**Data Focused Python**

**Homework 2**

**Mini 1 2020**

***Due at 11:59 pm on Tuesday, Sept. 15, 2020***

***You will lose 1 point every 6 minutes after that time***

1. (100 points) **Handling Idiosyncratically Formatted Data**

In Python terms, the purpose of this part of the homework is to gain experience with input and output files, variables, decisions, loops, string processing (including slices and formatting), conversions (string to number or number to string), and the like.

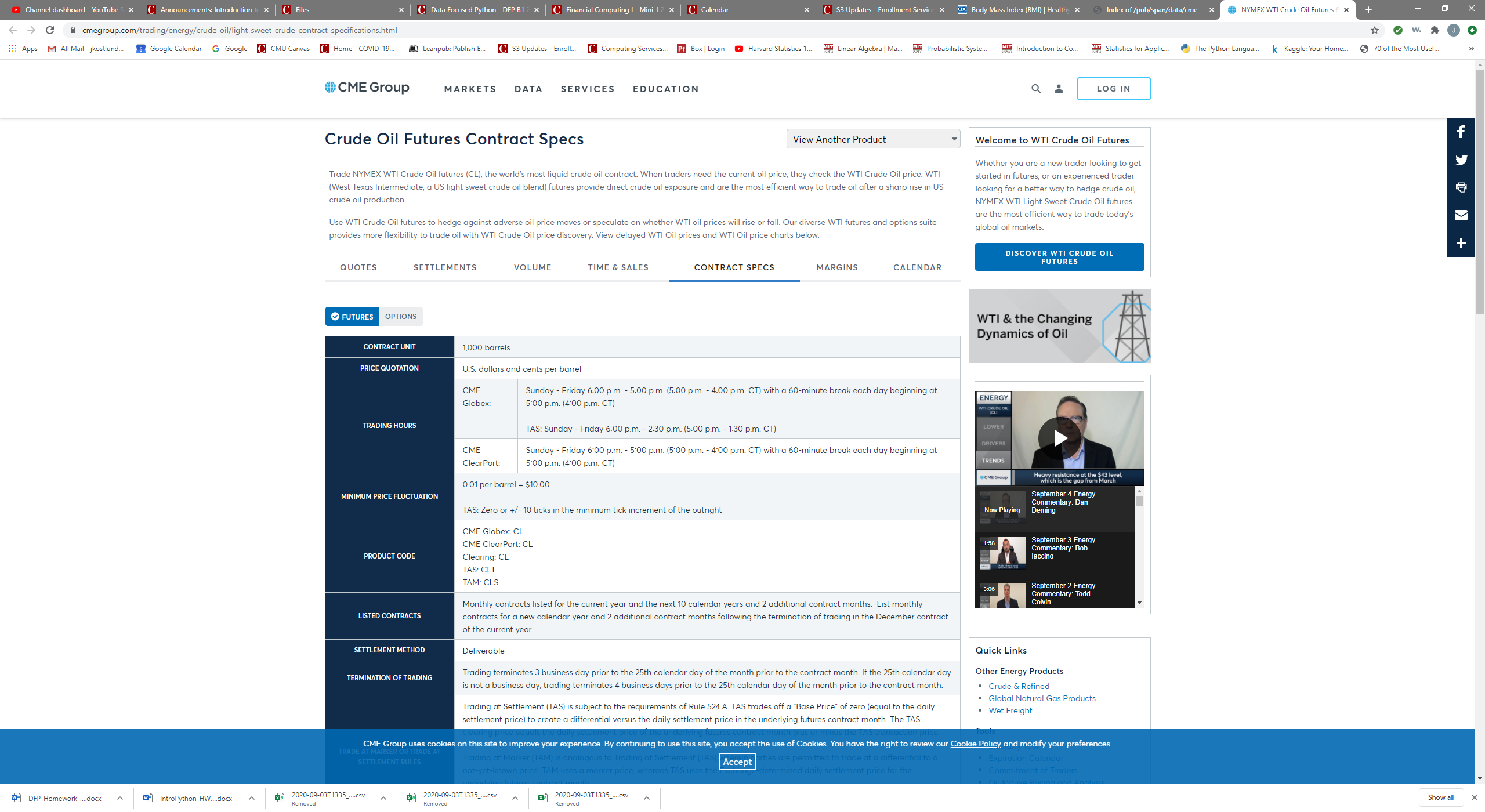
Some data sources are in convenient formats (CSV, JSON, HTML, XML, and so forth), and others are mostly unformatted (documents, email messages, system and web logs, etc.). There are also idiosyncratically formatted files, with their own strange formats often made up years in the past, before standards like CSV or JSON were invented. You must be able to handle *all* of these kinds of data sources.

Commodity futures and option contracts of many kinds are traded on NYMEX, owned by CME Group. Each evening of each trading day, sometime between about 6:00 pm and 8:00 pm Central Time, a SPAN (Standard Portfolio Analysis of Risk) file is posted to <ftp://ftp.cmegroup.com/pub/span/data/cme> containing information about the day’s trading. For a given day, the name of this file is cme.*YYYYMMDD*.c.pa2.zip, where *YYYYMMDD* is the 8-digit year, month, and day of the file. Files for months prior to the current month are moved into an archive subdirectory.

Download the zipped SPAN file for Friday, Sept. 4, 2020, **cme.20200904.c.pa2.zip**. Unzip, then display this SPAN file. You will see that it is an enormous text file with its own unique format, unfortunately *not* something simple and convenient like CSV or XML or JSON.

The *settlement prices* (in U.S. dollars) contained in the SPAN file are used to *mark to market* each trader’s account, so that gains/losses can be credited/debited each day to reduce the risk of counterparty default (a trader who has to cover modest losses each day is less likely to default than a trader who has to cover huge losses at the end of a year, say). Your job is to extract these settlement prices, as well as contract expiration dates (last trading dates), for one of the globally most heavily traded energy contracts: West Texas Intermediate (WTI) Crude Oil.

To learn about WTI Crude Oil futures contract details, see: <http://www.cmegroup.com/trading/energy/crude-oil/light-sweet-crude_contract_specifications.html>



Notice that the CME Globex Product Code is **CL**; you will need this information for scanning the SPAN file. Using other tabs at the top of this web page, you can see current price quotes, recent settlement prices, trading volume, etc. If you click the **Options** button, just to the right of the **Futures** button near the upper left, you will see information about options contracts based on the underlying futures contracts. There are about two dozen different types of option contracts for WTI Crude Oil futures; we are interested in the American Options. When you look at the contract specifications, you will discover that its Product Code is **LO**.

Write a Python program named **hw2.1.py** that reads **cme.20200904.c.pa2** as its input file, and produces **CL\_expirations\_and\_settlements.txt** as its output file. The output should be in exactly this form:

Futures Contract Contract Futures Options Options

Code Month Type Exp Date Code Exp Date

------- -------- -------- -------- ------- --------

CL 2020-11 Fut 2020-10-20

CL 2020-12 Fut 2020-11-20

*… and so forth, through contract month 2023-12 …*

CL 2020-11 Opt LO 2020-10-15

CL 2020-12 Opt LO 2020-11-17

*… and so forth, through contract month 2023-12 …*

Futures Contract Contract Strike Settlement

Code Month Type Price Price

------- -------- -------- ------ ----------

CL 2020-11 Fut 40.15

CL 2020-12 Fut 40.59

*… and so forth, through contract month 2023-12 …*

CL 2020-11 Call 0.50 39.31

CL 2020-11 Put 0.50 0.01

CL 2020-11 Call 1.00 38.81

CL 2020-11 Put 1.00 0.01

*… and so forth, through contract month 2023-12 …*

***Do not try to create a better output format***: it needs to be very easy for us to compare your output to our solution output, and to other students’ outputs. We will take off points if your output format varies too much from what is shown above.

Our output format takes into account the order in which records appear in the SPAN file, so you don’t have to remember or accumulate much information as you go. In particular, you do not have to process the contents of **cme.20200904.c.pa2** more than once in order to create the table.

***Do not*** include contract months earlier than 2020-11, or later than 2023-12.

Since there are many, many strike prices for options on futures contracts, the output file is going to be very long, but not nearly as long as the SPAN file itself.

Fortunately, there is documentation online that describes the contents of CME SPAN files. If you Google for “cme span pa2 file format” you will find a page named “Risk Parameter File Layouts for the Positional Formats – SPAN…”. You will want to look at Type “B” Records, Expanded Format, and Type “8” Records, Expanded Format, to learn how to obtain the contract name, type, month, expiration date, strike, and settlement prices that you need.

**Here are detailed examples of the structure of the Type B and Type 81 records for a different energy product, Natural Gas, for an earlier year.**

B NYMNG        FUT201810            000000000900000001100030000330000000021643800000

001000020180926NG          00000000         0010000000000000 00 00 010000000000P 00

According to the Type B Expanded documentation at <https://www.cmegroup.com/confluence/display/pubspan/Type+B+-+Expanded>

the Record ID (record type) is "B ",

the Exchange Acronym is "NYM"

the **Commodity Code** is "NG        " (for Natural Gas) **# futures code**

the **Product Type Code** is "FUT" (futures contract) **# contract type**

the **contract month** is "201810" (October, 2018) **# contract month**

and the **Expiration (Settlement) Date** is "20180926" **# fut exp date**

You need to extract and reformat the **Commodity Code, Product Type Code, Contract Month, and Expiration Date for Crude Oil (CL) records for the top half of the first table.**  In your output table, these will be the Futures Code, Contract Type (display "Fut" rather than "FUT"), Contract Month, and Futures Exp Date, respectively

B NYMON        OOF201810   201810   002093720900000001100030000330000000021369900

000001000020180925NG        M 00000000N0280500+0010000000000000 00 00 010000000000P 00

The Commodity Code is "ON        " ("LO        " for WTI crude) # **options code**

the Product Type Code is "OOF" (option on futures) **# contract type**

the contract month is "201810**" # contract month**

the expiration date is "20180925" # **options exp date**

The Underlying Commodity Code for this option is "NG        "

This provides what you need to extract and reformat for the bottom half of the first table.  These correspond to the Options Code, Contract Type ("Call" or “Put”), Contract Month, and Options Exp Date.  The Futures Code for ON options is NG; **the Futures Code for LO options is CL**; you can know this from looking at the CME contract specification.

For the second table, you will need to extract and reformat data from the Type 81 records. Here are examples for Natural Gas.

For the first part of the second table:

81NYMNG        NG        FUT 201810            000000000000+00000+00367-00367-00367+00367+00733-00733-00733+00000000280500N

The futures code is "NG        " **# commodity/product code**

the contract month is "201810" **# fut contract month**

the contract type is "FUT"

and the settlement price is "00000000280500", which for natural gas you need to divide by 100000.0 to get 2.805 (natural gas prices are displayed to tenths of cents, unlike crude oil futures prices which are displayed to cents). For WTI crude, you will need to figure out the correct divisor, by comparing the contents of the settlement price field with actual current WTI crude futures contract prices (Google is your friend, here).

For the second part of the second table: 

81NYMON        NG        OOFC201810   201810   000275000087-00100+00314-00133-00123+00311+00558-00386-00314+00000000000820N

81NYMON        NG        OOFP201810   201810   000275000087-00100+00053+00233+00244-00055-00175+00347+00420-00000000000420N

The first record is for a call option (Option Right Code "C"), which is an option to buy a futures contract; the second record is for a put option (Option Right Code "P"), which is an option to sell a futures contract.

The Underlying Commodity (Product) Code (futures code) is "NG        "

The Product Type Code is "OOF" (option on futures)

The Option Strike Price is "0002750" which you need to divide by 1000.0 to get 2.750,

and the settlement price is "00000000000820" for the Call and

"00000000000420" for the Put.

Dividing by 10000.0, you get 0.082 as the price of a Call option, and 0.042 as the price of a Put option. For WTI crude, you will only need to figure out one divisor, not two.

A few more hints:

(a) Notice that the documentation counts character column positions from 1, whereas in your code you will need to count character positions from 0 for **str** slices.

(b) Check the contract specifications to discover the number of decimal places you should display for price (U.S. dollars) of WTI Crude Oil futures and options contracts.

(c) Approach the program in stages:

First, make sure you can write a program that simply copies the SPAN file to the output file.

Next, modify your program to copy the type B and type 8 records from the SPAN file to the output file.

Next, modify your program again to copy the type B CL and type 8 CL records

And so forth, making definite steady progress with each revision.

As your coding skills improve, you can do two or three or four things in each revision step. Eventually, you will find that you can write dozens of lines of code encompassing many different tasks and goals, and it will work the first time! (Or maybe not.)

(d) There are subtypes of the type 8 records: it turns out you can just use the type 8 subtype 1 (or just type 81) records, and ignore the type 82 records. (For WTI Crude Oil, there are no type 83 records.)

(e) There is a brief description of string formatting in McKinney’s book, under 2.3 Python Laguage Basics => Scalar Types => Strings. More examples can be found here: https://docs.python.org/3.8/tutorial/inputoutput.html

(f) Remember that collaboration is encouraged: in addition to your homework partner(s), feel free to compare what you are doing with other students, as well. Just make sure you submit your own homework team’s code, after whatever discussions you have with others.

(g) Please feel free to email the TAs or myself with any questions you may have.

***When finished, put your hw2.1.py source code file into a zip archive named* Team***N***\_HW2.zip *file, where*** *N* ***is your team number, and upload your .zip file to Canvas.***