PHW251 Problem Set 6

your name here

today

Part 1

For this part we will work with fictional data comparing the efficacy of two interventions. The interventions took place across several states and cities, with slight variations in dates. The outcome is a continuous variable.

Question 1

There's missing data in this data set. Can you identify them? In the next question you will re-code these values to NA.

```
# your code here
```

1A. How many NAs did you find?

YOUR ANSWER HERE

1B. Are there other values you think may count as NA?

YOUR ANSWER HERE

Question 2

2A. For the other values you believe may also be NAs, re-code them as NA.

```
# your code here
```

2B. Print the head() of the dataframe

Question 3

Now that we've fixed our NA values, let's address the errors we see with city and state names. Let's fix these entries to have uniform naming where cities are properly capitalized and state abbreviations are in all capital letters. For example, we want to see "San Antonio" and "TX" rather than "san Antonio" and "tx".
3A. Use `distinct()` and `pull()` to see all the variations you need to account for.

3B. Then, use `case_when()` to fix the values.

We have provided the code to fix the variation for Georgia and Texas using `case_when()`. Expand this code to fix the state abbreviations for California and all the city names.

```
df <- df %>% mutate(state = case_when(state %in% c("GA", "gA", "ga", "G A") ~ "GA",
                                      state %in% c("TX", "tX", "tx") ~ "TX"))

# your code here
```

Question 4

4A. Format the date column into a date format using a lubridate function.

Ominously, these interventions all occurred on the 25th day of the month.

```
# your code here
```

Question 5

You may have noticed that some of the cities don't match their state. We can't, at least from our data, distinguish which value is correct (the city or the state). The correct city and state pairings are:

- Atlanta, GA
- Austin, TX
- San Antonio, TX
- Hayward, CA
- Oakland, CA

5A. Drop the rows with this city/state inconsistency.

One suggestion is to create a variable indicating whether to drop the row. If you performed this step correctly you should have 33 rows.

```
# your code here
```

5B. Print the unique combinations of city and state that are now in the data frame

Use the code below and modify if needed.

```
unique(df[,c("city", "state")])
```

```
## # A tibble: 15 x 2
##    city        state
##    <chr>       <chr>
##  1 atlanta     GA
##  2 Atlanta     GA
##  3 atlAnTa     TX
##  4 San Antonio TX
##  5 austin      TX
##  6 oakland     <NA>
##  7 Hayward     <NA>
##  8 hayward     GA
##  9 hayward     TX
## 10 atlanta     <NA>
## 11 san Antonio TX
## 12 iakland     <NA>
## 13 austin      <NA>
## 14 Haywarf     <NA>
## 15 hayward     <NA>
```

Question 6

Another issue: our interventions column has missing data. We have two interventions that occurred in these locations:

- Intervention 1: Hayward, Atlanta, San Antonio
- Intervention 2: Oakland, Atlanta, Austin

For all of the cities except Atlanta it's clear what intervention took place.

6A. In these clear instances, replace NAs with the appropriate intervention.

6B. For Atlanta, drop the observations with missing intervention data since we cannot determine which intervention occurred.

```
# your code here
```

6C. How many observations did you drop?

YOUR ANSWER HERE

Question 7

We have a few NAs in the outcome column. Our on-site researchers informed us that when a score of "0" was provided, the data collection team left the cell blank.

7A. Re-code the NAs to 0.

```
# your code here
```

7B. Use code to confirm that there are no longer any NAs in the outcome column.

```
# your code here
```

Question 8

8A Use ggplot to create a box plot comparing the two interventions and their outcome.

The outcome is a continuous variable from 0 to 10. You may need to factor one of your variables. Look at the visualization cheatsheet if you don't know the "geom" for creating a boxplot.

```
# your code here
```

Part 2

For this part we will use *fictional* data inspired by research on non-deceptive or open-label placebos. Non-deceptive placebos are placebos but without the deception. Some studies have found suggestions that, despite not being tricked, participants are reporting similar benefits to what they would have with placebos! You can read more here:

NPR: Is A Placebo A Sham If You Know It's A Fake And It Still Works?

Nature Communications: Placebos without deception reduce self-report and neural measures of emotional distress

In this fictional data we conducted an experiment across two university sites to investigate whether non-deceptive placebos decreased self-report pain ratings. There were three groups: control, placebo, and non-deceptive placebo. Each participant completed a pre- and post- pain induction task and provided a pain rating. All participants completed the same procedures during the pre-test. Only during the post-test did participants in the intervention arms (placebo, non-deceptive) receive additional instructions prior to the pain induction task (i.e., placebo or non-deceptive placebo ratings).

Data coding:

- ID: Contains participant ID number, a letter to indicate group, and pre or post tags.
  C = Control P = Placebo N = Non-deceptive
- LOCATION: Research Site
- PAIN RATING: Self report of pain based on a 0-10 scale
- DATE: Date of observation

Question 9

9A Read in the data.

To make it slightly more challenging we have changed the format from a .csv to .xlsx and "hidden" the data one level deeper in the /data folder. Take a look at the data to get oriented. Please use "placebo_df" as the name of your data frame.

```
# your code here
```

Question 10

It's a bit difficult to tell what group (control, placebo, or non-deceptive placebo) each participant is in with their IDs combined with their grouping.

10A. Create a new column called "GROUP" based on the letter assignment for IDs.

The stringr function 'str_detect()' will be useful here!

```
# your code here
```

10B. Print the head() of the dataframe

```
# your code here
```

Question 11

We have a similar issue telling apart the pre- and post- observations.

11A. Create a new column called "TEST" that distinguishes whether the observation is a pre- or post-test.

Unfortunately, the two research sites were not consistent in their naming convention. You will need to consider the different cases.

```
# your code here
```

11B. Print the head() of the dataframe

```
# your code here
```

Question 12

There were differences in the formatting for dates across the two research sites.

12A. Create a new column called "DATE_FIX" that grabs only the date. Make sure this new date column takes the following format: yyyy-mm-dd

Hint: Check out ?parse_date_time

```
# your code here
```

12B. Print the head() of the dataframe

```
# your code here
```

Question 13

You realize there was a strange error in your excel file that, for every date, pushed the date forward by 1 year. Rather than editing your excel sheet and potentially making an incorrect permanent change to your raw data you decide to fix the error in R.

13A. Create a new column called "DATE_FIX_2" that fixes the date.

```
# your code here
```

Question 14

14A. Clean up the data frame by removing DATE and DATE_FIX.

14B. Afterwards, rename DATE_FIX2 to DATE

```
# your code here
```

14C. Print the head() of the dataframe

```
# your code here
```

Question 15

We're interested in plotting our data to begin digging into the results. Below is dplyr and ggplot code to do this.

15A. Uncomment and run the following code as-is (visualization is not the focus of this problem set).

You may need to install ggthemes.

```
# install.packages("ggthemes")
# library(ggthemes)
#
# df_plot <- placebo_df %>%
#   group_by(GROUP, LOCATION) %>%
#   summarize(MEAN_PAIN = mean(PAIN_RATE))
#
# ggplot(df_plot, aes(x = LOCATION, y = MEAN_PAIN, fill = GROUP)) +
#   geom_col(position = "dodge") +
#   ylim(0, 10) +
#   theme_few() +
#   scale_fill_few("Medium") +
#   theme(axis.title = element_blank(),
#         axis.title.y = element_text()) +
#   labs(fill = "Group",
#        title = "Non-deceptive placebo study",
#        y = "Pain rating")
```

For a quick first pass we think this visualization isn't so bad. However, logically, we think that the order of the groups should be: Control, Placebo, Non-deceptive.

15B. Make GROUP into a factor that reflects this order.

If done correctly, when you re-run the above chunk, the plot should show the bars in that order

```
# your code here
```

You're done! Please knit to pdf and upload to gradescope.