Philippine Stock Market Data: (some simple) Analysis

> # You are NOT allowed to use functions which were not discussed in class (unless specified otherwise).

> Refer to the accompanying file **LASTNAME_C4to5.c** for the skeleton code that you'll need to complete.

> Familiarize yourself with accompanying folder named **SHD** which contains 30 text files which in turn contains the stock historical data for the 30 companies. **You should use the SHD text files in the subsequent challenges.**

Challenge #4: Load ALL the 30 Text File SHD Values into the Primary Memory.
Implement the following function which should:
1. read the SHD from ALL the 30 text files and
2. load/store them into your data structure (DS) which is in the primary memory.

```
void
Initialize_SHD_Data_Structure( _datatype_ DS );
```

where **DS** is the parameter that corresponds to your data structure, and **_datatype_** is the **typedef** name you declared in Challenge #3 above. After executing this function, the stock name, the number of daily transactions and the values of the historical data for ALL the 30 PSEI stocks should be accessible via the DS parameter.

Notes:
- You will need to use **fopen()**, **fclose()**, **fscanf()** to solve this challenge.
- DO NOT call **printf()** in your solution.

NOTE: The function **Initialize_SHD_Data_Structure()** should be called first before executing the functions in all the remaining challenges.

Challenge #5: Display the Daily Data for a Given Stock Code, Start Date and End Date
Implement the following function which should **printf()** the daily data for the company with the specified stock code, start and end dates. Assume that the function **Initialize_SHD_Data_Structure()** was called and executed properly without any error.

```
void
Print_SHD( _datatype_ DS, const char *stock_code,
                 const char *start_date, const char *end_date );
```

where **DS** is the parameter that corresponds to your data structure, and **_datatype_** is the **typedef** name you declared in Challenge #3 above.

Some example function calls are shown below:

```
Print_SHD(DS, "AC",  "01/01/2010", "12/31/2014"); // Example #1: multiple years
Print_SHD(DS, "MBT", "01/01/2015", "12/31/2015"); // Example #2: one year
Print_SHD(DS, "SM",  "07/01/2016", "07/31/2016"); // Example #3: one month
Print_SHD(DS, "DMC", "03/10/2017", "03/16/2017"); // Example #4: one week
Print_SHD(DS, "GLO", "08/08/2018", "08/08/2018"); // Example #5: one day (with SHD)
Print_SHD(DS, "TEL", "12/30/2018", "12/30/2018"); // Example #6: one day (no SHD)
```

Make sure that your output complies with following output format. The first line of output should indicate the stock code. The succeeding lines of output should indicate the values for the dates, OHLC and volume. Floating point values should be printed with two digits after the decimal point. All data should be printed starting from older dates to more recent dates. You may specify any number of spaces to separate values printed on the same line.

Notes:
- Assume that all parameters are correct; thus, there is no need for you to validate them.
- Assume that the **start_date** is less than or equal to the **end_date**.
- The **start_date**, **end_date** and dates in between may or may not have matching dates in the SHD currently loaded in your data structure. For example, there are no data for years 2010 to 2013 because we gathered and loaded data only from 2014 to 2018. Another example, there are no data for dates that fall on Saturdays, Sundays and holidays because the stock market is closed! Make sure that your program prints only the stock name and NOTHING ELSE for dates that do not have a match in the SHD.

Some example output are shown below for your reference.

The expected output corresponding to the function call:

```
Print_SHD(DS, "DMC", "03/10/2017", "03/16/2017"); // Example #4: one week
```

is as follows:

```
DMC
03/10/2017        12.38        12.54        12.34        12.46         6672100.00
03/13/2017        12.46        12.58        12.44        12.52         1657500.00
03/14/2017        12.50        12.52        12.28        12.36         9919300.00
03/15/2017        12.36        12.40        12.18        12.26         5156500.00
03/16/2017        12.28        12.28        12.00        12.04        13285000.00
```

Note that the output is sequenced chronologically from the oldest date to more recent dates. Note also that there are no data for 03/11/2017 and 03/12/2017 because they fall on Saturday and Sunday respectively.

The expected output corresponding to the function call:

```
Print_SHD(DS, "GLO", "08/08/2018", "08/08/2018"); // Example #5: one day (with SHD)
```

is as follows:

```
GLO
08/08/2018       1883.00       2030.00       1883.00       2024.00        306500.00
```

The expected output corresponding to the function call:

```
Print_SHD(DS, "TEL", "12/30/2018", "12/30/2018"); // Example #6: one day (no SHD)
```

is as follows:

```
TEL
```

Note that 12/30/2018 is a holiday, and therefore there is no transaction. Thus, there is no output other than the stock name.

DELIVERABLES: submit FOUR files described below via Canvas. Take note of the Canvas submission deadline.

1. Your completed C program based on the given skeleton file **LASTNAME_C4to5.c**.

2. Your header file **LASTNAME_DS.h.**

3. Your own test case files containing the test parameters that you used in testing your solution. Submit TWO test case files with filenames and contents described below:

    a. **LASTNAME_TEST1.TXT:** The first test case file should contain parameters for testing ALL of the following:
        i.   multiple years
        ii.   one year of the same year (example: from 1/1/2017 to 12/31/2017)
        iii. one year with a starting year different from the ending year (example: 7/1/2017 to 6/30/2018)
        iv.   one month
        v.   one week
        vi.   one day with a transaction, and
        vii. one day without a transaction.

    b. **LASTNAME_TEST2.TXT**: The second test case file should contain parameters that will print ALL the stock historical data values (i.e., from 1/1/2014 to 12/30/2018) for ALL companies (i.e., from AC to URC) in alphabetical order.

TESTING: Always make sure to test your programs thoroughly. Your program should not have any syntax error, warning and logical error. The testing technique that you should apply have already been described and discussed in our class.

In case you already forgot about it, please click this link and make sure to learn and practice first the recommended testing technique.

To be continued… (Part 3 will be sent in another email).