Lab 01 Report

YOUR NAME HERE

DATE HERE

Skills

```
# load package of helpful datasets and functions
library(tidyverse)

# load sleep dataset
data("msleep")

?msleep

# print out the first few rows
head(msleep)
```

```
## # A tibble: 6 x 11
##   name  genus vore  order conservation sleep_total sleep_rem sleep_cycle
##   <chr> <chr> <chr> <chr> <chr>              <dbl>     <dbl>       <dbl>
## 1 Chee~ Acin~ carni Carn~ lc                  12.1      NA        NA
## 2 Owl ~ Aotus omni  Prim~ <NA>                17         1.8      NA
## 3 Moun~ Aplo~ herbi Rode~ nt                  14.4       2.4      NA
## 4 Grea~ Blar~ omni  Sori~ lc                  14.9       2.3       0.133
## 5 Cow   Bos   herbi Arti~ domesticated         4         0.7       0.667
## 6 Thre~ Brad~ herbi Pilo~ <NA>                14.4       2.2       0.767
## # ... with 3 more variables: awake <dbl>, brainwt <dbl>, bodywt <dbl>
```

Below is an example code chunk. Use chunks like these to run the Lab 01 code. You do not need to turn anything in for the Skills section of Lab 01.

```
# YOUR CODE HERE
```

Practice

1. The `ChickWeight` dataset provides data on a sample of fifty chicks, each fed one of four different diets. For each chick, the data frame includes weight measurements at birth and every other day thereafter. Let's first take a look at the birth weights of the chicks. Here, we use the term in brackets `[ChickWeight$Time == 0]` to take a subset of our `ChickWeight$weight` vector. The line below says we want all of the `weight` values in `ChickWeight` for which `Time == 0`, meaning that the time is zero. We use the double equals sign `==` to test for equality.

```
ChickWeight$weight[ChickWeight$Time == 0]
```

```
##  [1] 42 40 43 42 41 41 41 42 42 41 43 41 41 41 41 41 42 39 43 41 40 41 43
## [24] 42 40 42 39 39 39 42 42 41 39 41 41 39 41 41 42 41 42 42 42 42 41 40
## [47] 41 39 40 41
```

2. The last day of the experiment is Day 21. Print out the ending weights of the chicks.

3. Calculate the mean ending weight of the chicks.

4. Next, let's take a look at the starting weights of the chicks that ate Diet 1. Below, the `&` sign indicates that we want only the weight measurements that were from time zero **and** from chicks that ate Diet 1.

```
ChickWeight$weight[ChickWeight$Time == 0 & ChickWeight$Diet == 1]
```

```
##  [1] 42 40 43 42 41 41 41 42 42 41 43 41 41 41 41 41 42 39 43 41
```

5. Print out the ending weights of the chicks that ate Diet 1.

6. For each diet, calculate the mean ending weight of the chicks that ate that diet. Is this a good way to compare the diets?

7. Make a histogram of all of the chicks' birthweights. Use the x-axis label "Birth Weights" and the title "Frequency of Birthweights". Provide a one-sentence summary of the plot.

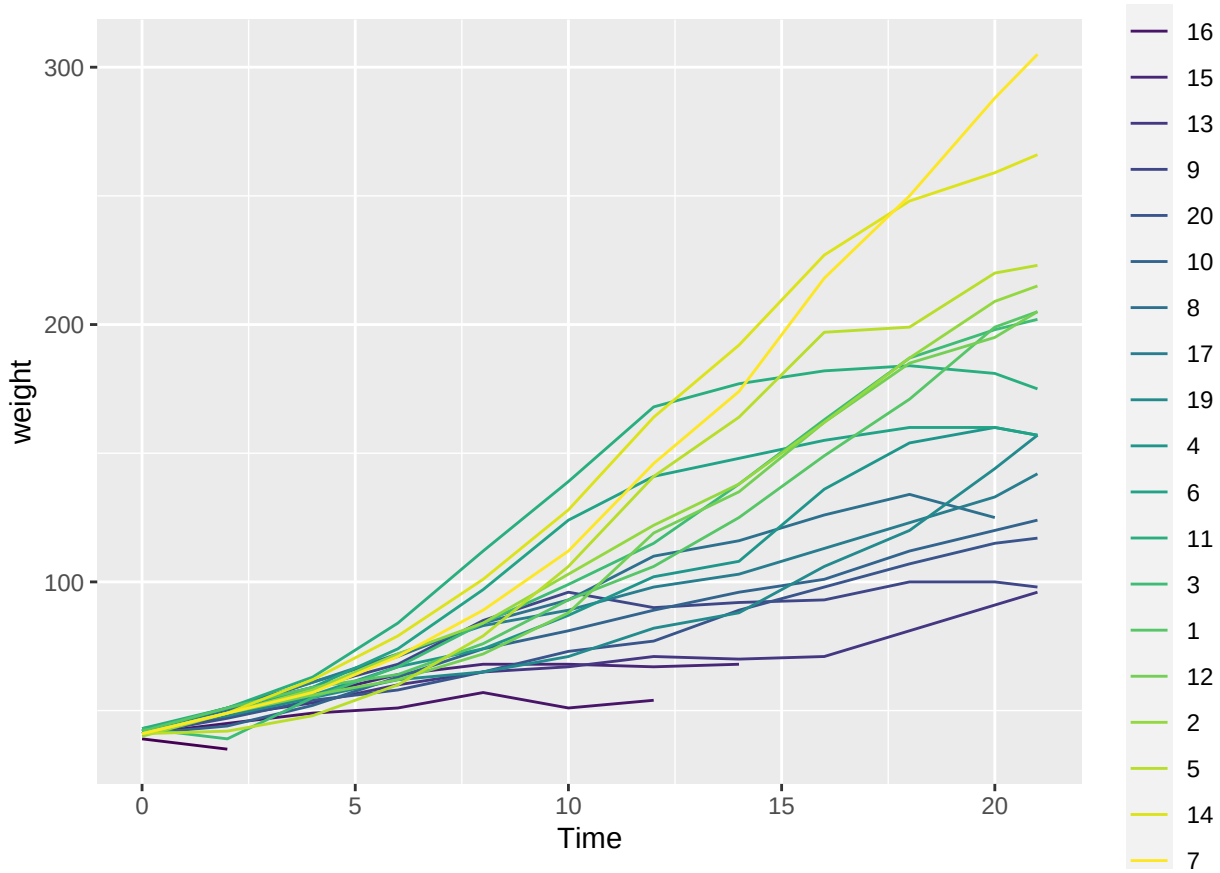
8. Make a scatterplot with weight on the y-axis, and time on the x-axis. Use an appropriate title and axis labels, and change the color and shape of the points from their default values. Provide a one-sentence summary of the plot. Additionally, give one reason this is a good way to plot this data, and one reason this plot may be incomplete.

9. Finally, let's look at another way to subset our data. Below, we introduce the pipe operator `%>%`. The below code essentially says to take our `ChickWeight` data and include only the rows that satisfy `Diet == 1`, and store it into a new data frame named `diet_one`.

```
diet_one <- ChickWeight %>%
  filter(Diet == 1)
```

10. The below line of code introduces a new plotting package called `ggplot2`, which can be difficult to learn at first but allows us to quickly produce complicated plots that display many variables at once. The first part `ggplot(diet_one)` tells R we want to generate a ggplot based on the data in `diet_one`. The `geom_line()` function tells R we want to add line(s) to our plot. Finally, within `geom_line()`, we provide aesthetics `aes(x = Time, y = weight, color = Chick)` that tell R we want the x-values for our lines to be times, the y-values to be weights, and the different colors should represent different chicks. **Provide a one-sentence summary of the resulting plot.**

```
ggplot(diet_one) + geom_line(aes(x = Time, y = weight, color = Chick))
```



11. Generate your own visualization of the `ChickWeight` data. You can use any of the functions we've used above, but you are encouraged to think about other kinds of plots and figure out how to make them in R. Provide a one-sentence summary of your plot. Additionally, give one reason this is a good way to plot this data, and one reason your plot may be incomplete.

12. Which diet produces the most weight gain? If you're not sure how to compute this in R, that is okay. If you can describe the set of steps you would take to compare the diets, that will be enough.