Overview:

- This project is designed around the Beverage Company case study provided with our textbook. You can access the case study and other resources for the text book by going to my website and clicking the desired link.
  - The Business Case Study link provides an overview of the context, links to the data sets, and includes some descriptions of some of the output objects as well as the output objects themselves.
  - If any instructions below conflict with instructions in the case study, follow the directions below!
  - The case study was written to allow readers to complete portions as they finished each chapter. For example, all activities numbered 3.x can be completed with only the material up through Chapter 3. However, since this is your final, you are not restricted in how you can produce an output object. Just like in a real project, you are free to use - and in fact, encouraged to use - all the techniques you've learned during the semester!
- Remember, this project is divided into two phases: data handling and result generating.
  - Data Handling: The first section of the project and this code is due at the first due date. The purpose of this phase of the project is to build all the data sets you need for phase two. *No output is due at the end of phase 1.*
  - Result Generating: The second section of the project this code is due at the second due date. The purpose of this phase of the project is to create all the analyses - both tables and graphs. *To ensure that everyone can earn full credit on Phase 2, you must use **my** data sets from Phase 1 when you work on Phase 2.*
- Data Sets:
  - As stated above, all data (raw, SAS, Excel, Access, etc.) for this case study have already been placed on the shared drive in `Data/BookData/BeverageCompanyCaseStudy` – **this means you do not need to download any of the data!**
  - All data sets from raw files (.txt., .dat, .csv, etc.) contain headers that describe the data included in the file. The list below is all data files you will need. If I have extra information beyond what is in the file itself, I have included it below.
    - 2016Counties.accdb is a Microsoft Access database.
      - You must use this file to retrieve information regarding the Counties.
      - Store the County table from this file into a SAS data set named Counties.
      - Clear the library for this file as soon as you read the necessary information from it.
      - Note: Normally we would not bother making a copy – we would just read from it directly – but since there are many of you trying to access it at once, it makes more sense for you to each create your own local copy to work with.
    - Non-Cola–NC,SC,GA.dat
      - This file contains information about beverages which are from the South region (NC, SC, GA) and which are not colas, energy drinks, or other specialty drinks.
      - Use formatted input at most once when reading this file and do not use any list-based input styles.
      - You may use at most one column pointer control to read this file.
      - Name the resulting data set NonColaSouth
    - Energy–NC,SC,GA.txt
      - This file is tab delimited. Only use list-based input styles.
      - Name the resulting data set EnergySouth
    - Other–NC,SC,GA.csv
      - This file is comma delimited. Only use list-based input styles.
      - Name the resulting data set OtherSouth
    - ColaDCMDVA is a SAS data set
    - ColaNCSCGA is a SAS data set
    - Non-Cola–DC-MD-VA.dat
      - This contains the non-cola, non-energy, non-specialty drinks for the North region (DC, MD, VA)
      - This is a positional file but **use only formatted input** to read the data.
      - Name the resulting data set NonColaNorth

    - Energy–DC-MD-VA.txt
        - Another tab-delimited file. Only use list-based input styles.
        - Name the resulting data set EnergyNorth
    - Other–DC-MD-VA.csv
        - Another comma-delimited file. Only use list-based input styles.
        - Name the resulting data set OtherNorth
    - Sodas.csv
        - Comma-delimited file with repeated (and nested) values within each row
        - This contains information on the sodas that allows you to build product codes.
        - Store this in a data set named Sodas
        - This data should **never** be combined with any other data set.
        - This data set is only for producing Output 7.1
        - HINT: Do not try to read this data set using brute force - start by determining what variables you need and what should go in them. Then determine how to chop up the data you were given to produce the values you want!
        - For example, the first record shows that flavor (Cola) comes in 9 different products: a 12 oz can, a 20 oz bottle, a 1 liter bottle, a 2 liter bottle a 6-pack of 12 oz cans, a 12-pack of 12 oz cans, a 24-pack of 12 oz cans, an 8-pack of 20 oz bottles, and a 12-pack of 20 oz bottles.
        - HINT: You may find that you want to use some additional functions to help you out with this - specifically, `INDEX` and `CATX`.
  - *No other data sets are needed to start the project. Any other data sets you need are to be created by you from the data sets described above.*

- Results:
  - You are not being asked to reproduce the entire set of output objects. Instead, I've selected (not at random) a subset of the output objects for you to produce. Because not all activities have a number in their title, I have included the table of contents number as well as the activity number title, if appropriate. For example, #3 (Activity 2.1) refers to Item #3 in the Table of Contents which is Activity 2.1.
    - Chapter 2: #3 (Activity 2.1), #5 (Activity 2.3)*
    - Chapter 3: #12 (Activity 3.1)*, #15 (Activity 3.3)*, #19 (Activity 3.6)
    - Chapter 4: #22 (Activity 4.1), #23 (Activity 4.2), #25 (Activity 4.4)
    - Chapter 5: #28 (No activity number), #33 (Activity 5.5)
    - Chapter 6: #36 (Activity 6.2)
    - Chapter 7: #44 (Activity 7.1)*, #40 (Activity 7.4), #41 (Activity 7.5)
  - The provided results are HTML, which we did not learn this semester, so you'll be delivering your results to a PDF. We'll use our usual naming conventions where Tony Stark would end up with a file named StarkFinalReport.pdf.

    Note: If you are interested in reproducing the HTML output, I put a note all the way at the bottom of this file to show you how to do that. This is not an expectation, so I don't want anyone feeling obligated to get the HTML to work!
  - Unless otherwise specified, your output should be identical to those in the book. Color choices do not need to match exactly.
  - While you should be able to look at the outputs in the results and determine most of what needs to be done, I'm including the following list of details to either provide additional clarity on producing an output object or change the way an output object is created. I've also included any instructions that you cannot determine from the output objects themselves. These items are in no particular order, but are numbered to make it easier for us to communicate with each other about the specifics.

Data Handling:

1. Budget: 9 DATA steps and 8 PROC steps
2. Note, these data sets may take a while to build. Once you have them working, you don't want to rerun your code – so even save your intermediate data sets to your permanent library to help save some time for future you! Essentially, you should never be saving anything to the Work library for this assignment unless you are building some data sets for self-validation purposes.
3. After you read the non-SAS data sets into permanent SAS data sets, combine the data sets as follows:
   - Ensure the names are standardized. I gave you my PROC CONTENTS results so you could plan ahead!
   - Produce a data set named AllDrinks that contains all records from Cola, Non-Cola, Energy, and Other for both the North and South regions. Choose an appropriate method for joining the data sets – there are only two reasonable choices and your budget may dictate which one of those you should use.
     - **This is where all your data cleaning and derivations of new variables should go for these data sets. *Do not do any data cleaning when reading in the individual files!***
   - Combine the AllDrinks data set with the Counties data set to produce a data set named AllData. Again, use the appropriate technique to combine these data sets.
     - There is a new variable to be created here because it depends on information from the Counties data set.
   - The AllDrinks data set will be what you use as a basis to produce all requested output objects through Chapter 6 and most of the objects from Chapter 7.
4. As you know, we all need to be prepared to handle a variety of data cleaning issues as well as deriving new variables. As you are producing the data sets above, you will need to carry out a variety of cleaning tasks and derive a number of new variables. **Absolutely no cleaning of variable values or deriving new variables should take place when reading in the raw files. Instead all cleaning and deriving should be done when creating the AllDrinks data set except for the creation of SalesPerThousand (see below).**
   - Deriving new variables
     - **SalesPerThousand** is the ratio of UnitsSold to the average population from 2016 and 2017. Since it is per thousand, be sure to scale by 1000 to get the units correct.
     - **Region** is either South or North based on which file provided the data.
     - **Date** is provided uniformly in the raw files for the South but in the raw files for the North you must deal with the fact that dates are displayed differently. Here are some hints to simplify this process:
       - For Non-Cola and Energy, you need to decide when to use `MMDDYY` and when to use `DATE`. That can be done using only functions we have learned this semester, but it might be easier to use the `INDEX` function. If you want to go that route, use the SAS documentation to see some examples of the function. I've also included some examples below.
       - For the Other data set, many different styles of dates were used; however, they can all be interpreted using an informat called `ANYDTDTE`. While this approach cannot always work, it will be quite helpful to read the dates using that informat in this case.
     - **ProductName**, **Type**, **ProductCategory**, **ProductSubCategory**, **Flavor**, **Size**, and **Container** are all derived based on the Product Information and Categorization Table given in the Case Study. Take a look at entry #28 in the Table of Contents. This table shows all valid values for all of these variables. It is possible to derive all the variables from this table (that's how I had to do it!) but I've provided additional info below in case it is helpful.
     - **ProductName** is provided directly in some data sets and in other data sets you must determine the value of productName by using product codes.
       - Do not use conditional logic when deriving the product names. I have provided several formats to help determine ProductName. You'll have to write any other formats you need for this.
       - Product name codes can be found in the Appendix of the Case Study. Output 7.1 has the product codes for Sodas, 7.2 has the codes for Energy Drinks, and 7.3 has the codes for Other drinks.
     - **Type** is either Diet or Non-Diet based on the name of the product
     - **ProductCategory** is partially based on the categorization of the product (Cola, Non-Cola, Energy, Other) but includes more information as you can see from the tabl.e
     - **ProductSubCategory** only applies to energy drinks and is derived from the ProductName
     - **Flavor** is derived from the name of the product.
     - **Size** is always provided, but does require some data cleaning as described below.
     - **Container** is derived from Size.

- Data cleaning
  - **Size** comes in various units such as 12 oz or 2 liter. Values such as 12 OZ, 1 LITER, etc. are present and need to be addressed. The only acceptable units are "oz" and "liter"
  - **ProductName** should always appear with proper casing.
- I have provided several new formats that you may find helpful. You should be able to explore our usual formats catalog and determine which ones are helpful here.
- Because you may want to use functions we have not used before (`INDEX` and `CATX`) I wanted to provide a few examples.
  - The `INDEX` function finds the first instance of a string within a larger string.
    - `INDEX(''Jonathan'',''n'')` would equal 3 since the first "n" is in the third column.
    - `INDEX(''Jonathan'', ''N'')` would equal 0 since there is no capital N in my name – remember, string matching is case-sensitive!
    - `INDEX(''Jonathan'', ''Jon'')` would equal 1 since the string "Jon" is found starting in the first column of the string "Jonathan"
  - The `CATX` function allows you to concatenate strings with a delimiter in between each concatenation.
    - `CATX('-','123','45','6789')` would produce '123-45-6789'
    - `CATX(' na ', 'NA', 'NA', 'NA', 'Batman')` would produce 'NA na NA na NA na NA na Batman' which you are now obligated to sing in your head for the rest of the day.

5. To produce Activity 3.6, you'll need to create the appropriate data set from your **AllData** file.
6. To produce Activity 4.4, you'll need to create the appropriate data set from your **AllData** file.
7. To produce the Optional Activity, you'll need to create the appropriate data set from your **AllDrinks** file.
8. To produce Activity 5.5, you'll need to create the appropriate data set from your **AllData** file.

Result Generating:

Remember, you are not being asked to complete the entire case study.

- Chapter 2: #3 (Activity 2.1), #5 (Activity 2.3)*
- Chapter 3: #12 (Activity 3.1)*, #15 (Activity 3.3)*, #19 (Activity 3.6)
- Chapter 4: #22 (Activity 4.1), #23 (Activity 4.2), #25 (Activity 4.4)
- Chapter 5: #28 (No activity number), #33 (Activity 5.5)
- Chapter 6: #36 (Activity 6.2)
- Chapter 7: #44 (Activity 7.1)*, #40 (Activity 7.4), #41 (Activity 7.5)

1. Budget: 1 DATA step and 14 PROC steps
2. For Activity 2.3, only include Table 1 and Table 2 regardless of the values of ProductName. (I.e., it's not that we want Cherry Cola and Cola – it's that we want whatever the first two tables are.)
3. For Activity 3.1, restructure the legend to be a 3x2 set of names that lists Citrus, Grape, and Lemon-Lime in the first column then Orange and Zesty in the second column.
4. For Activity 3.3, it can be hard to tell which dataskin is applied, but the bars have the `SHEEN` dataskin. Also, your legend will show up with the label "Containers per Unit" instead of the variable name – that is OK.
5. For Activity 3.6, the narrower bar is 60% of the wider bar and the front bar has 40% transparency.
6. For Activity 4.4, use the `INTERVAL = MONTH` option to force SAS to only place tick marks for new months. (SAS will do this by default in this case, but doing it explicitly is (a) always better and (b) avoids a lot of notes in the log.) Also, use the format `MONYY` for the same reason – it avoids notes in the log and you can pick the width needed to get the values like `JAN2016` that you want. (Also, don't worry, SAS knows how to split up the months and years and to only print the year when it changes – that's all built in!)
7. For the Product Information and Categorization table, the values are sorted by: Category, Subcategory, Name, Type, Container, Flavor, Size.
8. For Activity 5.5, use the `TYPE=LINEAR` option when constructing the horizontal axis. It should be the default, but we are grouping based on dates which confuses SAS so it tries to use a DATE axis even though we are not actually plotting dates on the horizontal axis. (This prevents several notes in your log that tell you SAS had to fix this.)
9. For Activity 6.2, compute the required statistics in PROC REPORT using aliasing. For the dates, you must use the `QTRR` format provided by SAS to get the quarters with Roman numerals.
10. The one DATA step is to read in the Sodas.csv file which is used to produce Output 7.1. This is included in Phase 2 because it requires skills from Lecture 25 as well as other techniques from the semester. If I were you, I would focus on everything else first, then worry about the data/output for 7.1.
11. Note that for Output 7.4 the rows are banded with different colors. In 7.4 use white for the first row, then progressively darker shades of gray for the next three rows. Use black for the summary row. Ensure that if there were not four rows per group, that the colors would still cycle every 4 rows. I.e., the first row should always be white within a group, but so should the 5th row, 9th row, etc. The 2nd, 6th, 10th, etc. rows should always be the same color and so on. [This allows us to keep the color banding working even if we summarized biennially instead of annually, for example.] The header row should be a unique shade of gray with blue font. (The blue doesn't have to match the one we used because I think it is ugly.)
12. For Output 7.5, the header row is still a special shade of gray, but now rows are shaded gray only if the row is flagged as per the footnote in which case the Sales per 1000 column is flagged as well using red text. County-level summaries should have their own shade of gray, and population counts should be in a black row.
13. Unless otherwise specified, your output should be effectively identical to those in the case study. Color choices do not need to match exactly unless you've been told what colors we used. If you use my labels, your output may differ slightly since my labels don't always agree with those in the case study in the book.
14. All images use 300 dpi for resolution and are 6 inches wide. For all bar graphs, select your own color palette for the fill colors and ensure the line colors match.
15. For all output objects, include titles that match the captions from the case study. If the output object has a footnote, include that as well!
16. Use the ODS option NOPROCTITLE to ensure that only the headers we want are appearing in the output.