Computational Thinking 2 - Conditionals

Alex Stadlinger & Ellie Kuskie

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.2     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.4
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becor
```

```
library(here)
```

```
here() starts at C:/Users/conno/OneDrive/Documents/UCSC Undergrad/DataScience4EEB/Computatio
```

1

1.1

```
x <- 5

# Check if the value of x is greater than 10
if(x > 10)
```

```
{
  # Paste takes the value stored in x and combines that with a character string
  print(paste(x, "is greater than 10"))
} else
{
  print(paste(x, "is less than or equal to 10"))
}
```

```
[1] "5 is less than or equal to 10"
```

Q1.1

```
x <- 11

# Check if the value of x is greater than 10
if(x > 10)
{
  # Paste takes the value stored in x and combines that with a character string
  print(paste(x, "is greater than 10"))
} else
{
  print(paste(x, "is less than or equal to 10"))
}
```

```
[1] "11 is greater than 10"
```

output: “11 is greater than 10”

```
x <- "five"

# Check if the value of x is greater than 10
if(x > 10)
{
  # If x is > 10, multiple x by 2
```

```
  print(x*2)
} else
{
  # If x is not > 10, divide x by 2
  print(x/2)
}
traceback()
```

```
# define a variable
x <- 11

# check the value of x using nested if-else statements
if (x < 10) {
  # if x is less than 10
  print("x is less than 10")
} else {
  # if x is exactly equal to 10
  if (x == 10) {
    print("x is 10!!!")
  } else {
    # if x is greater than 10
    print("x is greater than 10")
  }
}
```

```
[1] "x is greater than 10"
```

```
vec <- c(9, 10, 11, 12)
```

```
# For 1 through the length of the vector "vec"
for (i in 1:length(vec)) {

  # check the value of using nested if-else statements
  if (vec[i] < 10) {
    # if the element is less than 10
    print("value is less than 10")
```

```
  } else {
    # if the element is exactly equal to 10
    if (vec[i] == 10) {
      # if the element equals 10
      print("value is 10!!!")
    } else {
      # if the element is greater than 10
      print("value is greater than 10")
    }
  }

}
```

```
[1] "value is less than 10"
[1] "value is 10!!!"
[1] "value is greater than 10"
[1] "value is greater than 10"
```

Q1.2

```
y <- c(-2, 42, 0, 10)

# For 1 through the length of the vector "vec"
for (i in 1:length(y)) {

  # check the value of using nested if-else statements
  if (y[i] < 0) {
    # if the element is negative
    print("value is negative")
  } else {
    # if the element is exactly equal to 0
    if (y[i] == 0) {
      # if the element equals 0
      print("value is 0!!!")
    } else {
      # if the element is positive
      print("value is positive")
```

```
    }
  }

}
```

```
[1] "value is negative"
[1] "value is positive"
[1] "value is 0!!!"
[1] "value is positive"
```

1.2

```
library(lterdatasampler)
```

Q1.3

```
?nwt_pikas
```

```
starting httpd help server ... done
```

Stress was measured by observing the amount of glucocorticoid metabolite present in pika feces. The variable is called concentration_pg_m and the units are in picogram GCM/gram dry pika feces

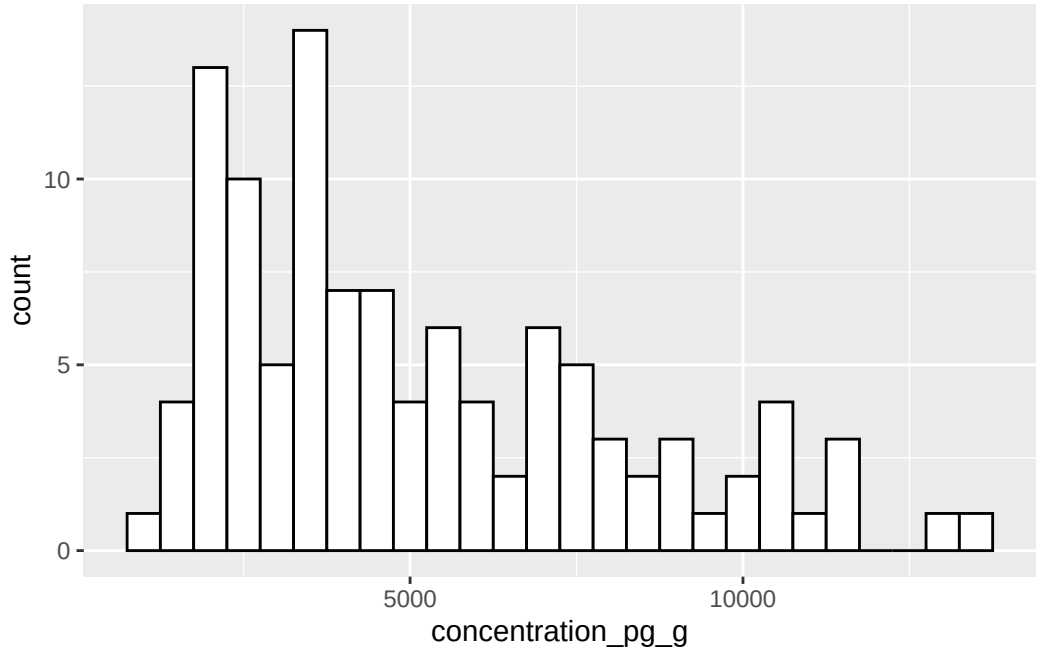
---

Q1.4

```
nwt_pikas <- nwt_pikas
```

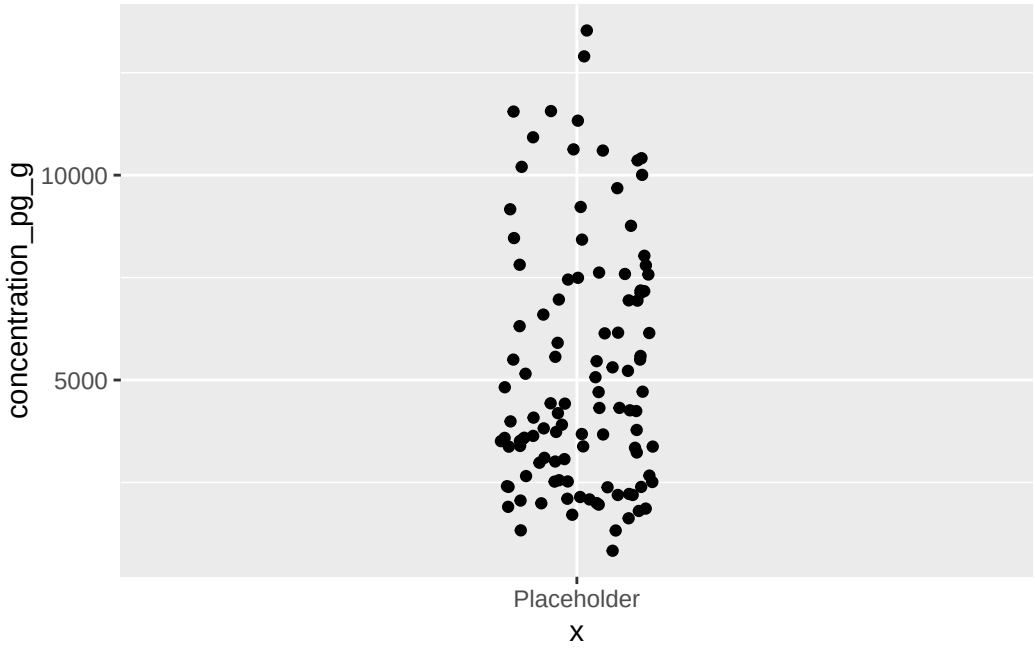
each row represents an individual pika poop

```
# Make a histogram
nwt_pikas %>%
  ggplot(aes(x = concentration_pg_g)) +
  # Add the histogram geom, which only needs an x-axis
  # Choose a binwidth of 500 picogram GCM/gram
  geom_histogram(binwidth = 500,
                 fill = "white",
                 color = "black")
```




```
# Make a scatterplot with jittered points
nwt_pikas %>%
  # We're adding a little placeholder axis just so we can see the point distribution
  ggplot(aes(x ="Placeholder",
```

```
            y = concentration_pg_g)) +
  # Add the geom_jitter geom
  geom_jitter(width = 0.1)
```



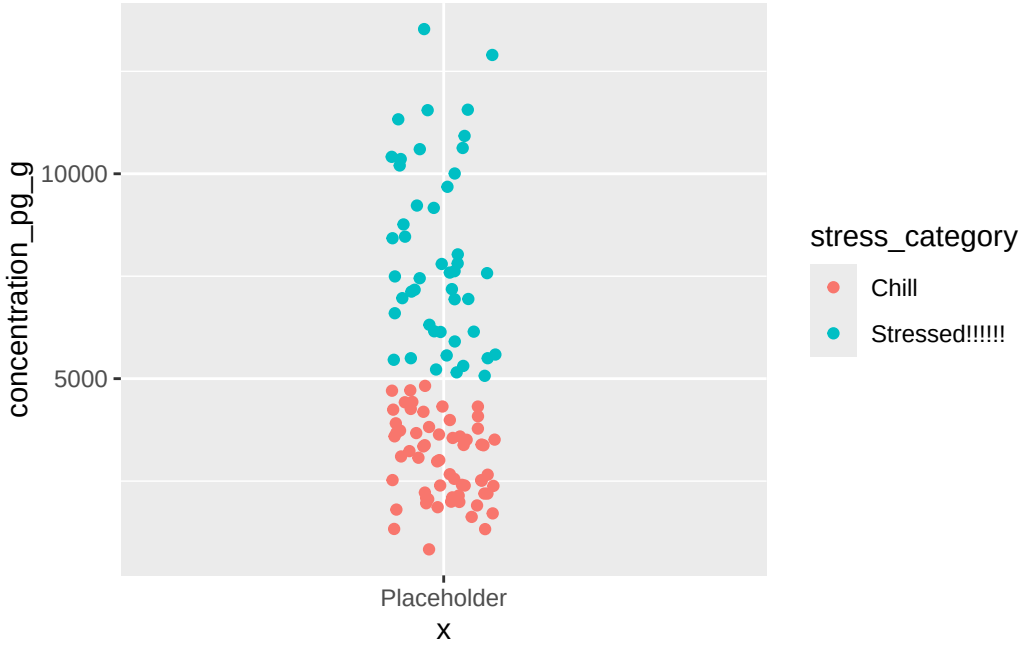

```
nwt_pikas_categ <- nwt_pikas %>%
  # Call the new column stress_category
  mutate(stress_category = case_when(
    # When the value is > 5000, make the new column's value "Stressed!!!!"
    concentration_pg_g > 5000 ~ "Stressed!!!!!!",
    # Otherwise, make the new column's value "Chill"
    .default = "Chill"
  ))

# Check out the first 6 rows, but remove the utm columns just for visibility
head(nwt_pikas_categ %>% select(-c(utm_easting, utm_northing)))
```

```
# A tibble: 6 x 7
  date       site       station  sex   concentration_pg_g elev_m stress_category
  <date>     <fct>      <fct>    <fct>              <dbl>  <dbl> <chr>
1 2018-06-08 Cable Gate Cable G~ male              11563.  3343. Stressed!!!!!!
2 2018-06-08 Cable Gate Cable G~ male              10629.  3353. Stressed!!!!!!
3 2018-06-08 Cable Gate Cable G~ male              10924.  3358. Stressed!!!!!!
```

```
4 2018-06-13 West Knoll West Kn~ male              10414.  3578. Stressed!!!!!!
5 2018-06-13 West Knoll West Kn~ male              13531.  3584. Stressed!!!!!!
6 2018-06-13 West Knoll West Kn~ <NA>               7799.  3595. Stressed!!!!!!
```

Q1.5

```
# Make a scatterplot with jittered points
nwt_pikas_categ %>%
  # We're adding a little placeholder axis just so we can see the point distribution
  ggplot(aes(x ="Placeholder",
             y = concentration_pg_g,
             color = stress_category)) +
  # Add the geom_jitter geom
  geom_jitter(width = 0.1)
```




```
nwt_pikas_categ2 <- nwt_pikas_categ %>%
  # Create a new column called month
  # then, extract the month from the date using the month() function
```

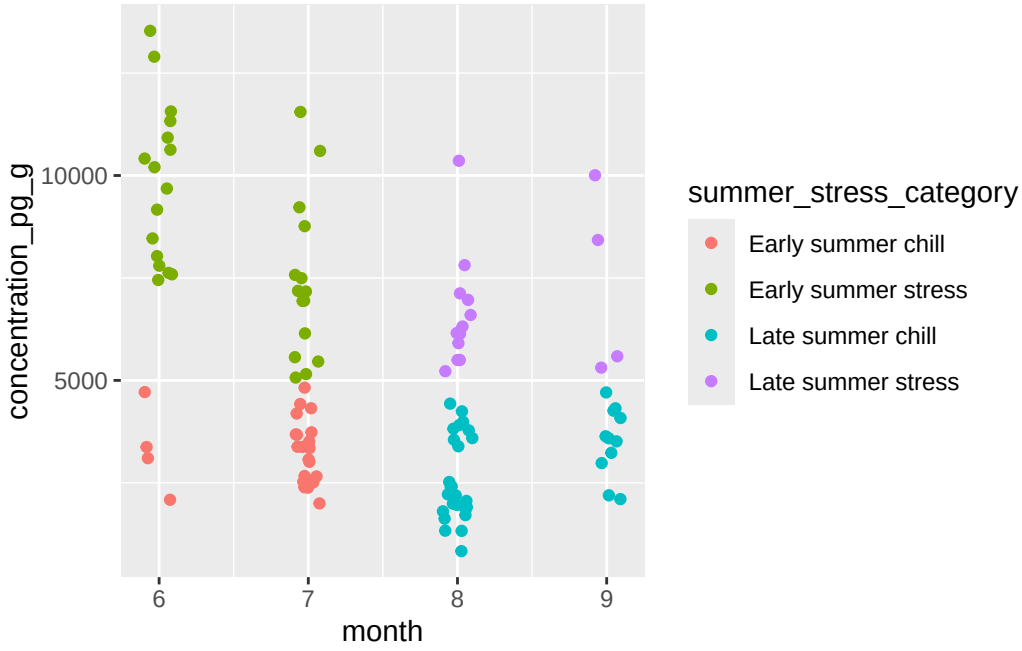
```
  mutate(month = month(date)) %>%
  # Lastly, relocate the month column after the date column so it's more easily visible to us
  relocate(month, .after = date)

head(nwt_pikas_categ2)
```

```
# A tibble: 6 x 10
  date       month site       station      utm_easting utm_northing sex
  <date>     <dbl> <fct>      <fct>              <dbl>        <dbl> <fct>
1 2018-06-08     6 Cable Gate Cable Gate 1      451373      4432963 male
2 2018-06-08     6 Cable Gate Cable Gate 2      451411      4432985 male
3 2018-06-08     6 Cable Gate Cable Gate 3      451462      4432991 male
4 2018-06-13     6 West Knoll West Knoll 3      449317      4434093 male
5 2018-06-13     6 West Knoll West Knoll 4      449342      4434141 male
6 2018-06-13     6 West Knoll West Knoll 5      449323      4434273 <NA>
# i 3 more variables: concentration_pg_g <dbl>, elev_m <dbl>,
#   stress_category <chr>
```

```
nwt_pikas_summerstress <- nwt_pikas_categ2 %>%
  mutate(summer_stress_category = case_when(
    (month == 6 | month == 7) & concentration_pg_g > 5000 ~ "Early summer stress",
    (month == 6 | month == 7) & concentration_pg_g <= 5000 ~ "Early summer chill",
    (month == 8 | month == 9) & concentration_pg_g > 5000 ~ "Late summer stress",
    (month == 8 | month == 9) & concentration_pg_g <= 5000 ~ "Late summer chill",
    .default = "NA"
  ))

head(nwt_pikas_summerstress)
```

```
# A tibble: 6 x 11
  date       month site       station      utm_easting utm_northing sex
  <date>     <dbl> <fct>      <fct>              <dbl>        <dbl> <fct>
1 2018-06-08     6 Cable Gate Cable Gate 1      451373      4432963 male
2 2018-06-08     6 Cable Gate Cable Gate 2      451411      4432985 male
3 2018-06-08     6 Cable Gate Cable Gate 3      451462      4432991 male
4 2018-06-13     6 West Knoll West Knoll 3      449317      4434093 male
5 2018-06-13     6 West Knoll West Knoll 4      449342      4434141 male
6 2018-06-13     6 West Knoll West Knoll 5      449323      4434273 <NA>
# i 4 more variables: concentration_pg_g <dbl>, elev_m <dbl>,
#   stress_category <chr>, summer_stress_category <chr>
```

```
# Make a scatterplot with jittered points
nwt_pikas_summerstress %>%
  # We're adding a little placeholder axis just so we can see the point distribution
  ggplot(aes(x = month,
             y = concentration_pg_g,
             color = summer_stress_category)) +
  # Add the geom_jitter geom
  geom_jitter(width = 0.1)
```



2

Q2.1

We are using the iris dataset.

```
iris <- iris
```

Q2.2

We are going to try to arbitrarily classify whether a flower is "big" or small" based on its sepal length. We are using if/else statements to iterate through the column.

Q2.3

```
#assign this column to a vector
sepal_length <- iris$Sepal.Length

for (i in 1:20) {
    #determine if flower is big
    if (sepal_length[i] >= 5.5) {
      print("flower is big")
      #determine is flower is small
  } else {
      print("flower is small")
    }
}
```

```
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
[1] "flower is small"
```

```
[1] "flower is small"
[1] "flower is big"
[1] "flower is big"
[1] "flower is small"
[1] "flower is small"
[1] "flower is big"
[1] "flower is small"
```

```
#Ouput is REALLY long, so we will not iterate through the whole vector. We will only do the i
```

Q2.4

We are going to try to arbitrarily classify whether a flower is "big" or small" based on what species it is using sepal length. Use case_when() to iterate through the dataset based on species type and sepal length, taking both into consideration. This will make our previous for loop better by making the output into a column in the dataset.

Q2.5

```
#make new dataset using iris
iris_size <- iris %>%
  #use mutate to make new column for flower size
  #use case_when() to set instructions for new column
  mutate(flower_size = case_when(
    #separate based on species and sepal length
    Species == "setosa" & Sepal.Length >= 5.0 ~ "Big",
    Species == "setosa" & Sepal.Length < 5.0 ~ "Small",
    Species == "versicolor" & Sepal.Length >= 6.5 ~ "Big",
    Species == "versicolor" & Sepal.Length < 6.5 ~ "Small",
    Species == "virginica" & Sepal.Length >= 6.5 ~ "Big",
    Species == "virginica" & Sepal.Length < 6.5 ~ "Small"
  ))

head(iris_size)
```

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	flower_size
1	5.1	3.5	1.4	0.2	setosa	Big
2	4.9	3.0	1.4	0.2	setosa	Small
3	4.7	3.2	1.3	0.2	setosa	Small
4	4.6	3.1	1.5	0.2	setosa	Small
5	5.0	3.6	1.4	0.2	setosa	Big
6	5.4	3.9	1.7	0.4	setosa	Big