AIM 5001 Module 11 Assignment (100 Points)

Part 1: Tidying and Reshaping Data

	Month	Category	Caltex	Gulf	Mobil
2	Open	Engine Oil	140 : 000	199 : 000	141 : 000
3		GearBox Oil	198 : 000	132 : 000	121 : 000
4	Jan	Engine Oil	170 : 103	194 : 132	109 : 127
5		GearBox Oil	132 : 106	125 : 105	191 : 100
6	Feb	Engine Oil	112 : 133	138 : 113	171 : 101
7		GearBox Oil	193 : 148	199 : 119	134 : 127
8	Mar	Engine Oil	184 : 100	141 : 141	114 : 108
9		GearBox Oil	138 : 121	172 : 133	193 : 115
10	Apr	Engine Oil	149 : 150	117 : 118	117 : 118
11		GearBox Oil	185 : 125	191 : 133	119 : 121
12	May	Engine Oil	170 : 139	104 : 119	200 : 117
13		GearBox Oil	168 : 117	138 : 102	121 : 146
14	Jun	Engine Oil	159 : 129	170 : 138	169 : 105
15		GearBox Oil	107 : 129	195 : 141	141 : 112

The chart above describes purchases and use of marine oil available at a major seaport. There are two types of oil available for use at the seaport: **engine oil** and **gearbox oil**. Each type of oil is provided by three distinct oil manufacturers/suppliers: **Caltex, Gulf**, and **Mobil**. The contents of the 'Caltex', 'Gulf', and 'Mobil' columns contain the number of gallons of oil purchased and consumed (e.g., **purchased : consumed**) for each month, with the '**Open**' indicator shown at the top of the chart telling us how much of each type of oil was on hand at the start of the chronological period (i.e., the 'purchased' amounts are the starting inventories for each type/brand of oil). The content of the chart has been re-created within the provided **M11_Data.csv** file. Get started as follows:

- Upload the provided **M11_Data.csv** file to your online AIM 5001 GitHub repository.
- Using the **pd.read_csv()** function, read the **M11_Data.csv** file from your GitHub repository into a Jupyter Notebook WITHOUT removing any empty rows or columns from the content of the file. The content of the resulting dataframe should appear as follows:

	Month	Category	Caltex	Gulf	Mobil
0	Open	Engine Oil	140 : 000	199 : 000	141 : 000
1	NaN	GearBox Oil	198 : 000	132 : 000	121 : 000
2	Jan	Engine Oil	170 : 103	194 : 132	109 : 127
3	NaN	GearBox Oil	132 : 106	125 : 105	191 : 100
4	Feb	Engine Oil	112 : 133	138 : 113	171 : 101
5	NaN	GearBox Oil	193 : 148	199 : 119	134 : 127
6	Mar	Engine Oil	184 : 100	141 : 141	114 : 108
7	NaN	GearBox Oil	138 : 121	172 : 133	193 : 115
8	Apr	Engine Oil	149 : 150	117 : 118	117 : 118
9	NaN	GearBox Oil	185 : 125	191 : 133	119 : 121
10	May	Engine Oil	170 : 139	104 : 119	200 : 117
11	NaN	GearBox Oil	168 : 117	138 : 102	121 : 146
12	Jun	Engine Oil	159 : 129	170 : 138	169 : 105
13	NaN	GearBox Oil	107 : 129	195 : 141	141 : 112

1.1 (**30 Points**): Use your knowledge of combining and reshaping data in Pandas to tidy and transform/reshape the data contained within the dataframe. To get started, think about how you would want the data to appear if it were converted to "long" format, e.g., how would you define a "single observation" for the data shown in the graphic?; How many key values are associated with each data value?; How many columns should your long format structure contain based on the information provided in the graphic shown above?; What would the column headings for the long structure be?; etc. Use your answers to these questions to guide your reshaping/transformational work on the data. **Your reshaping/transformational steps must include converting the above table to a "tidy" long format.** Additional transformational steps (e.g., filling in missing data values, renaming columns, etc.) should be performed as needed to ensure that your data is, in fact, "tidy".

1.2 (15 Points) Using your reshaped/transformed data, perform analysis to answer the following questions:

- What was the amount of oil remaining for each type/brand **at the end of the chronological period**?
- What was the most consumed brand of oil across the two separate categories/types of oil?

1.3 (15 Points) Finally, given your "tidy" long format structure, describe what, if any, changes you would make to the visual presentation of the data if you were then asked to transform your "long" data back into a "wide" format: would you mimic the structure of the graphic shown above? If not, how might you transform your "long" data to "wide" format to make its "wide" presentation easier to understand and work with? Provide an example of your recommendation and explain your rationale for preferring your specific structure.

Part 2: Using Your GroupBy and Data Aggregation Skills

Three Short Coding Challenges
Can you complete these three tasks using no more than 17 lines of code in total?

These coding challenges will give you a chance to exercise your **GroupBy/Aggregation/Split-Apply-Combine** skills based on your readings from Chapter 10 of the "**Python for Data Analytics**" textbook. See if you can answer these three questions using **no more than *17 total lines of Python code***.

For each of the three questions you will be making use of the Pittsburgh Bridges data set: https://archive.ics.uci.edu/ml/datasets/Pittsburgh+Bridges. (Links to an external site.) Upload the provided **briges.data.version1.csv** to your online AIM 5001 GitHub repository and then read the file from GitHub into your local Python environment

2.1 (**12 Points**): You've been asked to generate a quick report that tells us how many bridges of each '**Purpose'/'Material**' grouping within the data set have been constructed over each of the rivers listed in the data set. **For each river**, your output should include the Purpose, Material, and count (aka 'How Many?'), similar to the output shown in the graphic below for River 'A', and **your report should include similar content for each of the rivers** contained within the data set.

River	Purpose	Material	How Many?
A	AQUEDUCT	IRON	1
		WOOD	3
	HIGHWAY	?	1
		IRON	2
		STEEL	21
		WOOD	8
	RR	IRON	1
		STEEL	9
		WOOD	2
	WALK	STEEL	1

You are allowed to use **no more than three (3) lines** of Python/Pandas code to generate this report in its entirety (i.e., you **MUST** produce the results for all of the rivers at once) and you **MUST** use Pandas' groupby and/or aggregation functionality to accomplish the task. **Be sure to include a brief narrative explaining how your proposed code would accomplish the task.**

2.2 (14 Points): You've been asked to generate a second report that shows the average length for each 'Purpose'/'Material' bridge grouping within the data set. As you should recall from our previous work with the Pittsburgh Bridges data set, the 'Length' attribute is not provided to us in a numeric format and also contains many missing values. As such, you should clean up the contents of that column and convert it to numeric format before attempting to generate your report. The output of your report should appear as shown in the graphic below.

Purpose	Material	Average Length
AQUEDUCT	IRON	1000.000000
	WOOD	1092.000000
HIGHWAY	?	NaN
	IRON	1216.666667
	STEEL	1557.804348
	WOOD	1053.375000
RR	IRON	1100.000000
	STEEL	1946.850000
	WOOD	NaN
WALK	STEEL	NaN

You are allowed to use **no more than four (4) lines** of Python/Pandas code ***AFTER*** you've finished cleaning up the 'Length' column (which should take no more than 2-3 lines of code) and you **MUST** use Pandas' groupby and/or aggregation functionality to accomplish the task. **Be sure to include a brief narrative explaining how your proposed code would accomplish the task.**

2.3 (14 Points) Finally, you've been asked to generate one last report that shows the average length, count, minimum length, and maximum length of bridges built during 4 equal length time periods (1818 – 1860; 1860-1902; 1902-1944; 1944-1986). The output of your report should appear as shown in the graphic below.

Erected	Average Length	Count	Max Length	Min Length
(1818.0, 1860.0]	1094.625000	8.0	1500.0	990.0
(1860.0, 1902.0]	1603.347826	23.0	4558.0	1000.0
(1902.0, 1944.0]	1676.181818	33.0	3000.0	860.0
(1944.0, 1986.0]	1530.411765	17.0	3756.0	804.0

You are allowed to use **no more than seven (7) lines** of Python/Pandas code and you **must** use Pandas' groupby and/or aggregation functionality to accomplish the task. **Be sure to include a brief narrative explaining how your proposed code would accomplish the task.**

Save all of your work for this assignment within **a single Jupyter Notebook** and upload / submit it within the provided M11 Assignment Canvas submission portal. Be sure to save your Notebook using the following nomenclature: **first initial_last name_M11_assn**" (e.g., J_Smith_M11_assn).