STA 141A: Homework 2

Instructor: Akira Horiguchi

Student name: ABCDE FGHIJ; Student ID: 123456789

Due date: Apr 16, 2025 at 9 PM (PT)

The assignment must be done in an R Markdown or Quarto document. The assignment must be submitted by the due date above by uploading:

1. a .pdf file in GRADESCOPE (if you can knit/compile your .rmd to a .html file only, please save the created .html file as a .pdf file (by opening the .html file -> print -> save to .pdf)).

Email submissions will not be accepted.

Each answer has to be based on `R` code that shows how the result was obtained. The code has to answer the question or solve the task. For example, if you are asked to find the largest entry of a vector, the code has to return the largest element of the vector. If the code just prints all values of the vector, and you determine the largest element by hand, this will not be accepted as an answer. No points will be given for answers that are not based on `R`. This homework already contains chunks for your solution (you can also create additional chunks for each solution if needed, but it must be clear to which tasks your chunks belong).

There are many possible ways to write `R` code that is needed to answer the questions or do the tasks, but for some of the questions or tasks you might have to use something that has not been discussed during the lectures or the discussion sessions. You will have to come up with a solution on your own. Try to understand what you need to do to complete the task or to answer the question, feel free to search the Internet for possible solutions, and discuss possible solutions with other students. It is perfectly fine to ask what kind of an approach or a function other students use. However, you are not allowed to share your code or your answers with other students. Everyone has to write the code, do the tasks and answer the questions on their own.

During the discussion sessions, you may be asked to present and share your solutions.

```
set.seed(2025*2)  # do not change this; this helps to reproduce the "random" results
# install.packages("tidyverse")  # you might need to run this line of code
```

Problem 1

Consider the scores of students on exams:

```
scores
```

```
# A tibble: 5 x 4
     id midterm1 midterm2 final_exam
  <dbl>    <dbl>    <dbl>      <dbl>
1     1       80       90         85
2     2       NA      100         90
3     3       75       95         60
4     4       95       NA         60
5     5       95       98         NA
```

(a) Use the `filter()` function from `dplyr` to see which students did better on their final exam than on their midterm 1. Now do the same but instead using base R functions.

```
# put your code here
```

(b) Use the `select()` function from `dplyr` to select all columns that contain the character string "id". Now do the same but instead using base R functions.

```
# put your code here
```

(c) Use the `summarize()` function to find the average score for midterm 1 and the average score for midterm 2. (All averages should omit any `NA` values.) Now do the same but instead using base R functions (you might find the `colMeans()` function helpful).

```
# put your code here
```

(d) We now have some extra information! Students with `id` 1, 2, and 5 are in the `A` section of the class, while students with `id` 3 and 4 are in the `B` section. To the tibble `scores`, add a new column called `section` to the tibble `scores`, which contains the section for each student. Then, for each section, use the `summarize()` function to find the average score for midterm 1 and the average score for midterm 2. (All averages should omit any `NA` values.)

```
# put your code here
```

(e) To the tibble `scores`, add a new column called `final_grade` that is calculated by the following:

- midterm contribution is 40%, and consists of the largest of the two midterm scores, with missing data treated as zeros
- final exam counting for 60%, with missing data treated as zero

Hint: you might find the function `replace_na()` in `dplyr` useful. You might also find the function `pmax()` useful.

```
# put your code here
```

Problem 2

Consider the following dataset:

```
reviews
```

```
# A tibble: 5 x 2
     id reviewtext
  <dbl> <chr>
1     1 I had a great experience, the product was as described.
2     2 Good, but not great. There were some issues.  Awfully crowded.
3     3 The service was excellent and the staff was very helpful.
4     4 I had an awful time.
5     5 Excellent, excellent!
```

(a) Use the function `filter()` from `dplyr` to return a tibble with only the rows where the sequence of characters "great" appears in `reviewtext` (with any lowercase/uppercase combos). *Hint*: you might find the functions `str_to_lower()` and `str_detect()` in `stringr` useful.

```
# put your code here
```

(b) Provide code which modifies `reviews` to have two new columns, `excellent` and `awful`, where:

- `excellent` is `TRUE` if the sequence of characters "excellent" appears in `reviewtext` (with any lowercase/uppercase combos) and `FALSE` otherwise
- `awful` is `TRUE` if the sequence of characters "awful" appears in `reviewtext` (with any lowercase/uppercase combos) and `FALSE` otherwise
- *Hint*: you might find the functions `str_to_lower()` and `str_detect()` in `stringr` useful.

```
# put your code here
```

(c) Use the `summarize()` function to calculate the percent of reviews where "awful" appears and the percent of reviews where "excellent" appears (any lowercase/uppercase combos).

```
# put your code here
```

Problem 3

Consider the following dataset:

```
health_data
```

```
# A tibble: 3 x 5
  PatientID Weight_2019 Weight_2020 Height_2019 Height_2020
      <dbl>       <dbl>       <dbl>       <dbl>       <dbl>
1         1          70          72         170         171
2         2          65          68         165         166
3         3          80          82         180         181
```

(a) Transform this tibble into a tibble `long_health_data` so that there are four columns:

- PatientID, numeric type
- Weight, numeric type
- Height, numeric type
- Year, numeric type

Hint: Use `?pivot_longer` to read the documentation for the `names_to` and `names_sep` arguments of `pivot_longer()`.

```
# put your code here
```

(b) Transform the tibble `long_health_data` from part (a) into the tibble `wide_health_data` which is back in the wide format, with columns "Weight_2019", "Weight_2020", "Height_2019", "Height_2020". Check that your calculation was correct by checking `all.equal(wide_health_data, health_data)`.

```
# put your code here
```

Problem 4

The task is to explore the US census population estimates by county for 2022 from the package `usmap`. The tibble `countypop` has `3122` rows and `4` variables: `fips` is the 5-digit FIPS code corresponding to the county; `abbr` is the 2-letter state abbreviation; `county` is the full county name; `pop_2022` is the 2022 population estimate (in number of people) for the corresponding county. Each row of the data frame represents a different county or county equivalent. For simplicity, `county` stands also for a county equivalent, and District of Columbia for a 'state'.

```
# install.packages("usmap")  # you might need to run this line of code
library(usmap)
```

Without creating new functions, and without using `for` loops, answer the following questions.

(a) How many unique county names are there? How many unique fips codes are there? How many unique states (as encoded in `abbr`) are there?

```
# Your solution
```

(b) Which county has a population of more than 5 million people and has a fips code that starts with 0?

```
# Your solution
```

(c) What are the populations of the 10 largest counties (in terms of population)? *Hint*: you might find the functions `sort()` and `tail()` (or `head()`) useful.

```
# Your solution
```

(d) What is the largest county (in terms of population) in each state? *Hint*: you might find the functions `order()` and `tail()` (or `head()`) useful.

```
# Your solution
```

Problem 5

Pick a dataset from the World Health Organization Global Health Observatory data repository. For whichever dataset you choose, download the csv file and do some exploratory data analysis. In particular, make some plots to learn more about your data. (For inspiration, see Chapter 10 of R4DS2.)

- If you instead want to explore data from other sources, the following also do not require registration:
  - https://opendata.cern.ch/search?q=&f=type%3ADataset&l=list&order=desc&p=1&s=10&sort=mostrecent
  - https://www.bfi.org.uk/industry-data-insights
  - https://datahub.io/collections
  - https://www.earthdata.nasa.gov/
  - https://data.gov/
  - https://datasetsearch.research.google.com/

```
# Your solution
```