Lab stock_data - 주식 데이터 모아오기 (Stock Data Crawler)

Introduction

PDF 파일 다운로드

이제까지 모든 Lab은 프로그램 실행을 위한 사람이 직접 데이터를 입력했다면, 이번 숙제는 미국의 구글 금융 서버에서 제공하는 주식 데이터를 사용합니다. 즉 처음으로 실제 데이터를 가지고 프로그램을 작성하는 것입니다. 우리가 다룰 데이터는 CSV 형태로 제공되고 있습니다. 간단히 인터넷 주소인 URL 주소만 입력하면 데이터를 가져올 수 있습니다.
이번 숙제는 상당히 재밌으면서도 조금 어려울 것입니다. 일단 데이터를 가져오는 것이 상당히 생소할 것이고, 필요한 데이터만 추출한다는 개념도 상당히 새로울 것입니다. 그럼에도 불구하고 처음으로 외부 데이터를 만지는 의미있는 프로그램이므로 기쁜 마음으로 도전하길 바랍니다.

숙제 파일(lab_10.zip) 다운로드

https://github.com/TeamLab/introduction_to_python_TEAMLAB_MOOC/blob/master/lab_assignment/lab_10/lab_10.zip

다운로드를 위해 `View Raw` 또는 `Download` 버튼을 클릭합니다. 또는 아래 다운로드 링크를 클릭하면 자동으로 다운로드가 됩니다.
Lab stock_data - 다운로드
다운로드 된 `lab_10.zip` 파일을 작업 폴더로 이동한 후 압축해제 후 작업하시길 바랍니다.
압축해제 하면 폴더가 `linux_mac` 과 `windows` 로 나눠져 있습니다. 자신의 OS에 맞는 폴더로 이동해서 코드를 수정해 주시기 바랍니다.

Stock Data Overview

우리가 데이터를 가져올 곳은 미국의 구글 금융 서버입니다. 역시 구글 신은 다양한 데이터를 제공하고 있습니다(우리에게 제공할 숙제 데이터도 관리하면서요^^.) 데이터는 CSV로 제공됩니다. 간단히 아래의 URL을 여러분의 웹 브라우저에 입력봅시다.

> http://finance.google.com/finance/historical?q=KRX:005930&startdate=2013-01-01&enddate=2015-12-30&output=csv

사실 원래 숙제는 야후 데이터 서버에서 데이터를 가져오는 것이였는데 급작스럽게 지난 5월 부터 야후가 데이터를 제공해주지 않으면서 구글로 서버를 교체했습니다. 그와 동시에 숙제 내용도 약간의 변경이 있었습니다.
위 주소의 메인 서버 주소는 `http://finance.google.com/finance/historical?` 이며 그 뒤로 나오는 `q`, `strartdate`, `enddate` 등의 값에 입력에 따라 다양한 주식 정보 데이터를 다운로드 받을 수 있습니다. 설정할 수 있는 값은 아래와 같습니다.

키	설명	값	비고
q	종목(심볼)	KRX:005930	한국 Samsung Electronics Co. Ltd.
startdate	시작 년-월-일	2017-01-30	
enddate	종료 년-월-일	2017-04-30	
output	다운로드 데이터 Type	csv	csv 타입 데이터만 지원

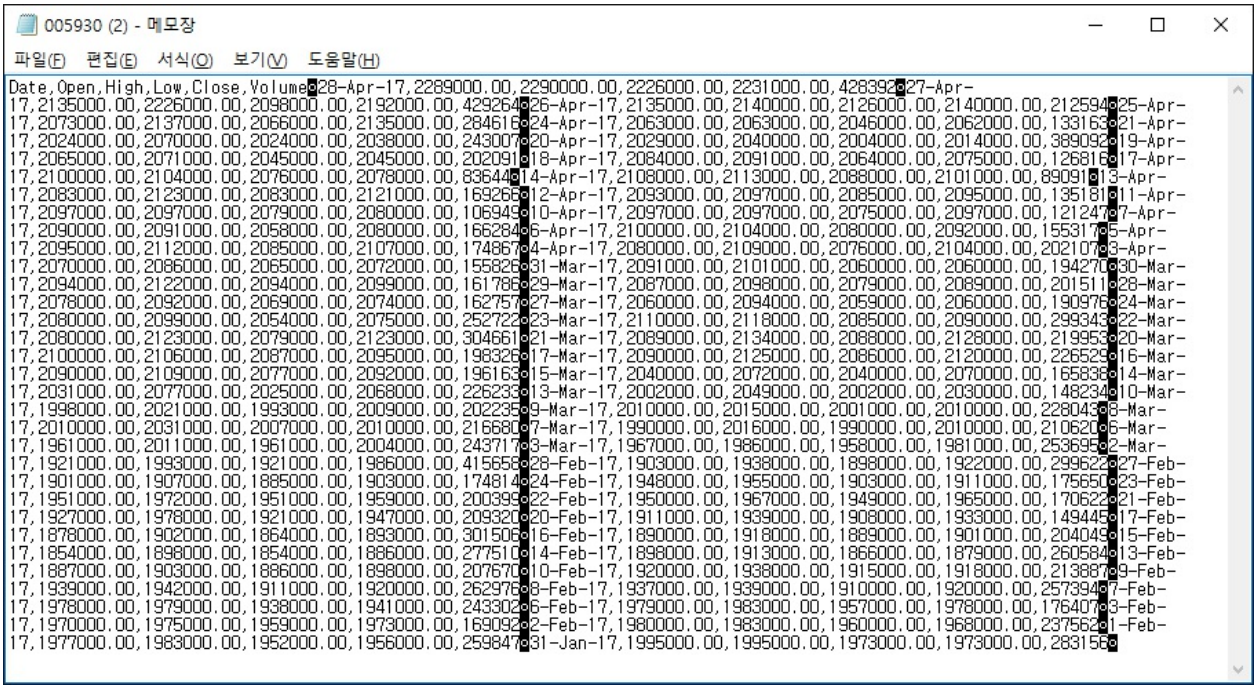
위의 값을 자유롭게 바꾸면 다양한 주식 정보를 구할 수 있습니다. 예를 들면, `q=KRX:005380` 를 입력한다면 현대 자동차의 주식정보를 다운로드 받을 수 있습니다.

기업들은 아래와 같이 다양한 기업들을 선택할 수 있고, 기업코드만 안다면 우리나라 전체 기업의 주식 정보를 기간을 정해서 다

운로드 받을 수 있습니다.

종목 심볼	종목명	종목 심볼	종목명
005930	삼성전자	032830	삼성생명
005380	현대차	051910	LG화학
005490	POSCO	009540	현대중공업
012330	현대모비스	017670	SK텔레콤
000660	SK하이닉스	105560	KB금융
035420	NAVER	096770	SK이노베이션
005935	삼성전자우	023530	롯데쇼핑
015760	한국전력	086790	하나금융지주
055550	신한지주	000810	삼성화재
000270	기아차	066570	LG전자

해당 주소를 웹 브라우저에 넣으면 table.csv라는 파일을 다운로드 받게 되고, 일반적으로 메모장을 통해서도 열 수 있습니다. 실제 파일을 열면 아래와 같이 보일 것입니다.



조금 복잡해 보이지만, 간단히 설명을 하자면 본 데이터의 첫 번째 줄인 Header Filed는 `Date`, `Open`, `High`, `Low`, `Close`, `Volume`, `Adj Close`로 구성되어 있고, 다음 줄 부터 데이터의 실제 값이 존재한다. 한줄이 띄어져야 할 거 같은 곳에는 모두 동그라미 검은색 네모에 동그라미가 들어가 있는데 이는 Encoding 문제로 리눅스의 줄바꿈 기호라고 생각하면 된다. 리눅스에서 데이터를 다운로드 받을려면 다음과 같이 입력한 후, vi editor로 열어볼 수 있다.

```
bash wget http://finance.google.com/finance/historical?q=KRX:005930&startdate=2013-01-01&enddate=2015-12-30&output=csv
```

각 header filed의 의미는 아래와 같습니다.

Open	Close	High	Low	Volume

시작가	종가	최고가	최저가	거래량

자 이제 데이터에 대한 기초적인 이해가 끝났으니 숙제를 시작해 보도록 합시다.

stock_data_crawler.py 파일 Overview

`atom editor`로 `stock_data_crawler.py`을 열어 전체적인 개요를 봅시다. `atom stock_data_crawler.py`명령으로 파일을 열어보면 `main` 함수와 여러개의 함수들이 존재할 것이다. 각 함수들은 여러분께서 직접 작성해서 제출해야 하는 함수이고, `main` 함수는 실제 주식 정보를 요청하여 프로그램을 실행하는 함수이다. 각 함수의 구현 내용은 아래와 같다.

함수	설명
get_stock_data	url_address를 Input 변수로 넣으면 구글 서버에 요청하면 해당 정보를 다운로드 받은 후 Two Dimensional List 변환하여 반환함
get_header_data	get_stock_data 함수의 반환 값을 Input 변수로 넣으면 Header Filed에 해당하는 값만 추출하여 list로 반환함
get_attribute_data	get_stock_data 함수의 Return 값, 추출하고자 하는 Header Field의 이름, 추출하고자 하는 년도, 월을 Input 변수로 입력받으면 Date Field와 해당 조건의 값만 추출하여 list로 반환함
get_average_value_of_attribute	get_stock_data 함수의 Return 값, 추출하고자 하는 Header Field의 이름, 추출하고자 하는 년도, 월을 Input 변수로 입력받으면 추출된 값의 평균을 계산하여 Float Type으로 반환함
write_csv_file_by_result	get_stock_data 함수의 반환 값 또는 get_attribute_data 함수의 반환값, 생성하고자 하는 파일 이름을 String Type의 Input 변수로 넣으면, 입력된 List 값이 들어 있는 파일을 생성함, 단 이때 파일 Type은 UTF8로 생성하거나, 입력하지 말 것(CP949로는 제출 불가)
separate_user_query	사용자의 입력 값을 `,`을 기준 값으로 list로 변환한 후 반환함, 이때 반드시 각 값들은 앞뒤 빈칸이 제거된 후 list에 할당 되어야 함 ex) 입력예시: SAMSUNG, 2014-12, Open, ALL

개별 함수 설명

이번 랩은 추가 설명이 꽤 필요하기 때문에 필요한 함수들에 대해서 추가 설명을 하도록 한다.

> get_stock_data

get_stock_data 함수에서는 아래 코드 처럼 urllib 이라는 모듈을 호출하여 데이터를 요청할 수 있다.

```
url_address = 'http://finance.google.com/finance/historical?q=KRX:005930&startdate=2013-01-01&enddate=2015-12-30&output=cs
r = urllib.request.urlopen(url_address)
stock_data_string = r.read().decode("utf8")    # String Type으로 다운로드 받은 데이터
print(stock_data_string[:100])
```

위 코드의 결과 값들은 `'Date,Open,High,Low,Close,Volume\n28-Apr-17,2214.36'` 와 같이 String Type으로 출력이 될 것이다.
`r=urllib.request.urlopen`는 특정 url 주소에서 데이터를 불러오는 구문이고, `r.read()` 해당 url에 값을 byte 값으로 호출 하는 함수이다. 호출된 byte 값을 string type으로 변환하기 위하여 `decode("utf8")`을 쓰며 여기서 `utf8`은 리눅스에서 흔히 사용하는 문자 인코딩 표준이며, 윈도우에 경우 `cp949`를 사용한다.
반환된 String 값은 `,`로 필드의 값이 구분되고, `\n`로 row가 구분된다.

> get_attribute_data & get_average_value_of_attribute

본 함수들은 `get_stock_data`에서 반환된 주식 정보에서 특정 필드의 값을 특정 기간으로 한정하여 추출하기 위한 함수들이다. 두 함수는 모두 아래와 같은 input 변수를 가진다.

```
def get_attribute_data(stock_data, attribue, year=None, month=None):
 pass
```

위 4가지 Input 변수 중 `stock_data`와 `attribute`는 각각 `get_stock_data`의 반환값과 추출하고자 하는 `header`이름을 의미한다. `header` 의 이름은 다음과 같이 `Open`, `Close`, `High`, `Low`, `Volume` 5개의 값이 존재한다. `stock_data` 와 `attribute` Input 변수는 반드시 입력되어야 한다.
year와 month의 경우는 반드시 입력하지 않아도 상관없는 값들이다.
year와 month가 입력되지 않으면 `stock_data`에서 `attribute`에 지정된 필드의 값만 추출한다. 추출되는 값은 반드시 `Date`와 `attribute`의 header field 이름이 함께 포함되어야 한다. 또한 `month`에 값이 지정되었을 경우, `year`에 반드시 값을 할당 하여야 한다. `year`에만 값을 지정하였을 경우에는 해당 년도에 해당하는 값만 반환한다. 실제 두 함수는 아래와 같이 사용가능하다.

```
import stock_data_crawler as sdc
url = 'http://finance.google.com/finance/historical?q=KRX:005930&startdate=2013-01-01&enddate=2015-12-30&output=csv'
stock_data = sdc.get_stock_data(url)
header = sdc.get_header_data(stock_data)
print(sdc.get_attribute_data(stock_data, "High"))
print(sdc.get_attribute_data(stock_data, "Open", 2014))
print(sdc.get_attribute_data(stock_data, "Close", 2013, 12))
print(sdc.get_average_value_of_attribute(stock_data, "High", 2014, 12))
```

한 가지 주의할 점은 `year`와 `month`의 input 변수가 모두 int type 이라는 것이다. stock_data의 date 부분 값의 경우 `2014-01` 처럼 string type으로 지정되어 있는데 int type으로 들어가기 때문에 type 변환을 환후 값을 비교해줘야 한다.

main 함수 수정하기

위의 함수들을 작성하고 나면 `main`함수를 수정해야 한다. `main`함수의 기본 template은 아래와 같다.

```
def main():
    print("Stock Data Crawler Program!!")
    user_input = 999
    url = 'http://finance.google.com/finance/historical?q=KRX:005930&startdate=2013-01-01&enddate=2015-12-30&output=csv'
    # ===Modify codes below=============

    # ==================================
```

본 숙제에 경우 url이 지정되어 있기 때문에 항상 삼성전자의 2013년 1월부터 2015년 11월 현재까지 데이터를 가져온다. 메인함수는 다음과 같은 규칙이 있다.

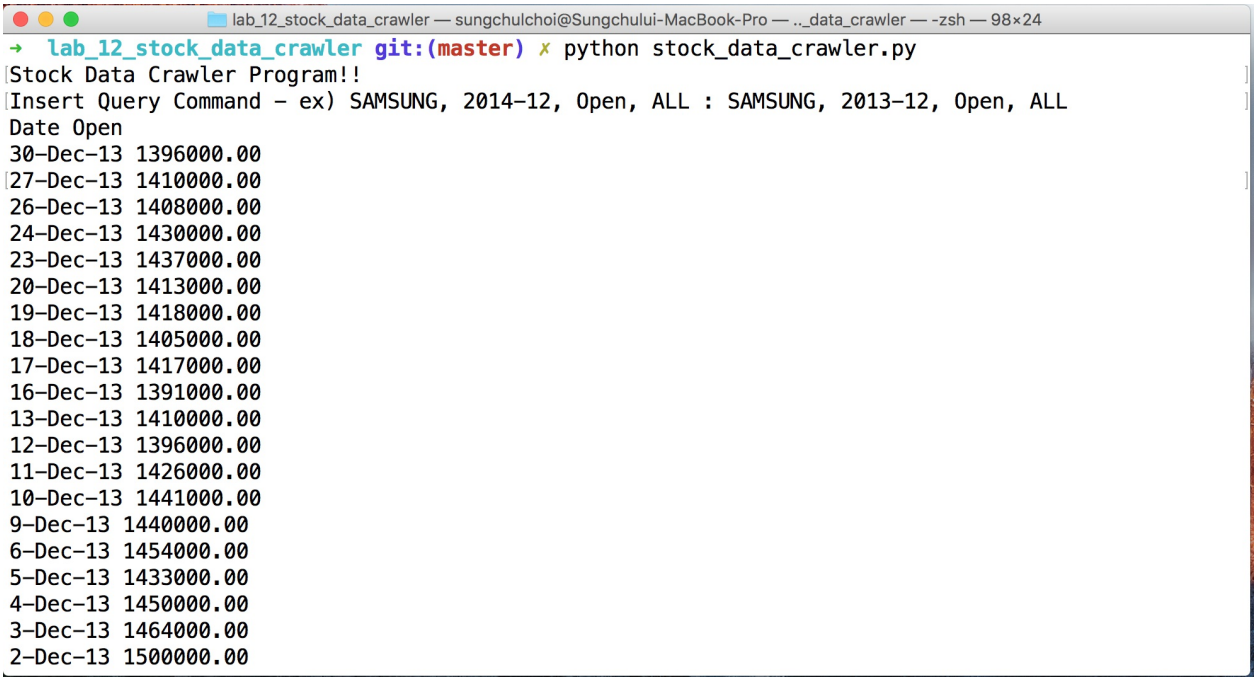
`main` 함수는 다음과 같은 규칙을 가진다.

1. 사용자가 0을 입력하면 종료된다.
2. 사용자는 아래와 같은 형태로 명령을 입력하고 입력된 값에 따라 결과물을 출력해 준다. 첫 번째 `SAMSUNG`은 고정된 값으로 수정이 필요없고, `2014-12`는 추출하고자 하는 데이터의 년-월, `Open`은 추출할 필드이름, `ALL`은 추출 유형이다. 각 추출 유형에 대한 설명은 아래 표와 같으며, 추출 유형은 대소문자를 구분하지 않고 처리할 수 있어야 한다.

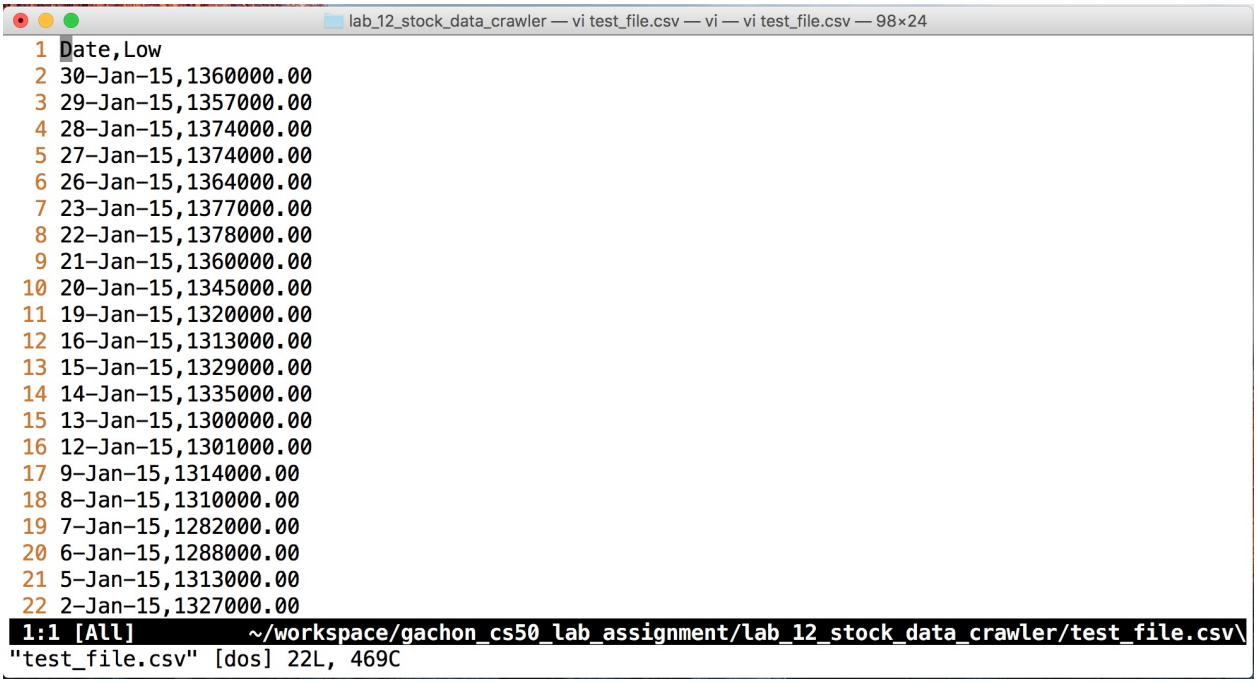
```
SAMSUNG, 2014-12, Open, ALL
```

추출 유형	사용예시	설명
ALL	SAMSUNG, 2014-12, High, ALL	조건의 맞는 모든 데이터를 모두 추출하여 화면에 표시, get_attribute_data 함수를 사용함
MEAN	SAMSUNG, 2014-12, Close, MEAN	조건의 맞는 모든 데이터를 모두 추출하여 평균을 계산하여 화면에 표시 get_average_value_of_attribute 함수를 사용함
FILE	SAMSUNG, 2014-12, Open, FILE, test.csv	조건의 맞는 모든 데이터를 모두 추출한 후 파일로 저장하는 명령어 write_csv_file_by_result 사용함, 유의할점은 FILE을 입력할 경우, test.csv 처럼 파일명을 FILE 다음에 입력해 주어야 함

실제로 작성된 프로그램의 실행화면과 생성된 파일의 모습은 다음과 같다. 숙제를 제출할 때는 반드시 utf8 형태로 파일을 생성할 것을 요청드립니다.




```
Insert Query Command - ex) SAMSUNG, 2014-12, Open, ALL : Samsung, 2014-12, Low, Mean
1298047.619047619
Insert Query Command - ex) SAMSUNG, 2014-12, Open, ALL : Samsung, 2015-01, Low, FILE, test_file.cs
v
test_file.csv file created
Insert Query Command - ex) SAMSUNG, 2014-12, Open, ALL : 0
Good Bye
```



###숙제 template 파일 제출하기 (윈도우의 경우)

1. `windows`+`r`를 누르고 cmd 입력 후 확인을 클릭합니다.
2. 작업을 수행한 폴더로 이동 합니다.
3. 밑에 명령어를 cmd창에 입력합니다.

```
install.bat
submit.bat [YOUR_HASH_KEY]
```

숙제 template 파일 제출하기 (Mac or Linux)

1. 터미널을 구동합니다.
2. 작업을 수행한 디렉토리로로 이동 합니다.
3. 밑에 bash창을 입력합니다.

```
bash install.sh
bash submit.sh [YOUR_HASH_KEY]
```

> backend.ai 서비스의 업데이트에 의해 실행전 반드시 `bash install.sh` 또는 `install.bat` 수행을 바랍니다.

제대로 작성했다면 아래와 같은 메세지가 뜰 것입니다. 실행시 이전과 다르게 상당히 오랜시간이 걸릴 것입니다. 그 이유는 코드를 테스트 하는 프로그램이 내부적으로 데이터를 다운로드 받고 하는 파일로 만드는 시간이 있기 때문이다 조금 기다려서 실행하면 결과가 나오니 중간에 끄거나 하지말기 바랍니다.
또한 안된다고 계속 시도하지 마시고 에러가 날 경우는 수정을 하고 제출해주시기 바랍니다. 구글 서버와 연결되기 때문에 너무 많은 요청이 가면 봇 으로 취급당할 수도 있습니다.

```
-------------------- | ---------- | --------------------
       Function Name |    Passed? |             Feedback
-------------------- | ---------- | --------------------
 separate_user_query |       PASS |             Good Job
     get_header_data |       PASS |             Good Job
get_average_value_of_attribute |       PASS |             Good Job
write_csv_file_by_result |       PASS |             Good Job
```

```
   get_attribute_data |       PASS |             Good Job
       get_stock_data |       PASS |             Good Job
                 main |       PASS |             Good Job
--------------------- | ---------- | --------------------
```

Next Work

이제 이쯤되면 "나도 파이썬 조금은 할 줄 알아요" 라고 말할 수준은 된거 아닌가 싶습니다. 상당히 어려운 숙제임에도 불구하고 잘 따라와 줘서 고맙습니다. 여전히 많은 시간과 에너지가 소비되었겠지만, 이젠 꽤 보람차게 따라하고 있을 거 같습니다. 앞으로도 좋은 코드 많이 작성하기 바랍니다. 고생하셨습니다. Good Luck.

> **Human knowledge belongs to the world** - from movie 'Passw ord' -