open('score.csv', 'r')
```

```
while True:
```

```
    v = f.readline()
```

더이상 읽을게  
없으면 while문 탈출!

```
    if v == '': break;
```

```
    v=v[:-1]
```

#왜 넣었을까?

```
    s = v.split(',')
```

제일 마지막 문자열을  
날려버린다는 의미!  
→ readline으로 읽으면  
제일 마지막에 줄바꿈 문자가 있을!  
→ 그래서 줄바꿈 없애!  $\backslash n$

```
    s.append(int(s[1])+int(s[2])+int(s[3]))
```

↳ 숫자로 바꿔주고,  
합계를 리스트에 추가!

```
    print(s)
```

```
f.close()
```

"20170101, 55, 44, 33"

["20170101", "55",  
"44", "33"]  
↳ 리스트로 변환

but 앞의 코드는 CSV로부터

불러온 파일이 적어도 첫번째 줄이

"학년, 국어, 영어, 수학, 총점"으로

시작하지 않았을 때 가능!

주의

파일의 맨 마지막에

① 끝바꿈 문자가 없다면?

② 공백이 들어가 있다면?

↳ 파일이 어떻게 닫혀있는지 보는 것!!

## 실습 (2) - CSV 파일 읽기

189쪽

◆ 앞 예제를 엑셀에서 데이터를 작성한 후 파이썬으로 가져와  
총점, 평균을 계산, 출력하는 프로그램을 작성하라.

- 만일 다음의 에러가 나온다면,
- UnicodeDecodeError: 'cp949' codec can't decode byte 0xed in position 40: illegal multibyte sequence
- open을 다음과 같이 수정하라.

```
f = open("data.txt" , 'r', encoding='utf-8')
```

↳ CSV파일이 utf-8 방식으로 encoding된 것!  
그래서 불러올 때 'utf-8'로 지정해주어야 함

- ◆ 컴퓨터는 숫자만 저장한다. 문자를 저장하려면 문자와 숫자를 대응시킨 후, 숫자를 저장해야 한다.
  - 이것이 ASCII이다. <sup>아스키</sup> 영문자, 숫자, 기호를 127자로 정의 (미국)
- ◆ 유럽의 일부 문자(ñ, ü)를 저장할 방법 추가
  - ISO-IEC 8859-1 (유럽)
- ◆ 한글, 일본어, 중국어는 저장할 수가 없다.
  - DBCS(2바이트, 16비트 문자) - EUC-KR, KSC5601, CP949 ↓ 영어만 가지고 표현 K
- ◆ 국가를 넘어서 사용할 표준이 필요하다. → 각나라의 표준을 전세계에서 쓰니까!
  - Unicode (2 바이트) 한글자당 2바이트!
- ◆ 최신 문자, 고대 문자 추가하니 2바이트로 부족
  - UTF 방식 (UTF-8을 가장 많이 사용)



# 파일 입출력 이외의 기능

◆ 파일 입출력은 운영체제 공통 기능이고, 그 외의 폴더(디렉토리) 관리 기능은 운영체제마다 다르다.

- 운영체제에 상관없이 동작하도록 만든 기능(모듈)을 쓰거나
- 운영체제마다 별도로 작성한 기능(모듈)을 써야 한다.
- <https://docs.python.org/3/library/filesys.html> → 참고

# 생각해야 할 점(1)

- ◆ 파일의 데이터가 예상처럼 있는 것은 아니다.
- ◆ 프로그램은 규칙대로 파일을 읽기 때문에 규칙과 다르게 파일의 데이터가 존재하면 정상적인 읽기에 실패한다.
- ◆ 따라서 파일에서 데이터를 읽을 때에는 해당 파일이 규칙에 맞는지 확인해야 한다.
  - 맞지 않으면 규칙에 맞도록 하든지
  - 규칙을 바꾸든지
  - 규칙에 맞지 않는 데이터를 삭제하든지

## 생각해야 할 점(2)

### ◆ 파일의 데이터를 어느 자료형에 담아야 할까?

- 사전은 단어와 뜻으로 이루어져 있다.
- 리스트? 딕셔너리?
- 정답) 프로그램의 요구사항에 맞는 자료형에 담아야 한다.

### ◆ 요구사항의 예(사전)

- 다음의 요구사항에 맞는 사전의 자료형을 생각해 보자.
  - 단어의 일부분만 넣어도 검색 가능한 사전
  - 뜻으로부터 단어를 찾는 사전
  - 품사가 중요한 사전