분석용 데이터 획득 전략

데이터의 저장 및 변환 기법

학습 목표

학습 목표

- 다양한 형식으로 데이터를 저장하는 원리를 익힐 수 있다.
- 자동 네이밍 기법을 통해 다양한 형식으로 데이터를 저장할 수 있다.

학습 내용

- 다양한 형식으로 데이터를 저장하는 원리
- 자동 네이밍 기법을 통한 데이터 저장하기

txt 형식으로 저장하기

1) txt 형식으로 저장 방식 1

1단계	파일준비 - open('파일명', 모드, encoding='utf-8') 모드 : w - 덮어쓰기 | a - 추가하기
2단계	내용쓰기 - write('파일에 쓸 내용')
3단계	저장하기 - close()

중요

불러올 파일이 없을 경우, 새 파일을 만들어 열어줌

utf-8

대부분의 환경에서 문자열 처리의 표준으로 사용되고 있는 가변 길이 문자 인코딩 방식

close()

메모리의 내용을 파일로 저장하기 위해 메모리와 디스크의 파일 간 연결을 종료하여 파일을 닫아주는 명령어

중요

open, write, close 3단계를 통해 데이터를 txt 형식으로 저장

txt 형식으로 저장하기

2) txt 형식으로 저장 방식 2

- 표준 출력 방향 바꾸기

1단계	표준 출력 방향 바꾸기 - sys.stdout 지정
2단계	화면에 출력할 내용
3단계	표준 출력 장치 원래대로 지정하기

리다이렉션

표준 입출력을 임의로 다시 지정하여 원하는 파일을 저장 또는 불러들이도록 하는 것

txt 형식으로 저장하기

2) csv, xlsx 형식으로 저장하기

(1) csv, xlsx 형식으로 저장 방식

- Data Frame 생성 후 저장하기

1단계	list나 dictionary 사용하여 컬럼 생성 후 데이터 입력
2단계	데이터 프레임으로 변환
3단계	변환된 데이터프레임을 csv, xlsx 형식으로 저장하기

- 디렉토리명과 파일명 자동 생성하기

중복되지 않도록 디렉토리명, 파일명 지정

결과를 저장할 디렉토리명과 파일명에 현재 날짜와 시간 사용

time 모듈의 localtime() 함수 사용

txt 형식으로 저장하기 - 실습

- 방법1: open, write, close 3단계

In [3]:
```
file = open("c:\\py_temp\\test1.txt","w")
file.write(" 텍스트 파일에 처음 쓴 글입니다")
file.close( )
```

In [2]:
```
file2 = open("c:\\py_temp\\test1.txt","w")
file2.write("텍스트 파일에 두번째 쓴 글입니다")
file2.close( )
```

In [25]:
```
file3 = open("c:\\py_temp\\test1.txt","a")
file3.write("텍스트 파일에 세번째 쓴 글입니다")
file3.close( )
```

In [26]:
```
file4 = open("c:\\py_temp\\test1.txt","a")
file4.write("\n" + "텍스트 파일에 네번째 쓴 글입니다")
file4.close( )
```

In [2]:
```
import sys

print('파이썬이 재미있다고 첫번째 쓴 글입니다')
orig_stdout = sys.stdout
file5 = open("c:\\py_temp\\test1.txt","a")
sys.stdout = file5  #모니터에 출력하지 말고 file 에 출력해라

print()
print('파이썬이 재미있다고 두번째 쓴 글입니다')

sys.stdout = orig_stdout  #원래대로 변경 - 다시 화면에 출력시켜라
print('저장이 완료되었습니다')
```

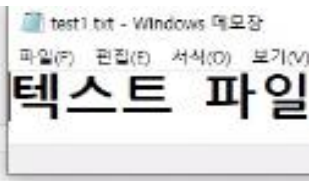
- w 모드 = 덮어쓰기

```
file = open("c:\\py_temp\\test1.txt","w")
file.write(" 텍스트 파일에 처음 쓴 글입니다")
file.close( )
```

txt 형식으로 저장하기 - 실습

- a 모드 = 추가하기(이어쓰기)

```
file = open("c:\\py_temp\\test1.txt","w")
file.write(" 텍스트 파일에 처음 쓴 글입니다")
file.close( )
```




```
file2 = open("c:\\py_temp\\test1.txt","w")
file2.write("텍스트 파일에 두번째 쓴 글입니다")
file2.close( )
```

```
file3 = open("c:\\py_temp\\test1.txt","a")
file3.write("텍스트 파일에 세번째 쓴 글입니다")
file3.close( )
```

- 내용 연결

```
file4 = open("c:\\py_temp\\test1.txt","a")
file4.write("\n" + "텍스트 파일에 네번째 쓴 글입니다")
file4.close( )
```

```
import sys

print('파이썬이 재미있다고 첫번째 쓴 글입니다')
orig_stdout = sys.stdout
file5 = open("c:\\py_temp\\test1.txt","a")
sys.stdout = file5  #모니터에 출력하지 말고 file 에 출력해라

print()
print('파이썬이 재미있다고 두번째 쓴 글입니다')

sys.stdout = orig_stdout  #원래대로 변경 - 다시 화면에 출력시켜라
print('저장이 완료되었습니다')
```

txt 형식으로 저장하기 - 실습

- 방법2 : 표준 출력 방향 바꾸기

```
file4 = open("c:\\py_temp\\test1.txt","a")
file4.write("\n" + "텍스트 파일에 네번째 쓴 글입니다")
file4.close( )
```

```
import sys

print('파이썬이 재미있다고 첫번째 쓴 글입니다')
orig_stdout = sys.stdout
file5 = open("c:\\py_temp\\test1.txt","a")
sys.stdout = file5  #모니터에 출력하지 말고 file 에 출력해라

print()
print('파이썬이 재미있다고 두번째 쓴 글입니다')

sys.stdout = orig_stdout  #원래대로 변경 - 다시 화면에 출력시켜라
print('저장이 완료되었습니다')
```

```
파이썬이 재미있다고 첫번째 쓴 글입니다
저장이 완료되었습니다
```

- 표준 출력 장치를 관리

```
import sys

print('파이썬이 재미있다고 첫번째 쓴 글입니다')
orig_stdout = sys.stdout
file5 = open("c:\\py_temp\\test1.txt","a")
sys.stdout = file5  #모니터에 출력하지 말고 file 에 출력해라

print()
print('파이썬이 재미있다고 두번째 쓴 글입니다')

sys.stdout = orig_stdout  #원래대로 변경 - 다시 화면에 출력시켜라
print('저장이 완료되었습니다')
```

```
파이썬이 재미있다고 첫번째 쓴 글입니다
저장이 완료되었습니다
```

csv, xlsx 형식으로 저장하기 - 실습

- 인덱스(Index)

```
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
```

```
1  # csv 형식으로 저장하기
2  subject.to_csv("c:\\py_temp\\test1.csv", encoding="utf-8-sig",index=False)
3
4  # xls 형식으로 저장하기
5  subject.to_excel("c:\\py_temp\\test1.xlsx" , engine='openpyxl' , index=False)
6
```

주의사항

openpyxl이 없을 경우, 'pip install openpyxl' 입력 후 실행

csv, xlsx 형식으로 저장하기 - 실습

```
print("=" *100)
print(" 이 크롤러는 대한민국 구석구석 사이트 정보 수집용 웹크롤러입니다.")
print("=" *100)

query_txt = input('1.정보를 수집할 키워드는 무엇입니까?: ')

cnt = int(input('2.몇 건의 정보를 수집할까요? :'))
page_cnt = math.ceil( cnt / 10 )
print('총 %s 건의 정보를 수집하기 위해 %s 페이지까지 이동할 예정입니다' %(cnt , page_cnt))

#Step 3. 수집된 데이터를 저장할 폴더와 파일명 지정하기
f_dir = input("3.파일을 저장할 폴더명만 쓰세요(기본값:c:\\py_temp\\):")
if f_dir == '' :
    f_dir="c:\\py_temp\\"

# 결과를 저장할 폴더와 파일명 지정
n = time.localtime()
s = '%04d-%02d-%02d-%02d-%02d-%02d' %(n.tm_year, n.tm_mon, n.tm_mday, n.tm_hour, n.tm_min, n.tm_sec)

if os.path.exists(f_dir) :
    print('지정한 %s 디렉토리가 존재하여 해당 디렉토리 아래에 파일을 생성합니다' %f_dir)
else :
    print('지정한 %s 디렉토리가 존재하지 않아 새로 생성합니다' %f_dir)
    os.makedirs(f_dir)

result_dir= f_dir+'대한민국구석구석'+'-'+s+'-'+query_txt
os.makedirs(result_dir)
ft_name = result_dir+'\\'+'대한민국구석구석'+'-'+s+'-'+query_txt+'.txt'
                 \\'+'대한민국구석구석'+'-'+s+'-'+query_txt+'.csv'
                 \\'+'대한민국구석구석'+'-'+s+'-'+query_txt+'.xlsx'
```

time.localtime()

현재 날짜와 시간을 가지고 오는 함수