

Chap 6. 디렉토리와 파일 관리하기

1. OS 모듈을 통한 디렉토리 관리

이번에는 파이썬 모듈을 이용해서 운영체제의 기본적인 기능을 사용하는 방법을 배울 거예요. 운영체제의 기본적인 기능이란 건 대부분 우리가 운영체제를 사용하는 명령을 의미합니다~ 예를 들어 파일을 복사하거나 폴더를 새로 만들거나 파일을 지우는 것 같은 작업들이죠. 파이썬에서는 아주 막강한 OS 관련 라이브러리들을 제공합니다. 그리고 이 책에서는 윈도 OS를 기준으로 설명 드리겠습니다.

파이썬에서 OS 관련 모듈 이름은 os(소문자) 입니다 ^^
os모듈에도 많은 기능들을 해 주는 함수가 있으므로 어떤 함수가 있는지 한 개씩 살펴보면서 공부해보도록 하죠~
우선 os모듈을 import 해야겠죠?

(1) getcwd() 함수 / chdir() 함수

```
1  #getcwd( ) 와 chdir( )
2
3  import os
4
5  print('현재 디렉토리:' , os.getcwd() )
6  os.chdir("c:\\\\temp")
7  print('이동 후 디렉토리:' , os.getcwd() )
```

현재 디렉토리: c:\\py_temp
이동 후 디렉토리: c:\\temp

위 그림에서 보듯이 getcwd() 함수는 현재 디렉토리의 위치를 보여주고 chdir() 함수는 디렉토리를 이동시키는 함수입니다.

(3) listdir() 함수 - 입력한 경로의 파일과 폴더 목록을 리스트로 반환하는 함수

```
1  # listdir( ) 함수 - 지정된 폴더내의 파일과 폴더의 목록을 출력하는 함수
2
3  os.listdir("c:\\\\py_module")
```

['ex_2.py', 'ex_3.py', '__pycache__']

위와 같이 지정된 디렉토리에 있는 내용들이 많을 경우 한꺼번에 모두 출력이 되어 보기가 불편합니다.

그래서 아래와 같이 for 문을 사용해서 보기 좋게 변경할 수 있습니다.

[파이썬 능력자 너도 될 수 있어~! - 서진수 저 -]

```

1 # listdir( ) 함수 - 지정된 폴더내의 파일과 폴더의 목록을 출력하는 함수
2
3 for i in os.listdir("c:\\py_module") :
4     print(i)

```

ex_2.py
ex_3.py
__pycache__

(4) os.path.exists() 함수 - 특정 폴더가 있는지 확인하는 함수

```

1 #os.path.exists( ) 함수 - 특정 폴더의 존재 여부 확인
2
3 print( os.path.exists("c:\\py_temp") )
4
5 print( os.path.exists("c:\\temp100") )

```

True
False

위 그림과 같이 폴더가 존재하면 True 값을 반환하고 없을 경우 False 값을 반환합니다.

위 함수와 비슷한 함수로 os.path.isdir() 함수도 있는데 괄호안의 디렉토리가 있으면 True 값을 반환하고 없으면 False 값을 반환합니다.

(5) mkdir() 함수와 makedirs() 함수 - 폴더를 만들어 주는 함수

mkdir() 함수와 makedirs() 함수는 모두 폴더를 만드는 함수입니다. 두 함수의 차이점은 mkdir 함수는 경로의 제일 마지막에 적힌 하나의 폴더만 생성하지만 makedirs 함수는 시작부터 끝까지 모든 폴더를 만들어줍니다. 아래 그림을 보세요.

```

1 #mkdir( ) 함수와 makedirs( ) 함수
2
3 os.mkdir("c:\\temp1\\temp2")

```

```

-----
FileNotFoundError                                Traceback (most recent call last)
<ipython-input-3-d70ddc73a966> in <module>
      1 #mkdir( ) 함수와 makedirs( ) 함수
      2
----> 3 os.mkdir("c:\\temp1\\temp2")

FileNotFoundError: [WinError 3] 지정된 경로를 찾을 수 없습니다: 'c:\\temp1\\temp2'

```

```

1 os.makedirs("c:\\temp1\\temp2")

```

참고로 두 명령 모두 생성하려는 폴더가 이미 존재할 경우 아래와 같이 에러를 발생합니다

```
1 os.makedirs("c:\\temp1\\temp2")
```

```
FileExistsError                                Traceback (most recent call last)
<ipython-input-5-432d97031410> in <module>
----> 1 os.makedirs("c:\\temp1\\temp2")

c:\\python 3.5\\lib\\os.py in makedirs(name, mode, exist_ok)
    239         return
    240     try:
--> 241         mkdir(name, mode)
    242     except OSError:
    243         # Cannot rely on checking for EEXIST, since the operating system
FileExistsError: [WinError 183] 파일이 이미 있으므로 만들 수 없습니다: 'c:\\temp1\\temp2'
```

(6) rmdir() 와 removedirs() 함수 - 폴더(디렉토리)를 삭제하는 함수

위 함수들은 모두 폴더를 삭제하는 명령입니다. 두 명령어 모두 삭제할 폴더는 반드시 비어 있어야 삭제가 됩니다. 아래를 보세요~

```
1 # rmdir( ) 함수와 removedirs( ) 함수
2
3 os.rmdir("c:\\temp1\\temp2\\")
4 os.rmdir("c:\\temp1\\")
5
6 os.makedirs("c:\\temp1\\temp2\\")
7 os.removedirs("c:\\temp1\\temp2\\")
```

위 그림에서 c:\\temp1\\temp2 폴더를 삭제하고 싶을 경우를 가정해서 설명하겠습니다.

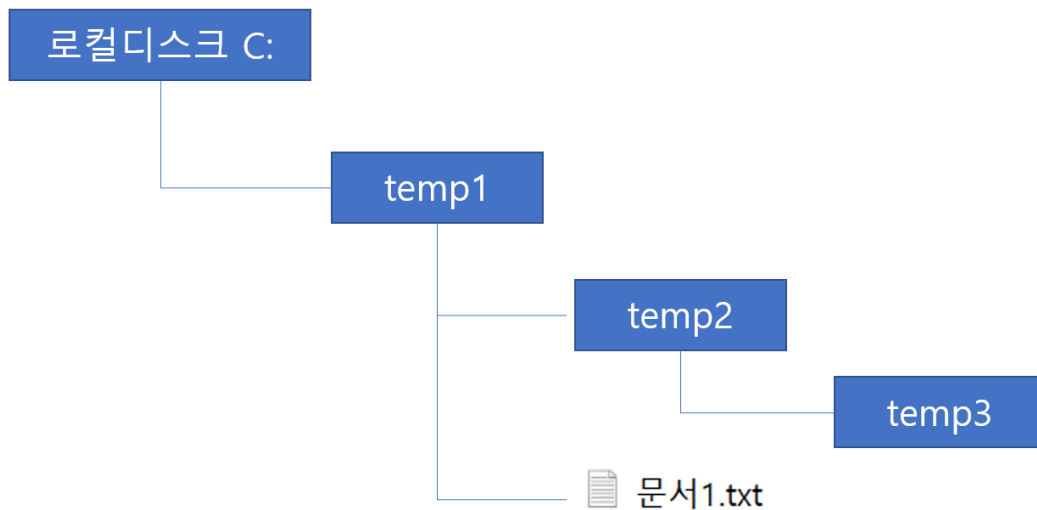
3번행은 rmdir() 함수를 사용해서 폴더를 지우는데 c:\\temp1\\temp2 폴더 아래에 있는 temp2 폴더를 삭제합니다. 이때 temp2 폴더의 상위 폴더인 temp1 폴더는 삭제가 안됩니다. 그래서 4번 행에서 c:\\temp1 폴더를 한번 더 삭제를 해 주어야 합니다.

그러나 7번행의 removedirs() 함수를 이용할 경우 모든 경로의 폴더를 한번에 전부 삭제를 할 수 있어서 훨씬 더 빠르고 편리합니다.

다만 삭제하고 싶은 폴더에 다른 파일이 있을 경우는 삭제할 수 없습니다.

만약 아래 그림과 같은 구조로 되어 있는 상황에서 removedirs() 명령으로 temp2 폴더를 삭제할 경우 temp3 폴더까지 문제없이 모두 삭제를 할 수 있습니다.

그런데 만약 temp1 폴더를 삭제할 경우 문서1.txt 파일 때문에 디렉터리가 비어 있지 않습니다 라는 에러 메시지와 함께 폴더를 삭제할 수 없습니다.



실습을 통해서 살펴볼까요?

```

1 import os
2 os.removedirs("c:\temp1")
  
```

```

-----
OSError                                Traceback (most recent call last)
~\AppData\Local\Temp\ipykernel_17196\4082110081.py in <module>
      1 import os
----> 2 os.removedirs("c:\temp1")

c:\program files\python37\lib\os.py in removedirs(name)
    237
    238     """
--> 239     rmdir(name)
    240     head, tail = path.split(name)
    241     if not tail:
  
```

```
OSError: [WinError 145] 디렉터리가 비어 있지 않습니다: 'c:\temp1'
```

만약 위와 같은 구조에서 temp1 폴더를 삭제하고 싶을 경우 rm() 명령으로 파일을 먼저 삭제한 후 폴더를 삭제해야 합니다.

그런데 권장 사항은 폴더나 파일을 삭제하는 경우는 눈으로 직접 확인한 os 명령어로 삭제하는 것입니다. 이렇게 권장하는 이유는 혹시라도 파이썬 명령어로 잘 못된 삭제가 일어나는 것을 막기 위해서 입니다.

연습문제 1.

사용자에게 폴더명을 하나 입력 받아서 해당 폴더가 존재하지 않을 경우에는 해당 폴더를 만들고 해당 폴더가 존재할 경우 "폴더명_2" 의 이름으로 폴더를 만들도록 코드를 작성하세요

2. 다양한 형식의 파일 생성 및 수정하기

이번 시간에 배울 내용은 파이썬으로 여러가지 형식의 파일을 생성하고 수정하는 방법에 대해서 살펴보겠습니다. 나중에 크롤러 만들 때 텍스트나 이미지 등을 모아서 파일로 저장해야겠죠?

1) 텍스트 형식(txt) 파일 관리하기

파일을 생성하거나 열 때 모드라는 것을 정해줄 수 있는데, 예를 들어 파일에 들어갈 데이터들이 텍스트라면 텍스트모드, 그림(바이너리)이라면 바이너리모드를 설정할 수 있어요~

아래 표를 보면서 모드들을 정리해 보세요~

기호	모드
t	텍스트(기본)
b	바이너리
r	읽기(기본)
w	쓰기
a	이어쓰기
+	읽기, 쓰기

위 표에 모드에 (기본)이라고 적혀 있는 것은 아무것도 설정하지 않았을 때 기본 모드입니다.

(1) txt 파일에 내용 쓰기 - 방법 1

```

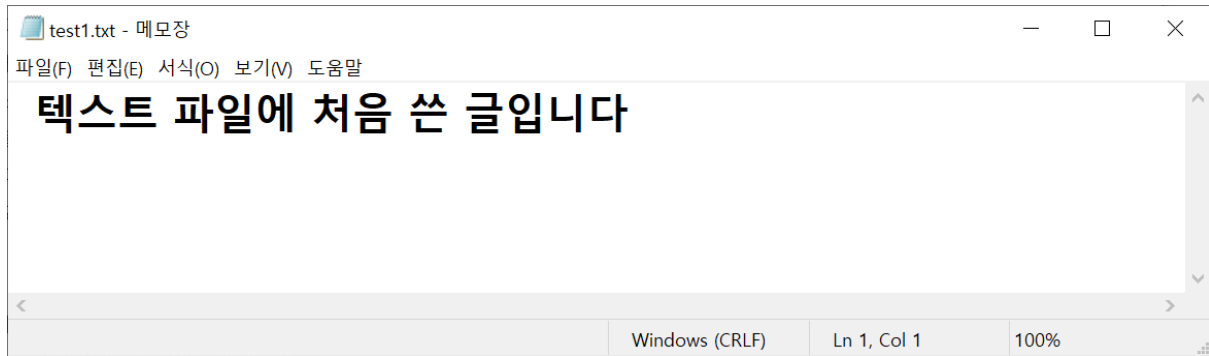
1 os.makedirs("c:\\py_temp2")
2 os.chdir("c:\\py_temp2")
3 print( os.getcwd( ) )
4
5 file = open("test1.txt", "w")
6 file.write(" 텍스트 파일에 처음 쓴 글입니다")
7 file.close( )

```

c:\\py_temp2

파일에 내용을 저장하는 순서는 open -> write -> close 의 순서라는 것을 기억하세요~

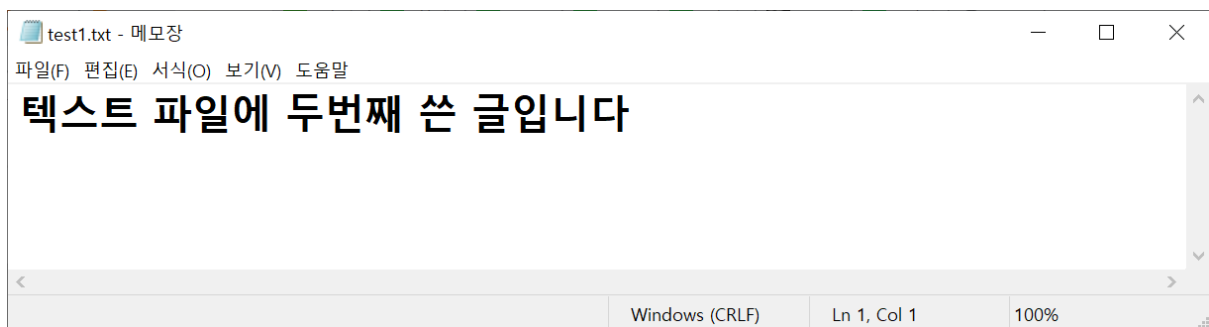
위에서 c:\\py_temp2\\ 폴더에 저장한 test1.txt 파일의 내용을 메모장으로 열어서 확인해 볼까요?



위 파일을 보면 실습에서 지정한 내용이 저장되어 있습니다.

그런데 w 모드는 무조건 덮어쓰는 모드라서 여러 건의 내용을 저장해야 할 경우에는 아주 위험합니다. 아래의 예를 보세요.

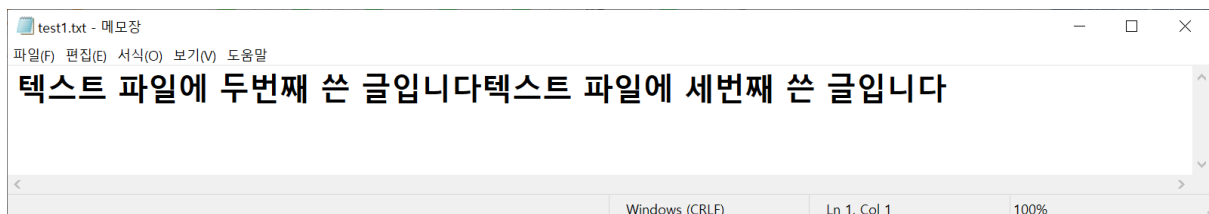
```
1 file2 = open("test1.txt", "w")
2 file2.write("텍스트 파일에 두번째 쓴 글입니다")
3 file2.close()
```



위 그림을 보면 두 번째 명령으로 저장한 글이 기존의 첫 번째 내용을 덮어썼다는 것을 알 수 있습니다.

웹 크롤러를 만들 때는 내용을 덮어쓰지 않고 먼저 크롤링한 내용 아래에 새로운 내용을 추가해야 하는데 이럴 때는 아래처럼 "a" (이어쓰기)모드를 사용하면 됩니다.

```
1 file3 = open("test1.txt", "a")
2 file3.write("텍스트 파일에 세번째 쓴 글입니다")
3 file3.close()
```



그런데 위의 그림을 보니까 줄 바꿈이 안되고 계속 이어서 추가가 되죠?

이럴 때는 줄바꿈 되는 엔터키를 함께 쓰도록 지정하면 됩니다.

아래 그림을 보세요.

```
1 file4 = open("test1.txt", "a")
2 file4.write("\n" + "텍스트 파일에 네번째 쓴 글입니다")
3 file4.close()
```

위 그림에서 2번째에 "\n" 기호가 엔터키를 의미하는 데 + 기호를 사용하여 텍스트 내용과 함께 사용을 했습니다. 저장된 결과를 볼까요?



이제 txt 형식의 파일에 저장하는 방법을 잘 아시겠죠?

(2) txt 파일에 내용 쓰기 - 방법 2

이번에는 다소 특이하지만 아주 요긴하게 사용되는 방법을 소개해 드립니다.

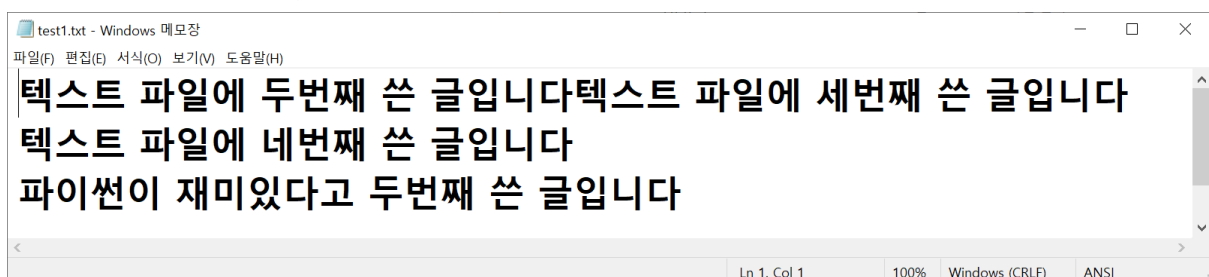
앞에서 배웠던 print()라는 명령어 기억하죠?

화면에 어떤 내용을 출력하는 함수인데 화면에 출력될 내용을 화면에 출력하지 말고 파일에 출력하도록 설정을 바꿀 수 있습니다.

아래 예를 보세요.

```
1 import sys
2
3 print('파이썬이 재미있다고 첫번째 쓴 글입니다')
4 orig_stdout = sys.stdout
5 file5 = open("test1.txt", "a")
6 sys.stdout = file5 #모니터에 출력하지 말고 file에 출력해라
7
8 print()
9 print('파이썬이 재미있다고 두번째 쓴 글입니다')
10
11 file5.close()
12 sys.stdout = orig_stdout #원래대로 변경 - 다시 화면에 출력시켜라
13 print('저장이 완료되었습니다')
```

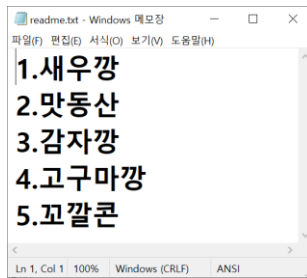
파이썬이 재미있다고 첫번째 쓴 글입니다
저장이 완료되었습니다



[파이썬 능력자 너도 될 수 있어~! - 서진수 저 -]

(3) txt 파일의 내용 읽기

테스트를 위해 아래 그림과 같은 파일을 c:\py_temp2\readme.txt 이름으로 만들어 주세요.



아래 그림처럼 여러 함수를 사용하여 읽기모드로 파일을 읽은 후 출력하겠습니다.

```
1  # txt 형식 파일 내용 읽기
2
3  f = open("c:\\py_temp2\\readme.txt", "r")
4  f.readlines()
```

['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']

또는 아래 그림처럼 인덱싱 기법으로 특정 행의 데이터만 조회할 수 있고 for 반복문을 이용해서 여러 건을 한꺼번에 출력할 수도 있습니다.

```
1  f = open("c:\\py_temp2\\readme.txt", "r")
2  snack = f.readlines()
3  print(snack)
4  print("\n")
5
6  #index 방법으로 특정 행을 조회하기
7  print(snack[0])
8  print(snack[1])
9  print("\n")
10
11 #for 반복문으로 한꺼번에 출력하기
12 for i in snack :
13     print(i)
14
15
```

['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']

1.새우깡

2.맛동산

1.새우깡

2.맛동산

3.감자깡

4.고구마깡

5.꼬깔콘

[파이썬 능력자 너도 될 수 있어~! - 서진수 저 -]

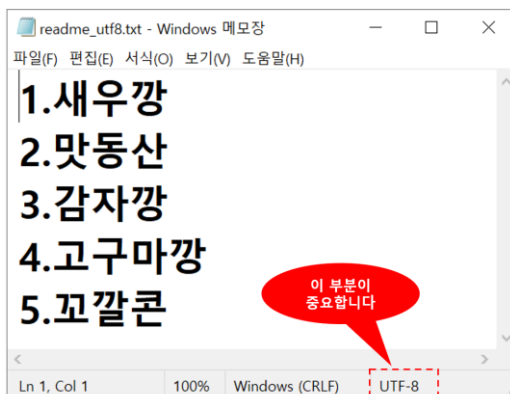
그런데 메모장에 저장된 파일을 불러올 때 아래와 같이 인코딩 관련 에러가 발생하는 경우도 있습니다.

```
1 f2 = open("c:\Wwpy_temp2\Wwreadme_utf8.txt", "r")
2 f2.readlines()

UnicodeDecodeError                                Traceback (most recent call last)
Input In [28], in <cell line: 2>()
      1 f2 = open("c:\Wwpy_temp2\Wwreadme_utf8.txt", "r")
----> 2 f2.readlines()

UnicodeDecodeError: 'cp949' codec can't decode byte 0xec in position 2: illegal multibyte sequence
```

위와 같은 에러의 원인은 파일을 저장할 때 사용한 인코딩 방식 때문에 발생한 문제입니다. 원본 파일을 다시 확인해볼까요?



위 그림에서 아래 부분에 UTF-8 부분이 저장할 때 사용한 인코딩 방식인데 이처럼 UTF-8로 되어 있을 경우 파일을 불러올 때도 반드시 `encoding="UTF-8"` 이나 `encoding="utf8"` 을 함께 적어야 합니다. 아래 예시를 보세요.

```
1 f2 = open("c:\Wwpy_temp2\Wwreadme_utf8.txt", "r" , encoding="UTF-8")
2 f2.readlines()

['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']

1 f3 = open("c:\Wwpy_temp2\Wwreadme_utf8.txt", "r" , encoding="utf-8")
2 f3.readlines()

['1.새우깡\n', '2.맛동산\n', '3.감자깡\n', '4.고구마깡\n', '5.꼬깔콘']
```

지금 배운 방법은 웹 크롤러를 만들 때 검색어 키워드나 검색 주소를 파일에 저장한 후 불러와서 자동으로 실행할 때 아주 많이 사용되는 방법이므로 꼭 이해하고 기억해 주세요~

2) xls 형식과 csv 형식의 파일 관리하기

(1) xls 형식과 csv 형식으로 저장하기

엑셀 형식과 csv 형식으로 저장하기 위해 표 형태로 데이터를 만들어야 합니다.

먼저 윈도우에 있는 cmd 창에서 아래와 같이 pandas 를 설치해 주세요.

```
C:\Windows\system32>pip install pandas
Collecting pandas
  Downloading https://files.pythonhosted.org/packages/b3/59/38c88e1b26779b287a82c3d7601ec42c15e4acef09196e870c4fe9b77bd4/pandas-0.24.2-cp35-cp35m-win_amd64.whl (8.5MB)
    100% |#####| 8.5MB 2.8MB/s
Requirement already satisfied: pytz>=2011k in c:\Wpython 3.5\lib\site-packages (from pandas) (2018.7)
Requirement already satisfied: numpy>=1.12.0 in c:\Wpython 3.5\lib\site-packages (from pandas) (1.15.1)
Requirement already satisfied: python-dateutil>=2.5.0 in c:\Wpython 3.5\lib\site-packages (from pandas) (2.7.5)
Requirement already satisfied: six>=1.5 in c:\Wpython 3.5\lib\site-packages (from python-dateutil>=2.5.0->pandas) (1.12.0)
Installing collected packages: pandas
Successfully installed pandas-0.24.2
You are using pip version 18.1, however version 19.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.

C:\Windows\system32>_
```

위 그림과 같이 설치가 완료되었다면 아래의 작업을 진행하면 됩니다.

```
1  import pandas as pd
2
3  # 표 ( 데이터 프레임 ) 만들기
4  no = [ ]
5  subject_name = [ ]
6
7  no.append(1)
8  no.append(2)
9  no.append(3)
10
11 subject_name.append('수학')
12 subject_name.append('과학')
13 subject_name.append('빅데이터')
14
15 subject = pd.DataFrame( )
16 subject['과목번호'] = no
17 subject['과목명'] = subject_name
18 print(subject)
19
```

	과목번호	과목명
0	1	수학
1	2	과학
2	3	빅데이터

위 표의 내용을 아래와 같이 csv , xls 형식으로 저장하면 됩니다.

그리고 아래 5번행의 to_excel() 함수를 사용하기 전에 먼저 xlwt 모듈이 설치되어 있어야 합니다.
pip install xlwt 명령으로 설치해 주세요

```
1 # csv 형식으로 저장하기
2 subject.to_csv("c:\\py_temp2\\subject.csv", encoding="utf-8-sig", index=False)
3
4 # xls 형식으로 저장하기
5 subject.to_excel("c:\\py_temp2\\subject.xls", index=False)
6
```

그런데 최근에 xls 형식으로 저장할 때 기존에 사용하던 xlwt 모듈을 더 이상 지원하지 않고 앞으로는 openpyxl 패키지를 지원한다는 안내 멘트가 아래 그림과 같이 출력되는 경우도 있습니다.

```
C:\Users\32160131\AppData\Local\Temp\ipykernel_22204\2701827497.py:3: FutureWarning: As the xlwt package is no longer maintained, the xlwt engine will be removed in a future version of pandas. This is the only engine in pandas that supports writing in the xls format. In stall openpyxl and write to an xlsx file instead. You can set the option io.excel.xls.writer to 'xlwt' to silence this warning. While this option is deprecated and will also raise a warning, it can be globally set and the warning suppressed.
  subject.to_excel("c:\\py_temp2\\subject.xls", index=False)
```

위와 같은 메시지가 나와도 xls 형식으로 저장하는 것은 문제가 없지만 위 메시지에서 권장하는 방법으로 저장하려면 cmd 창에서 pip install openpyxl 명령으로 openpyxl 패키지를 설치한 후 위의 5번 줄을 아래와 같이 변경해서 사용하면 됩니다.

```
관리자: 명령 프롬프트
C:\Windows\system32>
C:\Windows\system32>pip install openpyxl
Collecting openpyxl
  Downloading https://files.pythonhosted.org/packages/ba/06/b899c8867518df19e242d8cbc82d4ba210f5ffbeebb7704c695e687ab59c/openpyxl-2.6.2.tar.gz (173kB)
    100% |#####| 174kB 1.6MB/s
Requirement already satisfied: jdcal in c:\python 3.5\lib\site-packages (from openpyxl) (1.4)
Requirement already satisfied: et_xmlfile in c:\python 3.5\lib\site-packages (from openpyxl) (1.0.1)
Installing collected packages: openpyxl
  Running setup.py install for openpyxl ... done
Successfully installed openpyxl-2.6.2
You are using pip version 18.1, however version 19.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\Windows\system32>
```

subject.to_excel("c:\\py_temp2\\subject.xls", engine='openpyxl')

위와 같이 저장하면 폴더에 아래 그림과 같이 파일이 생성됩니다.

이름



subject.csv



subject.xls

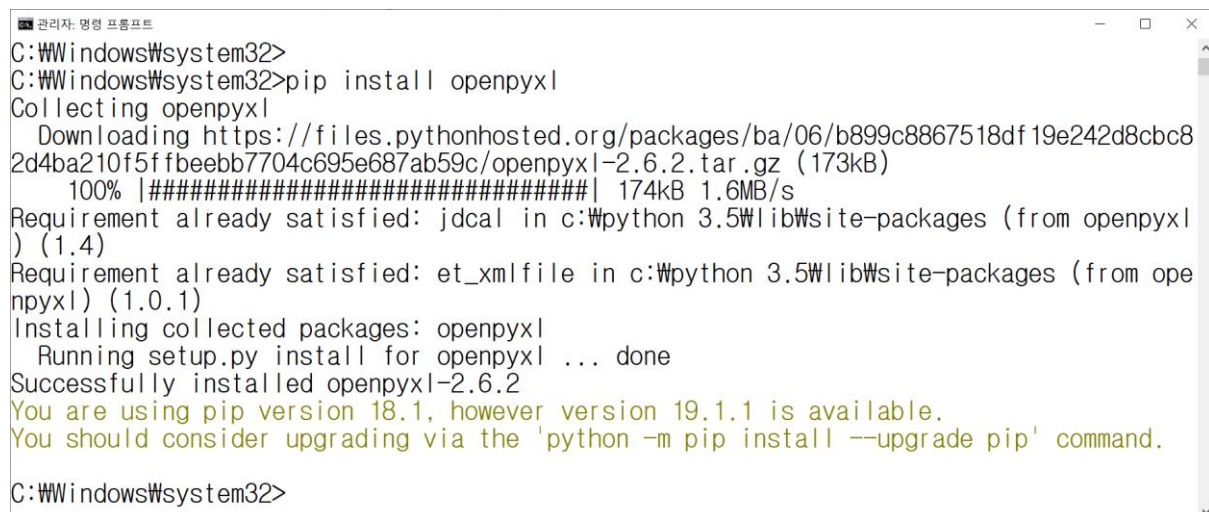
이 파일을 열어보면 앞에서 만든 데이터가 저장되어 있는 것을 확인할 수 있습니다.

(2) xlsx 형식과 xls 형식과 csv 형식의 파일 내용 읽어 오기

(a) xlsx 형식의 파일 내용 불러오기

파이썬에서 xlsx 파일의 내용을 불러오는 다양한 방법이 있는데 가장 널리 사용되는 패키지가 openpyxl 패키지입니다.

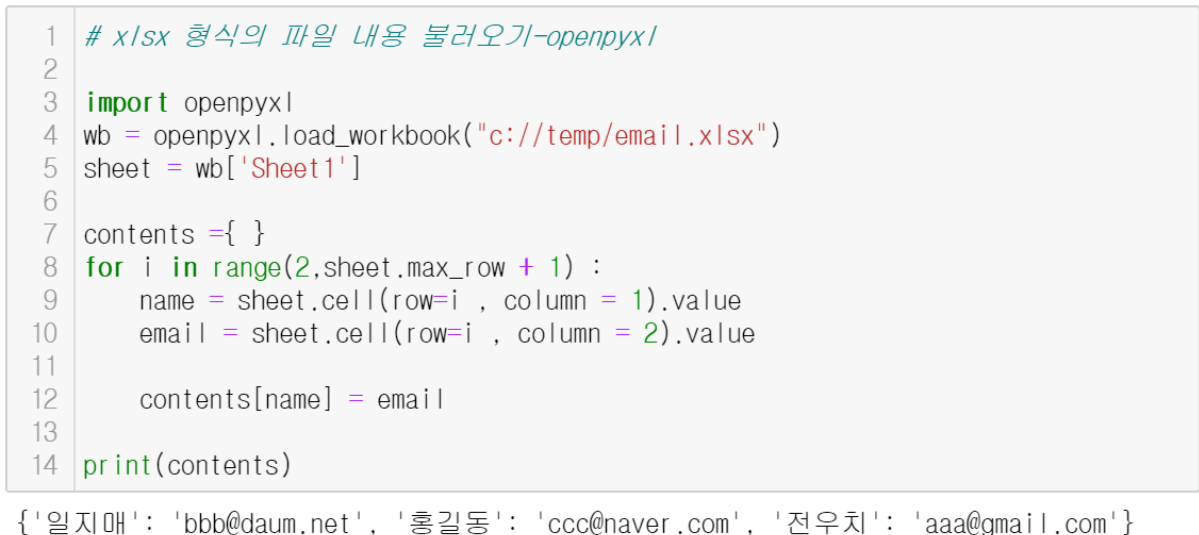
이 패키지는 기본으로 설치되어 있지 않기 때문에 아래와 같이 사용자가 추가로 설치한 후 import 해야 사용할 수 있습니다.



```

관리자: 명령 프롬프트
C:\Windows\system32>
C:\Windows\system32>pip install openpyxl
Collecting openpyxl
  Downloading https://files.pythonhosted.org/packages/ba/06/b899c8867518df19e242d8cbc82d4ba210f5ffb7704c695e687ab59c/openpyxl-2.6.2.tar.gz (173kB)
    100% |#####| 174kB 1.6MB/s
Requirement already satisfied: jdcal in c:\python 3.5\lib\site-packages (from openpyxl) (1.4)
Requirement already satisfied: et_xmlfile in c:\python 3.5\lib\site-packages (from openpyxl) (1.0.1)
Installing collected packages: openpyxl
  Running setup.py install for openpyxl ... done
Successfully installed openpyxl-2.6.2
You are using pip version 18.1, however version 19.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\Windows\system32>
  
```

위와 같이 openpyxl 패키지를 설치한 후 아래와 같이 엑셀 파일의 내용을 불러오면 됩니다



```

1  # xlsx 형식의 파일 내용 불러오기-openpyxl
2
3  import openpyxl
4  wb = openpyxl.load_workbook("c://temp/email.xlsx")
5  sheet = wb['Sheet1']
6
7  contents = { }
8  for i in range(2, sheet.max_row + 1) :
9      name = sheet.cell(row=i, column = 1).value
10     email = sheet.cell(row=i, column = 2).value
11
12     contents[name] = email
13
14  print(contents)
  
```

{ '일지매': 'bbb@daum.net', '홍길동': 'ccc@naver.com', '전우치': 'aaa@gmail.com' }

위 방법 말고 pandas 의 read_excel () 함수를 이용하여 xlsx 형식의 파일 내용을 불러올 수도 있습니다 . 아래 그림을 보세요.

```

1 # xlsx 형식의 파일 내용 불러오기-pandas
2 import pandas as pd
3 data = pd.read_excel("c:\\temp\\email.xls",
4                     sheet_name = 'Sheet1')
5 data

```

	name	email
0	전우치	aaa@gmail.com
1	일지매	bbb@daum.net
2	홍길동	ccc@naver.com

그리고 참고로 만약 xlsx 파일이나 xls 파일을 불러올 때 인코딩 에러가 발생할 경우 옵션으로 encoding='utf-8' 을 사용하면 가볍게 해결됩니다.

(b) csv 형식 파일 불러오기

파이썬에서는 csv 모듈이 제공이 되기 때문에 import csv 해서 바로 csv 파일의 내용을 불러올 수 있습니다.

```

1 # csv 형식의 파일 내용 불러오기
2
3 import csv
4
5 f = open('c:\\temp\\email.csv')
6 f_csv = csv.reader(f)
7 for i in f_csv :
8     print(i)

```

```

['name', 'email']
['전우치', 'aaa@gmail.com']
['일지매', 'bbb@daum.net']
['홍길동', 'ccc@naver.com']

```

그런데 만약 csv 파일의 인코딩 형식이 utf-8 일 경우 아래와 같이 추가 옵션을 사용해야 합니다.

```

1  # utf-8로 인코딩된 csv 형식의 파일 내용 불러오기
2
3  import csv
4
5  f = open('c:###temp###email_utf8.csv')
6  f_csv = csv.reader(f)
7  for i in f_csv :
8      print(i)

```

UnicodeDecodeError Traceback (most recent call last)

```

<ipython-input-26-4ac032c1d970> in <module>
      5 f = open('c:###temp###email_utf8.csv')
      6 f_csv = csv.reader(f)
----> 7 for i in f_csv :
      8     print(i)

```

UnicodeDecodeError: 'cp949' codec can't decode byte 0xec in position 15: illegal multibyte sequence

```

1  f2 = open('c:###temp###email_utf8.csv ', encoding="utf-8")
2  f2_csv = csv.reader(f2)
3  for i in f2_csv :
4      print(i)

```

```

['#ufeffname', 'email']
['전우치', 'aaa@gmail.com']
['일지매', 'bbb@daum.net']
['홍길동', 'ccc@naver.com']

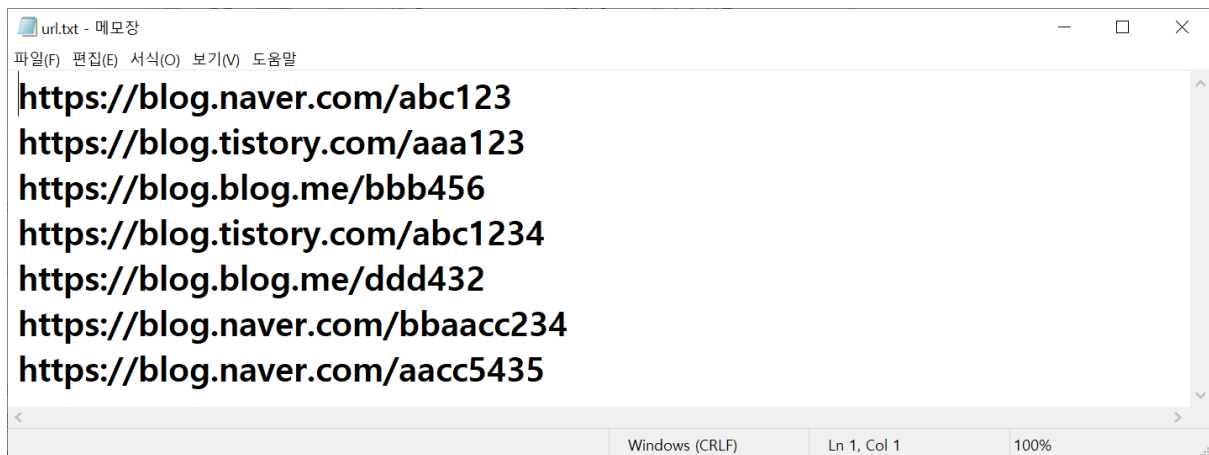
```

이 방법 외에도 pandas 모듈 안에 csv 파일을 불러오는 기능도 많이 활용되는데 이 부분은 이 책의 pandas 모듈 설명 부분을 참고하세요

연습문제 2.

주어진 "url.txt" 파일을 불러와서 url 주소가 *.naver.com 일 경우를 모두 모아서 "naver.com 주소는 몇 개입니다" 를 출력하고 url 주소가 *.blog.me 일 경우를 모두 모아서 "blog.me 주소는 몇 개입니다" 를 출력하고 url 주소가 *.tistory.com 일 경우를 모두 모아서 "tistory.com 주소는 몇 개입니다" 를 출력하도록 코드를 작성하세요.

[url.txt 파일 내용]



```
url.txt - 메모장
파일(F) 편집(E) 서식(O) 보기(V) 도움말
https://blog.naver.com/abc123
https://blog.tistory.com/aaa123
https://blog.blog.me/bbb456
https://blog.tistory.com/abc1234
https://blog.blog.me/ddd432
https://blog.naver.com/bbaacc234
https://blog.naver.com/aacc5435
Windows (CRLF) Ln 1, Col 1 100%
```

[출력 결과 내용]

```
blog.naver.com 의 갯수는 모두 3 개입니다
blog.tistory.com 의 갯수는 모두 2 개입니다
blog.blog.me 의 갯수는 모두 2 개입니다
```

연습문제 3.

아래 그림의 내용과 같이 주어진 '댓글내용.xlsx' 파일을 불러와서 댓글 내용만 아래의 예시화면처럼 출력하세요

[원본 xlsx 파일 내용]

	A	B	C
1	번호	댓글내용	
2		1 첫번째 댓글입니다	
3		2 두번째 댓글입니다	
4		3 세번째 댓글입니다	
5		4 네번째 댓글입니다	
6		5 다섯번째 댓글입니다	

[출력 결과 예시화면]

첫번째 댓글입니다
 두번째 댓글입니다
 세번째 댓글입니다
 네번째 댓글입니다
 다섯번째 댓글입니다