<파이썬을 활용한 실전 웹크롤링>

1. 웹 문서의 구조 이해하기

* 서버 : 사용자에게 네트워크를 통해 전달하는 정보를 저장하는 대규모 컴퓨터 장치.
* 클라이언트 : 정보가 저장된 서버에 접근하는 프로그램. Ex) .exe , 브라우저
* 웹브라우저 : 사용자로부터 요청을 받아 서버에 정보를 전달하고 전달받은 정보를 사용자가 볼 수 있도록 표현하는 프로그램.

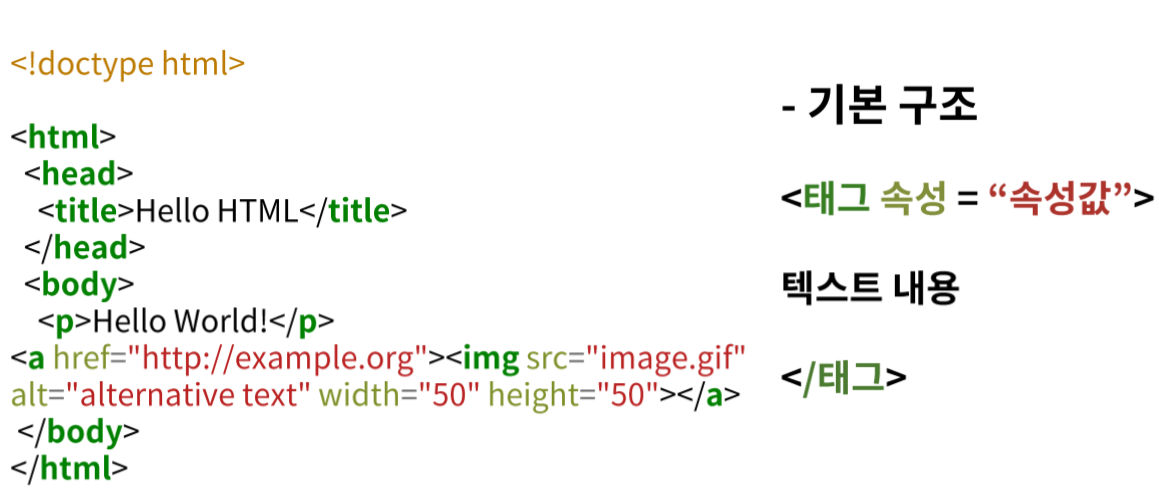
서버 - 브라우저 - 사용자

* Css : HTML문서에서 웹페이지의 배경색, 글자의 크기, 폰트 등을 정할 때 사용하는 언어
* Javascript : 정지된 웹페이지에서 사용자의 입력에 의해 동적인 표현을 수행하게 하는 언어. Ex) 인기검색어, 사이즈 변하는 광고
* HTML : Tag, attribute, CSS언어와 Javascript로 구성된 웹페이지 문서. 사용자가 요청한 정보는 웹브라우저가 서버로부터 HTML형태로 전달되며 웹브라우저에 의해 컴퓨터 언어로 된 내용을 사용자가 이해할 수 있도록 변환.
* Tag 종류

div : 공간을 의미하고 다른 element(<A>,</A>로 구분되는 수행단위)와 구분

p : 기사의 본문이나 댓글의 본문 입력 시 사용

* a : 링크를 의미하고 href 속성은 링크와 연결된 사이트 주소를 갖고 있음. <a href>형태로 사용
* td : 작업을 수행하는 element를 구분짓기 위해 사용하는 태그.
* img : 이미지를 입력할 때 사용하는 태그. src라는 속성으로 이미지 주소 표현
* span : 공간의 위치, 크기를 구분할 때 쓰는 태그.



1. Requests를 이용한 웹사이트 접근

* Requests 패키지 : 웹페이지 정보를 파이썬으로 직접 끌어오는 패키지. 크롤링 봇을 막기 위한 제어장치가 있을 경우 추가적이 코딩 필요.
* Selenium 키지 : 브라우저를 통해 웹페이지 정보를 끌어오는 패키지. 크롤링 도중 에러가 나면 브라우저를 통해 에러를 확인하고 수정 가능.

1. 실습

네이버 웹툰 평점 끌어오기

Parser : html 코드를 Python이 이해하는 객체 구조로 변환

|  |
| --- |
| #네이버 웹툰 평점 모으기  import requests #request 모듈 불러오기  ids = [648419] #url 입력값  for id\_ in ids:  response2 = requests.get('https://comic.naver.com/webtoon/list.nhn?titleId='+str(id\_)+'&weekday=mon')  # response2 변수에 url의 HTNL정보 저장    from bs4 import BeautifulSoup #  html = response2.text # response2 내용을 text파일로 html변수에 저장  html\_ = BeautifulSoup(html,'html.parser') # html\_ 변수에 html코드 구조 변환하여 저장  stars = html\_.find\_all('div',attrs = {'class' : 'rating\_type'})  # parsing한 html\_에서 태그 'div'에 해당하고 class 속성을 갖고 속성값이 'rating\_type'에  #해당하는 값을 stars에 저장  star\_list = []  for star in stars:  star\_list.append(float(star.find('strong').text))  # star text파일에서 'strong'  star\_list |

**<파이썬을 활용한 데이분석 입문>**

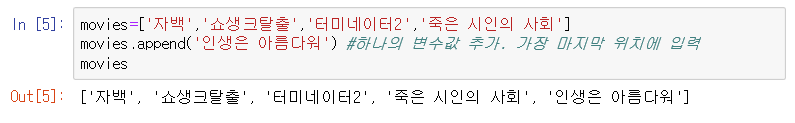
1. **List형 자료 편집**

* List형 : 대괄호를 이용하여 선언

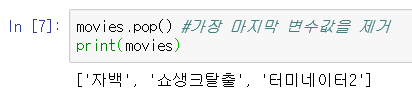
Ex) a = [1, 2, 3, 4] / a[1] = 2 / a[3] = 4

len(a) = 4 (len : 변수의 길이를 의미. 1부터 시작)

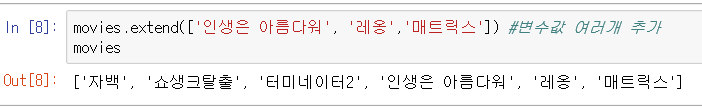
* movies.append() : list 변수값 제일 뒤에 변수값 하나를 추가



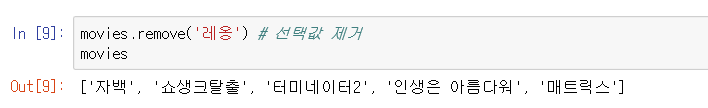
* movies.pop() : list 가장 마지막 값을 삭제(아래 결과는 두 번 실행 되었음.)



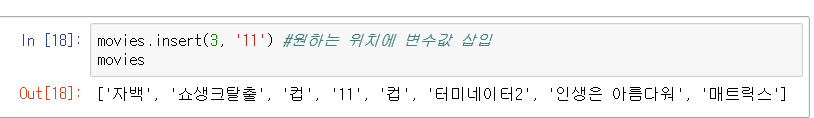
* movies.extend() : list 제일 뒤에 다수의 자료 입력



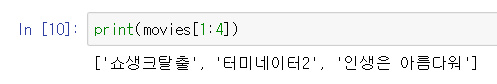
* movies.remove() : list 중 선택값 삭제



* movies.insert() : list 원하는 위치에 변수값 삽입

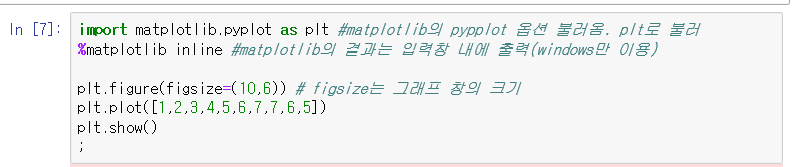


* 슬라이싱 : list 값의 범위를 편집가능([1]~[3]까지)

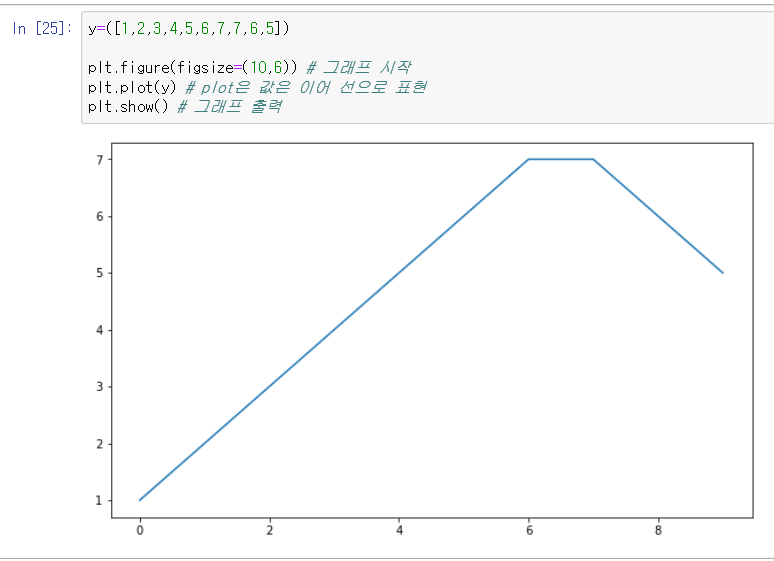


1. **Matplot**

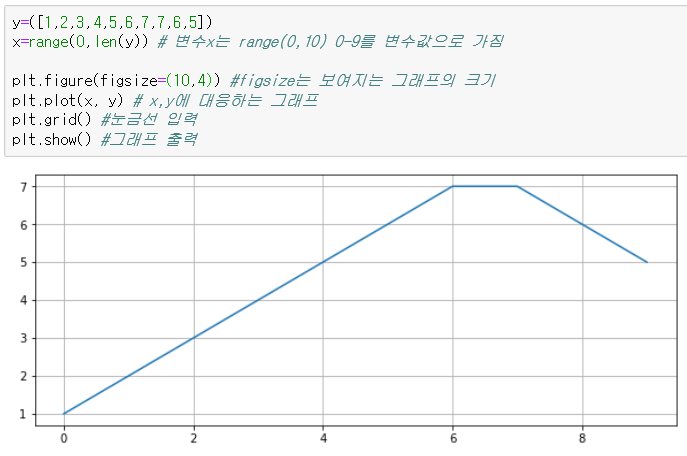
* 2D그래프를 그리는 모듈. Pyplot에 일반적인 명령 포함되어 있음



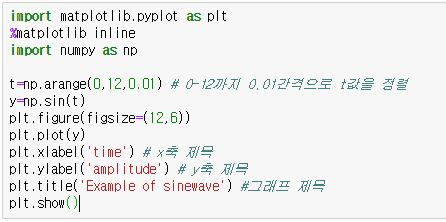
* 임의 값을 입력한 그래프. Y축 값은 입력값, x축 값은 순서대로 맞춰져.

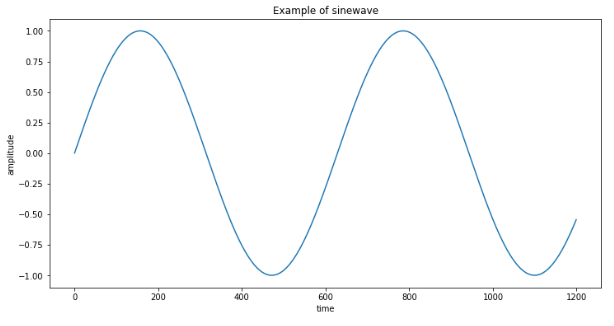


* 격자, x/y의 대응함수 출력

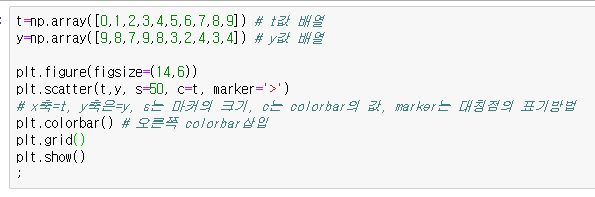


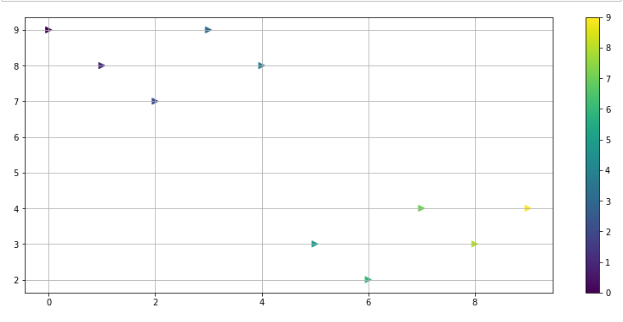
* 축 제목 설정

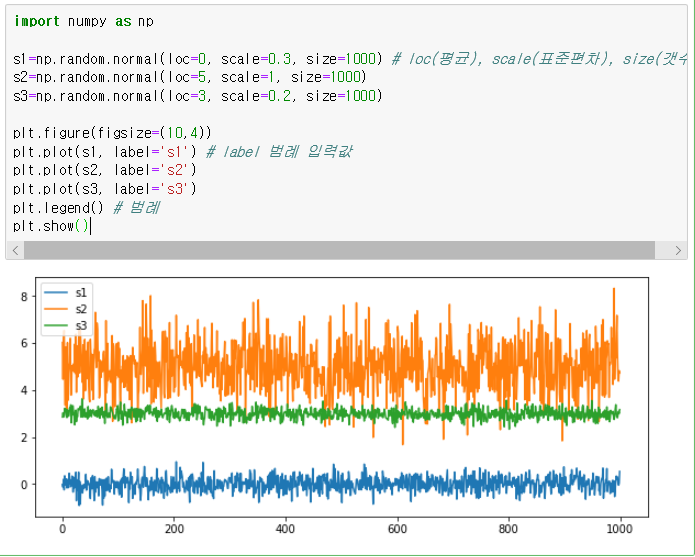




* 산점도(Scatter plot)



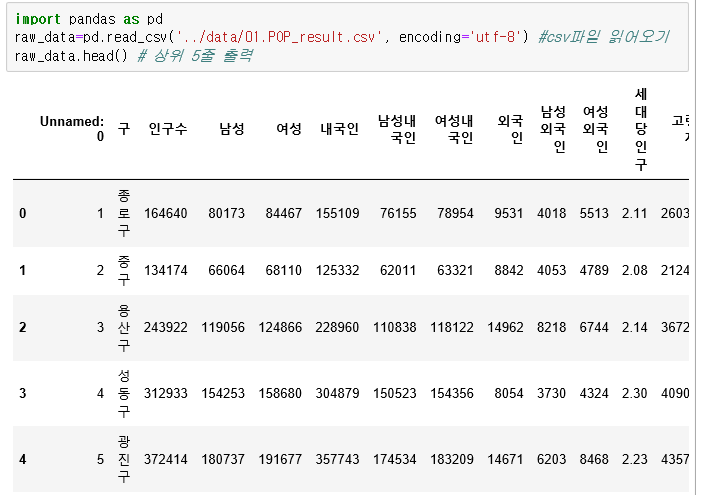


* plt.plot (lw(그래프 선의 굵기), colors=’red’(그래프 색상)) 입력 가능
* numpy모듈의 arange명령 이용한 그래프
* 

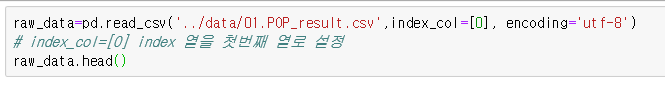
1. **Pandas모듈을 이용하여 그래프 그리기**

* csv파일 읽어오기

변수명 = pd.read\_csv(‘파일 저장위치’, encoding=’utf-8’) 한글을 읽을 수 있는 utf-8로 인코딩

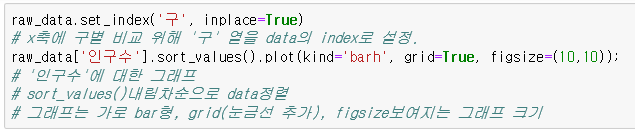


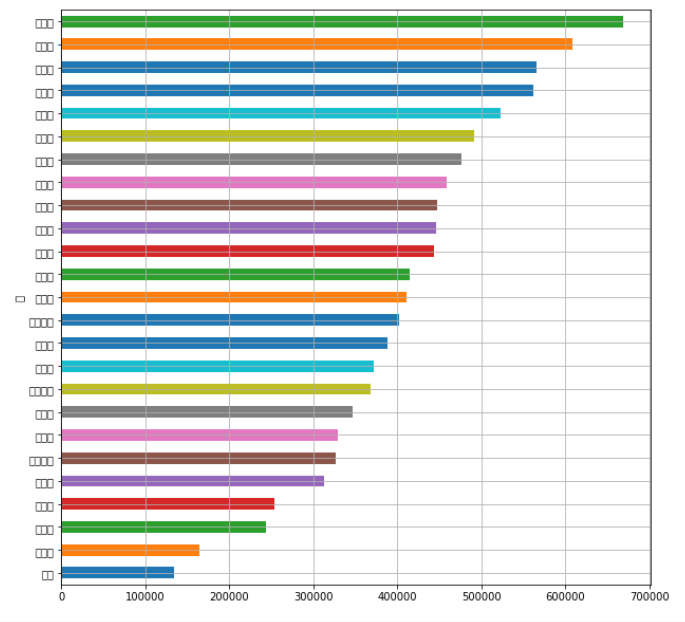
* index변경 (열의 첫번째 값 index열로 설정)



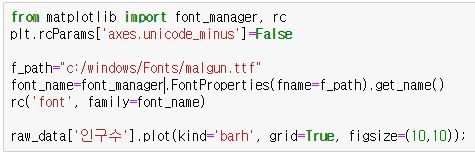


* data 그래프 그리기

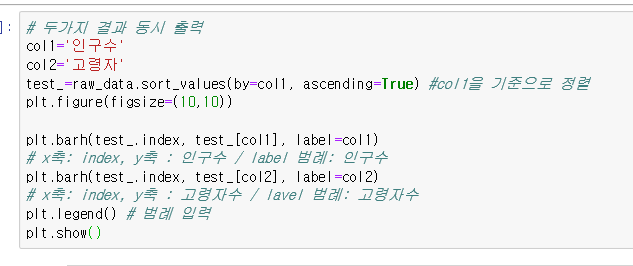


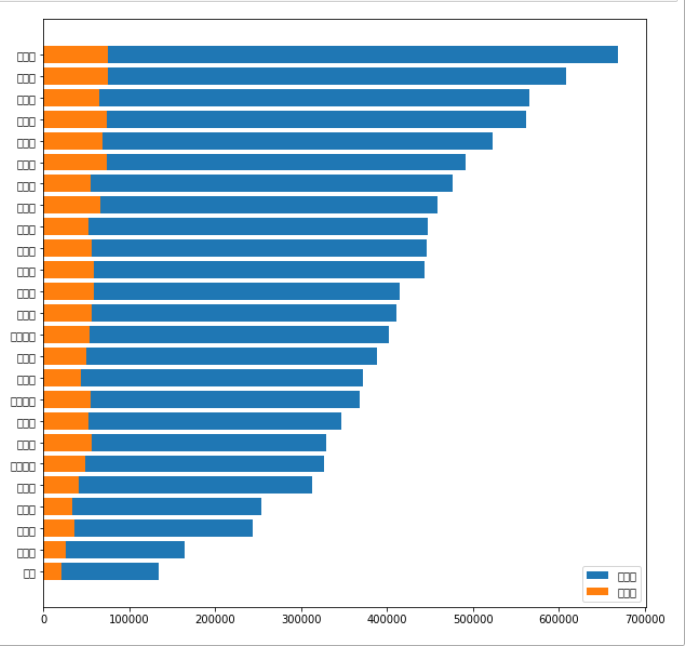


* 한글 폰트가 없어 한글이 그래프에 나타나지 않아 폰트 저장 필요. (아래 내용은 검색)



* plot함수를 이용하여 그래프 그리기





1. **반복문/조건문**
2. for문

* 사용자가 입력한 횟수만큼 횟수만큼 for문을 반복 실행
* 같은 결과를 내는 for문 예시

|  |  |  |
| --- | --- | --- |
| x=[1,2,3,4,5]  y=[]  *#비어있는 list형 y변수*  for n in x:  y.append(n\*\*2)  *#반복의 결과를 y에 입력*  Y  >>> 4, 9, 16, 25] | x=[1,2,3,4,5]  y=[n\*\*2 for n in x]  *#x값을 n에 입력하여 n^2의 결과를 y의 list형으로 저장*  y | import numpy as np  x=np.array(x)  *#앞서 입력된 x값의 배열을 생성*  y=x\*\*2  *#x^2값은 y에 저장*  y |

|  |  |  |
| --- | --- | --- |
| x=[1,2,3,4,5]  y=[]  for n in x:  if n%2 != 1:  #2로 나눈 나머지 1이 아닐 때  y.append(n\*\*2)  # y에 n^2값 입력  y  >>>[4, 16] | x=[1,2,3,4,5]  y=[n\*\*2 for n in x if n%2!=1]  y | x=[1,2,3,4,5]  y=[n\*\*2 for n in x if not bool(n&1)]  y |

1. **피벗테이블**

**:** 데이터 중 숫자 값에 대하여 필터링, 정렬, 재편성, 계산, 요약한 후 특정 정보를 추출

|  |  |
| --- | --- |
| * 최초 rawdata | * 이름으로 재정렬한 데이터   (default 값은 평균) |

1. Name, Rep, Manager 순서를 기준으로 테이블 계산



1. 원하는 내용에 대해 재정리

|  |
| --- |
| pd.pivot\_table(df,index=["Rep","Manager"],values=["Price"], aggfunc=[np.sum, len])  *# values는 원하는 숫자열, aggfunc는 함수 입력 옵션, np.sum은 values입력값을 합계로 반환*  *len은 해당 값이 수(빈도)* |

|  |
| --- |
| pd.pivot\_table(df,index=["Rep","Manager"],values=["Price"],columns=["Product"],aggfunc=[np.mean,np.sum], fill\_value=0, margins=True)  *# fill value는 가격이 none인 값을 0으로 변경. margins은 열 값에 따라 평균 또는 합계 계산.* |

|  |  |
| --- | --- |
| import pandas as pd  crime\_raw\_data=pd.read\_csv('../data/02\_Crime\_data.csv', encoding='euc-kr')  crime\_raw\_data.head()  *#파일 불러오기* | import numpy as np  crime\_station=crime\_raw\_data.pivot\_table(crime\_raw\_data, index=["구분"],columns=["죄종","발생검거"], aggfunc=[np.sum])  crime\_station.head()  # index를 ‘구분’, 열값을 ‘죄종’ ‘발생검거’ 재정렬 |

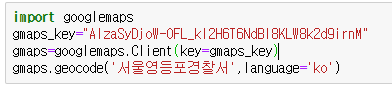
|  |
| --- |
| crime\_station.columns=crime\_station.columns.droplevel([0,1])  *#droplevel : 상위 columns 0,1 제거 (sum, 건수 제거)*  tmp=crime\_station.columns.get\_level\_values(0)+crime\_station.columns.get\_level\_values(1)  # columns의 0번째, 1번째 문자열을 index값으로 결합하여 변수 tmp에 저장  crime\_station.columns=tmp  *# 변수 tmp를 열의 index값으로 반영*  crime\_station.head() |

6. google API이용하기

- 구별로 분류해야 하기때문에 구분된 경찰서의 주소정보 필요

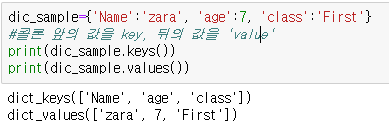
- 구글 maps API에서 ID와 PW발급

- API를 이용해 받은 자료는 dic형 구조로 전달

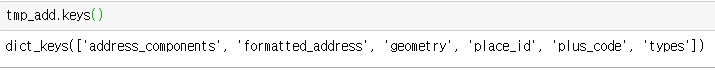




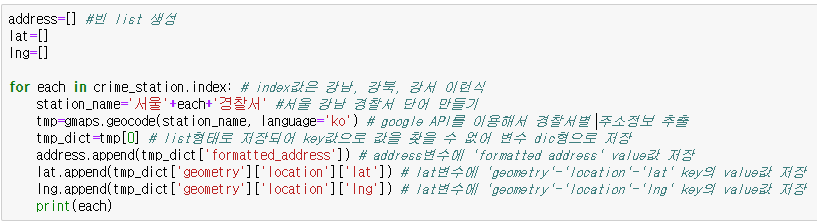
* Dic형 : key를 이용해서 value를 찾을 수 있는 구조 { ‘ ’ : ‘ ‘ }

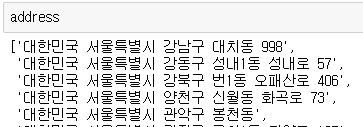


* 서울영등포경찰서의 key값 결과



* 자료의 구분 index에 따른 지역 주소 저장

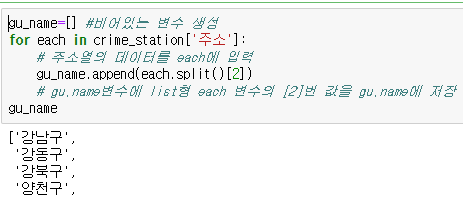




* 검색한 주소 결과 데이터에 추가



* 주소명에서 ‘구’ 명칭만 떼어서 저장



* 변수값 입력



* 파일저장



