STAT 435
Spring 2024

Name: William Franzen

Lab Assignment 5 (*20 points*)

Notes: ~~~~~~

- If you need help on an R function/command, type *?functionname* or *?commandname* and help for this function/command will appear in the Help window.
- Create a new folder on your desktop and name it **LA5_*your name***. Set this as your working directory in RStudio.
- In RStudio, open a blank source file (R Script) to work in, and make sure all History and Environment entries are cleared before you start your work on the following questions.
- In the R Script, add *Lab Assignment-5 (by **your name**)* as a comment line, and clearly label your answer to each question and part.

~~~~~~

1. Your goal is to construct a data frame that describes the main characteristics of eight planets in our solar system. The main features of a planet are:
   - The type of the planet (Terrestrial or Gas Giant).
   - The planet's diameter relative to the diameter of the Earth.
   - The planet's rotation across the sun relative to that of the Earth.
   - If the planet has rings or not (TRUE or FALSE).

   After doing some high-quality research online, the data on the necessary vectors: `planets`, `type`, `diameter`, `rotation`, and `rings` are obtained, and for convenience, they are created and stored in the R Script named **Planets**. To download this RScript, go to **Data, Script, and other R Files** folder under COURSE DOCUMENTS area of the Moodle Course page.

   Copy these five vectors from the Planets file, paste it into your RScript window, and run them.

   a) Using these five vectors, construct a data frame, naming it `planets_df`. Then, print the data frame to inspect its contents.

   b) Using `str()` make sure that you've actually created a data frame with 8 observations and 5 variables.
~~~~~~

2. In the previous question, you found out that both the `planets` and `type` columns of `planets_df` are character vectors. For the `type` column, this does not make sense, because a planet type is some kind of category. For the `planets` column, however, that contains the planet names, this is more logical.

   a) Encode the `type` vector in a factor, naming it `type_factor`. Then, print it.

   b) Next, overwrite the `type` column of `planets_df` using `type_factor`. Then, print the data frame `planets_df`.

   c) Display the structure of `planets_df` to assert that you got it right.

3. Rename the columns of `planets_df` using: `"NAME", "TYPE", "DIAMETER", "ROTATION", "HAS_RINGS"`. (Note: *Remember – R is case sensitive!*)
   As `planets_df` is already created, you'll want to use the `names()` function.
   Finally, print `planets_df` after you renamed the variables.

4. Revisit the data frame `planets_df` you created earlier.

   a) Select the type of Jupiter; store it in `jupiter_type`. Then, print it.

   b) Store the entire rotation column in `rotation` as a vector. Then, print it.

   c) Create a data frame, `closest_planets_df`, that contains all data on the first three planets. Then, print it.

   d) Likewise, build the data frame `furthest_planets_df` that contains all data on the last three planets. Then, print it.

   e) Select the diameter and rotation for the planet Venus, and save it in `venus_data`. Print and report the result here. DIAMETER = 0.949, ROTATION = -243.02.

   f) Select for the last five rows of only the diameter and assign this selection to `furthest_planets_diameter`. Then, print `furthest_planets_diameter`.

   g) Make use of the `$` sign to create the variable `rings_vector` that contains the entire `HAS_RINGS` variable in the `planets_df` data frame.

      Print the `rings_vector`; it should be a vector of logicals.

h) Assign to `planets_with_rings_df` all data in the `planets_df` data set for the planets with rings, that is, where `has_rings` is TRUE. (*Hint:* use `subset()` function)

Print the resulting data frame.

i) Create a data frame `small_planets_df` with planets that have a diameter smaller than the Earth (so smaller than 1, since diameter is a relative measure of the planet's diameter with respect to that of planet Earth). (*Hint:* use `subset()` function)

Print the resulting data frame.

j) Build another data frame, `slow_planets_df`, with the observations that have a longer rotation period than Earth (so absolute value of rotation greater than 1). Then, print it. *Hint:* The absolute value function `abs()` is a built-in function.

k) Write and run one-line of code that identifies the planet with the smallest diameter.

5. Suppose you've browsed the Internet and decided to add a column that lists the number of moons each of the planets has. Also, the planets' masses can be a cool addition. Here is the data on these two new variables:

	Earth	**Mercury**	**Venus**	**Mars**	**Jupiter**	**Saturn**	**Uranus**	**Neptune**
Moons:	1	0	0	2	67	62	27	14
Masses:	1.00	0.06	0.82	0.11	317.8	95.2	14.6	17.2

a) Add the number of moons to `planets_df` under the variable name "`MOON`".

b) In a similar fashion, add the planets' masses to `planets_df` under the variable name "`MASS`".

c) Then, print the data frame `planets_df` and inspect its contents.

d) Display the structure of `planets_df` to assert that you got it right.

e) Write and run one-line of code that identifies the planets that have rings and mass greater than 50. List those planets here: Jupiter, Satrun

f) Use `with()` function to calculate the average number of moons for the gas giant planets, and report the result here: 42.5

6. In the 90s, it was still widely believed that Pluto was the ninth planet of our solar system revolving around the sun. However, new discoveries have led this planet to be labeled as a "Dwarf planet". Because everybody deep down has a fascination for the nineties, let's pretend that Pluto is still a planet and add its information to `planets_df` (without the columns `MOON` and `MASS`).

   a) The data for Pluto is as follows:

      Type: Terrestrial planet
      Diameter: 0.18
      Rotation: -6.38
      Rings: FALSE

   Create a data frame for `Pluto`, and using `do.call()` function, append it to `planets_df` (without the columns `MOON` and `MASS` for any of the planets), and assign the result to `planets_df_ext`.

   b) Inspect the resulting data frame by printing it.

7. You would like to rearrange your `planets_df_ext` data frame such that it starts with the largest planet and ends with the smallest one. A sort on the `diameter` column, as you will.

   a) Assign to the variable `positions` the desired ordering for the new data frame that you will create in the next step.

   b) Now, create the data frame `largest_first_df`, which contains the same information as `planets_df_ext`, but with the planets in decreasing order of magnitude. Use the previously created variable `positions` as row indices to achieve this.

   c) Print `largest_first_df` to see what you've accomplished.

Save your RScript naming it **RScript*_your name_***. Then, email it along with this lab assignment file (with the blanks filled in and saved) to the professor at sgazioglu@mtech.edu.

Have '**Stat435 – LA5**' in the subject line of the e-mail.