**Winter 2024 Assignment 03 - Methods, Arrays, and File I/O**

**Weight:** 15% of final mark

**Submission requirements:** On or before the deadline, commit a Visual Studio 2022 project to the GitHub repository. **~~You must commit and push to the classroom repository supplied for the assignment~~**; do not create your own repository. It is your responsibility to ensure that your work is in the correct repository. ***Work not in the repository will not be graded***.

**CONTEXT**

A friend of yours sells handmade goods via tradeshows and pop-ups around town. They need help building a sound financial picture of how their small business performs every month. They shared with you that it would be nice to have a program that could **produce some basic sales stats for them to review**, such as the **average daily sales**, **highest sales day**, **lowest sales day**, etc. After hearing about their plight, you offer your expertise to help them by building a program to help them out. It should be simple enough ...

**Requirements**

Your program must meet the following requirements:

* Must allow the user to **enter** daily sales amounts (your friend doesn't set up shop every day)
* Must allow the user to **save** their entered daily sales to a file
* Must allow the user to **load** a previously saved daily sales file
* Must allow the user to **view** and **edit** previously entered sales values
* Must allow the user to **view simple analysis** of the currently entered/loaded data:
  + Median average sales
  + ~~Median of sales~~
  + Highest daily sales amount
  + Lowest daily sales amount
  + Chart of daily sales for the current month

**Implementation Details**

Your job will be to complete the missing requirements where indicated. There are a number of tasks that are all identified by // TODO:  comments throughout the Program.cs file.

1. The program makes use of **a main menu for top-level options** and **a sub-menu for the analysis options**. The program should continue to run until the user chooses to quit the program. **Ask the user to supply the desired filename when saving a new monthly sales file**. The user will need to enter new sales values for one month only and provide a value for each day in the month (days with no sales will be recorded as zero). **Ensure that date values are in valid MM-dd-yyyy (e.g., 02-21-2024) format and that sales values are zero or positive.**
2. Use two **parallel arrays** (dates & data) for storing the data in your program (one for date values and one for corresponding daily sales values). Keep an accurate record count for the number of days of data that have been loaded/entered.
3. Ensure that **duplicate entry dates** (when entering data) **are not allowed**; there should only be **one sales value per date**.
4. The format of the sales data files should be as follows (assume valid file format for input):

* **Include a header record** with the following headings: **Date and Sales**.
* Sales results are recorded to **two decimal places**.
* Data files must include the date in MM-dd-yyyy format and be ordered in **ascending date order**:

Date,Sales

02-01-2024,546.50

02-02-2024,0.00

02-02-2024,416.75

02-03-2024,674.25

...

02-29-2024,339.25

*Sample data file format*

1. You will use a modular approach when constructing this program. Ensure that, at a minimum, the following methods are present and used (difficulty level is rated 1-easy, 2-moderate, 3-challenging, 4-extreme):

* void DisplayMainMenu() --> displays the main menu options [difficulty 1]
* void DisplayAnalysisMenu() --> displays the analysis menu options [difficulty 1]
* string Prompt(string promptString) --> displays the prompt string and returns user-entered string (allow empty string to be returned (use “”)) [difficulty 1]
* double PromptDouble(string promptString) --> displays the prompt string and returns user-entered double (ensure that the program does not crash and always returns a valid double value) [difficulty 1]
* int HighestSales(double[] sales, int countOfEntries) --> returns the index of the *highest* sales amount in the sales array (requires that the original ordering of the arrays be retained) [difficulty 1]
* int LowestSales(double[] sales, int countOfEntries) --> returns the index of the *lowest* sales amount in the sales array (requires that the original ordering of the arrays be retained) [difficulty 1]
* double MeanAverageSales(double[] sales, int countOfEntries) --> returns the mean average of the *daily* sales; include all days for the month (yes, even days with zero sales) [difficulty 1]
* int EnterSales(double[] sales, string[] dates) --> allows the user to enter dailys sales entries (dates and sales values) into the arrays; returns the number of entries entered [difficulty 2]
* int LoadSalesFile(string filename, double[] sales, string[] dates) --> loads the records from a file (filename) into the associative arrays used by the program; returns the record count (i.e. how many days of data were loaded) [difficulty 2]
* void SaveSalesFile(string filename, double[] sales, string[] dates, int countOfEntries) --> writes the associative array data to a file (filename) in the correct format [difficulty 2]
* void DisplayEntries(double[] sales, string[] dates, int countOfEntries) --> displays the current entered/loaded sales entries in a formatted table (i.e. ensure that proper columns and alignment are used). **You must use a for loop to loop through the arrays and produce the display** [difficulty 2]
* void EditEntries(double[] sales, string[] dates, int countOfEntries) --> allows the user to view all current entries and choose one to edit (i.e. overwrite) [difficulty 3]
* void DisplaySalesChart(double[] sales, string[] dates, int countOfEntries) --> displays a chart of the pH log data in the following format: [difficulty 4]

=== Sales for the month of February ===

Dollars

700|

650| $675

600|

550|

500| $546

450|

400| $416

350|

300| $339

250|

200|

150|

100|

50|

0| $0

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

Days | 01 02 03 04 ... 29

Use the maximum value in the sales array to determine the y-axis markers (count by 50s for the y-axis values) and the dates for the x-axis. In the chart above, the ellipsis (...) is used for demonstration only, include all days in your implementation. [difficulty 4]

The program should never crash and must deal with errors gracefully.

**Aside from what’s been presented in this document, do not make any assumptions. Seek clarity from your instructor if you do not understand something in this document.**

**Coding Requirements**

* A C# comment block at the beginning of the source file describing the purpose, author, and last modified date of the program
* Write only one statement per line
* You must use two corresponding/parallel arrays for sales and dates in your solution
* You must not use built-in methods for finding the average, high, and low values in arrays
* Use camelCase for local variable names
* Use TitleCase for any constant variable names
* Use defensive programming where necessary
* Ensure graceful handling of exceptions (\*try/catch)

**Sample Runs**

**Sample Program Run**

// TODO