More R Markdown

Open RStudio and create a new project under your Module 2 folder and call it **Assignment3.** For this assignment, you will be creating an R Markdown document. **Once completed, all you need to do is submit the word document that is created**.

Part 1: Creating the R Markdown Document

1.) In the previous assignments, you created a markdown document by starting with a text file. For this exercise, I would like you to create an R Markdown document directly from the menu.

2.) In RStudio, select *File -> New File -> R Markdown...* You now have the option to choose your title, author and output type. Use the following:

Title:	**More R Markdown**
Author**:**	*Your Name*
Default Format:	Word

3.) This will create a Markdown file that has pre-built code within the file itself. Save this file to your project as **Mod2Assign3Answer.rmd**. Later, you will notice when you click on the Knit icon that it will automatically create a word document.

4.) Notice that there will be information at the top of the document concerning title, author, etc. This will be the title for your Word output file when you knit the markdown file.

5.) It is also a good habit to add comments to your code to know what is happening in the markdown document you are creating. However, you may not want this to appear in the output. To do this, you can put all comments between <!-- and -->. Write the following comments at the top of your RMarkdown document:

<!--
Date Created: *Use today's date in YYYY-MM-DD format*
Description: *Add a few sentences that includes what the program is doing, files you plan to use, where you got them, etc. For this project, add the following text under description:*

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. The following is just a sample of some of the things you can do with R Markdown.

Modified:
This will not have anything now but will be useful if you make changes in the future to a document. For this project, put today's date (in YYYY-MM-DD format) and type created
-->

6.) Click on the Knit icon at the top of the R Markdown Pane in RStudio. Since we have the default format set to Word, you do not need to actually select what type of output from the dropdown.

7.) Notice that you now have a document in your files for the project named **Mod2Assign3Answer.docx**. This is the file you will be uploading later to Canvas.

Part 2 – Adding code and commentary

1.) Because we are using the R Markdown file that comes with pre-existing text/code, keep the first chunk of code that appears (i.e., the setup chunk) and change the header below it from "R Markdown" to a the new title of "RMarkdown Coding Tips"

```
## R Markdown
```
Changed to =>
```
## R Markdown Coding Tips
```

2.) Delete all the remaining text/code after this title.

3.) After the title above, create text similar to below using the Markdown formatting features to match the text including anything hyperlinked (e.g., link the Github text), bolded, italicized or bulleted:

Markdown is an **easy to use** format for writing reports. It resembles what you naturally write every time you compose an email. In fact, you may have already used markdown *without realizing it*. These websites all rely on markdown formatting

- Github
- StackOverflow
- Reddit

4.) Next, create a header 3 that will contain the text: **Chunks of Code**

5.) After the heading, add the following text (be sure to include any formatting such as bold, bullet points, etc.):

It is always a good idea to include titles for your chunks of R code. This allows other users to quickly find the code you may be referencing when working with other developers. Additionally, within each chunk of code, there is a number of options for how your code will be handled when you knit a file. These are can be included in "{r }" where you include the title.

Options include:

- **eval = FALSE** prevents the code from being evaluate (not run and no output created). This is useful if you want to stop a chunk of code from running but you will probably not use this in the course.
- **include = FALSE** runs the code, but doesn't show the code or results in the final document. Use for setup code that you don't want cluttering your report.
- **echo = FALSE** prevents code (but not results) from appearing in the finished report. Useful when presenting to management so the code doesn't show on report.
- **message = FALSE** or **warning = FALSE** prevents messages or warnings from appearing in the output file
- **results = 'hide'** hides printed output.

- **fig.show = 'hide'** hides plots.
- **error = TRUE** causes the render to continue even if the code returns an error (rarely used).

While these are useful options, most of the time for this class I would like to see your code and results so please use these options sparingly.

6.) Next, create a header 3 that will contain the text: **Packages and Data Imports in R**

7.) After the heading, add the following text (be sure to include any formatting such as bold, bullet points, etc.):

A good habit is to include all the libraries and data import statements at the beginning of your R Markdown document. This is commonly the first chunk of code that you create in the Markdown file.

A few tips when working with packages in Markdown include:

- Don't install a package in your Markdown file. This will install the package everytime the file is knit. To load new packages, do this in the console outside of the Markdown file.
- If a new package does need to be installed, include this in the comments at the top of your R Markdown file.
- You only need to load the library once in the Markdown file. So, if you are using tidyverse, you will only have library(tidyverse) included once at the beginning.
- When you load some packages, they actually have multiple packages included in the load. For example, when you load tidyverse, this will load ggplot2, dplyr, tidyr, readr, etc.

8.) Next, create a header 3 that will contain the text: **Examples**

9.) After the heading, add the following text (be sure to include any formatting such as bold, bullet points, etc.):

For this project, we will be using the tidyverse which you just installed in a previous assignment. Remember that when you load a package in R, it will remain loaded throughout the program (you don't need to continually load the package).

Below is an example of loading the package and importing the data:

10.) Insert a new chunk of R data code. Name this chunk of code: **Packages**. For this code, I want you to load the tidyverse library and import the US_Population.csv file from Canvas.

11.) Finally, knit the R Markdown File (it should still be named **Mod2Assign3Answer.docx** and will appear in the files pane at the bottom right of R Studio).

12.) Upload the **Mod2Assign3Answer.docx** file to Canvas.