Homework #4

Question 1

This problem involves constructing a dataset to organize observations for future study.

Begin by downloading here a compressed folder containing images of fungi growth started in petri dishes. While the images themselves will be processed at a later date, for now we need to create a record of current observations. This will involve constructing a dataset that has the following variables:

- **Plate:** A record of the plate number, an integer value 1-50
- **Strain:** The strain of fungi being recorded, taking the values T2, T4, or T7
- **Competitor:** A value of `"C"` or `"NC"` according to whether or not the fungi sample is grown with a competing species
- **Time_Plated:** The date and time the fungi were plated. For each observation, this is the same value: February 15, 2026 at 8AM
- **Time_Observed:** The date and time the image was generated. This should be reported in 24H time (e.g., 16:00 instead of 4PM)
- **Duration:** Amount of time between plating and observation

All of this information is recorded in the file or path name of the images (the image files are empty). Your submission should include the code to generate this database, along with the use of `head()` to show the first 10 rows.

Solutions (Based off solution by student)

There are a bunch of ways to do this. As long as their output looks somewhat like what I have here

```
library(lubridate)
library(tidyr)
library(dplyr)

## Read in from local
files <- list.files("~/courses/sta230/s26/hw/4hw",recursive=TRUE)
files <- files[4:length(files)]

fungi_files <- files %>% data.frame(time_plated="February 15 2026 8:00am") %>%
  mutate(filenames=str_remove_all(., ".HEIC")) %>%
  select(filenames, time_plated) %>%
  separate(col=filenames,
           into=c("Strain", "filenames"),
           sep="/") %>%
  separate(col=filenames,
           into=c("Plate", "Competitor", "Time_Observed"),
           sep="_",
           extra="merge") %>%
  mutate(Time_Observed = str_replace_all(Time_Observed, c("_|-"), " ")) %>%
  mutate(Time_Observed = ymd_h(Time_Observed),
         time_plated = mdy_hm(time_plated),
         duration = difftime(Time_Observed, time_plated, units="hours"))
fungi_files %>% head(n=10)
```

```
##    Strain Plate Competitor       Time_Observed         time_plated  duration
## 1      T2     1          C 2026-02-23 16:00:00 2026-02-15 08:00:00 200 hours
## 2      T2    10          C 2026-02-24 08:00:00 2026-02-15 08:00:00 216 hours
## 3      T2    11          C 2026-02-20 14:00:00 2026-02-15 08:00:00 126 hours
## 4      T2    12          C 2026-02-21 09:00:00 2026-02-15 08:00:00 145 hours
## 5      T2    13          C 2026-02-25 08:00:00 2026-02-15 08:00:00 240 hours
## 6      T2    14          C 2026-02-22 11:00:00 2026-02-15 08:00:00 171 hours
## 7      T2    15          C 2026-02-25 13:00:00 2026-02-15 08:00:00 245 hours
## 8      T2    16          C 2026-02-18 11:00:00 2026-02-15 08:00:00  75 hours
## 9      T2    17          C 2026-02-25 12:00:00 2026-02-15 08:00:00 244 hours
## 10     T2    18          C 2026-02-17 12:00:00 2026-02-15 08:00:00  52 hours
```

Question 2

Bootstrapping is a statistical re-sampling technique that is used to construct confidence intervals for a given statistic. The general idea of the bootstrap is that from a given sample, we can construct "new" samples by simply resampling *with replacement* from the original. The general process works like this:

1. Begin with a vector `x`. This could be the column of a data.frame (e.g., `df$x`)
2. Decide on a statistic `statistic` that we wish to bootstrap. This should be a function of `x`
3. Decide on a number of bootstraps samples, `B`. A typical value is `B = 1000`, but this should be able to change
4. For each bootstrap, we should collect a `sample()` *with replacement* from the vector `x` the same length as `x`. See `?sample()`
5. From each sample, we should find the value of `stat(x)` and record it. We will end up with a length `B` vector of statistics
6. Return this vector as a data.frame

From this vector of statistics, we should then:

- Find the mean. This is our estimate of the statistic
- Find the quantiles of this vector using `quantile()`. This should give us a confidence interval

Here is an example of using a bootstrap function to find the trimmed mean of a vector

```
## Here is my data
set.seed(123)
x <- rnorm(n = 25, mean = 10, sd = 2)

## Here is trimmed mean function
trim_mean <- function(x) {
  mean(x, trim = 0.1)
}

## Calling my bootstrap function with 3 arguments, returns a data.frame
boot <- bootstrap(x = x, statistic = trim_mean, B = 500)

## Find mean and 90% confidence interval
mean(boot$Statistic)
```

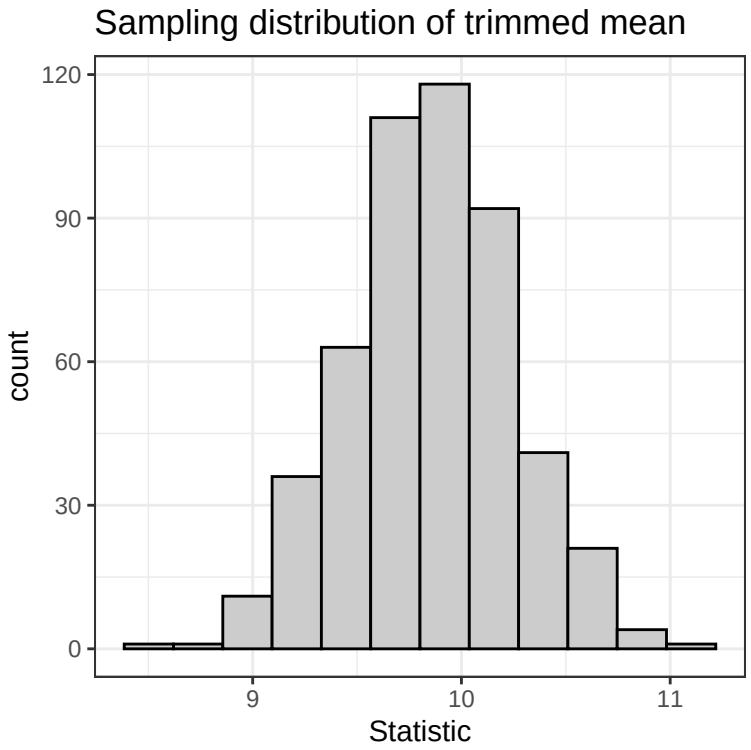
```
## [1] 9.8558
```

```
quantile(boot$Statistic, probs = c(0.05, 0.95))
```

```
##      5%     95%
##  9.2312 10.5353
```

By returning a data.frame, we are making it easy to quickly plot the sampling distribution of a statistic.

```r
library(ggplot2)
ggplot(boot, aes(Statistic)) +
  geom_histogram(color = "black", fill = "gray80", bins = 12) +
  ggtitle("Sampling distribution of trimmed mean")
```



- **Part 1:** Create your own function `bootstrap()` that behaves like the one above. You can verify it works by passing in the same `statistic` function and randomly generated `x` from above.
- **Part 2:** We are going to use our bootstrapping function to create a new statistic to investigate the Grinnell rainfall data `rainsub` given below:

```r
## Load data
rain <- read.csv("https://collinn.github.io/data/grinnell_rain.csv")

## Subset
set.seed(10)
idx <- sample(1:nrow(rain), size = 20)
rainsub <- rain[idx, ]
```

- First, create a function that generates a statistic consisting of a ratio of the mean to the median (i.e., `mean(x) / median(x)`). If this ratio is close to one, the mean and the median are similar and the data are not skewed. If it is larger or smaller than one, there is evidence of skew.
- Using this statistic, bootstrap the sample `rainsub` with `B = 1000` to generate an estimate of the sampling distribution. Find a 90% confidence interval for this statistic. Does this interval contain 1?
- Write your conclusion to the null hypothesis: There is no skew present in the distribution of rainfall in Grinnell, IA

```r
## Here is a version of the function
```

```r
bootstrap <- function(x, statistic, B) {
  v <- replicate(B, statistic(sample(x, replace = TRUE)))
  data.frame(Statistic = v)
}

## Find the confidence interval
rain <- read.csv("https://collinn.github.io/data/grinnell_rain.csv")

## Subset
set.seed(10)
idx <- sample(1:nrow(rain), size = 20)
rainsub <- rain[idx, ]

f <- function(x) mean(x) / median(x)

bs <- bootstrap(rainsub$precip, statistic = f, B = 1000)

## 90% CI
quantile(bs$Statistic, probs = c(0.05, 0.95))
```

Solution

```
##      5%     95%
## 0.81088 2.49502
```

Based on 90% confidence interval which contains value of 1, we fail to reject the hypothesis that there is no skew