Computational Thinking 2

Abbie & Sam

Activity 8: Computational thinking 2: conditionals

Read in packages

```
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.6
v forcats   1.0.0     v stringr   1.5.2
v ggplot2   4.0.1     v tibble    3.3.0
v lubridate 1.9.4     v tidyr     1.3.2
v purrr     1.1.0
-- 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 becom
```

```
library(here)
```

```
here() starts at /Users/Abbie1/Documents/Repositories/BIOE 176 DataScience4EEB/Comp-
Thinking-2
```

1. Conditionals

1.1 if else statements

Example of ifelse

```r
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: Modify the value of x

```r
x <- 11

# Check if the value of x is greater than 10
if(x > 10)
{
  print(paste(x, "is greater than 10"))
} else
{
  print(paste(x, "is less than or equal to 10"))
}
```

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

This shows that x is greater than 10. It executed the first function since that was true.

Using `traceback()` for errors

```r
x <- 5

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

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

```
[1] 2.5
```

Changing x<- to x <- “five”

```r
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)
}
```

```r
traceback()
```

```
No traceback available
```

Adding another condition

Reran for x=9, 10, and 11

```r
# 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"
```

For loops and ifelse

```
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: Create a new for loop + if else statement

Create vector

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

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

  # check the value of using nested if-else statements
  if (y[i] < 0) {
    # if the element is less than 0
    print("value is less than 0")
  } 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 greater than 0
      print("value is greater than 0")
    }
  }

}
```

```
[1] "value is less than 0"
[1] "value is greater than 0"
[1] "value is 0!!!"
[1] "value is greater than 0"
```

1.2 `case_when()` and pikas

Load Packages

```r
library(lterdatasampler)
library(tidyverse)
```

Q1.3: How do the researchers measure pika stress?

```
?lterdatasampler
?nwt_pikas
```

concentration_pg_m a number denoting the glucocorticoid metabolite (GCM) concentration in picogram GCM/gram dry pika feces

This column in the data set measures pika stress via feces samples as glucocorticoid metabolite (GCM) in picogram GCM/gram. "Stress was measured by observing the amount of glucocorticoid metabolite present in pika feces."

Q1.4: What does a row represent in this data?

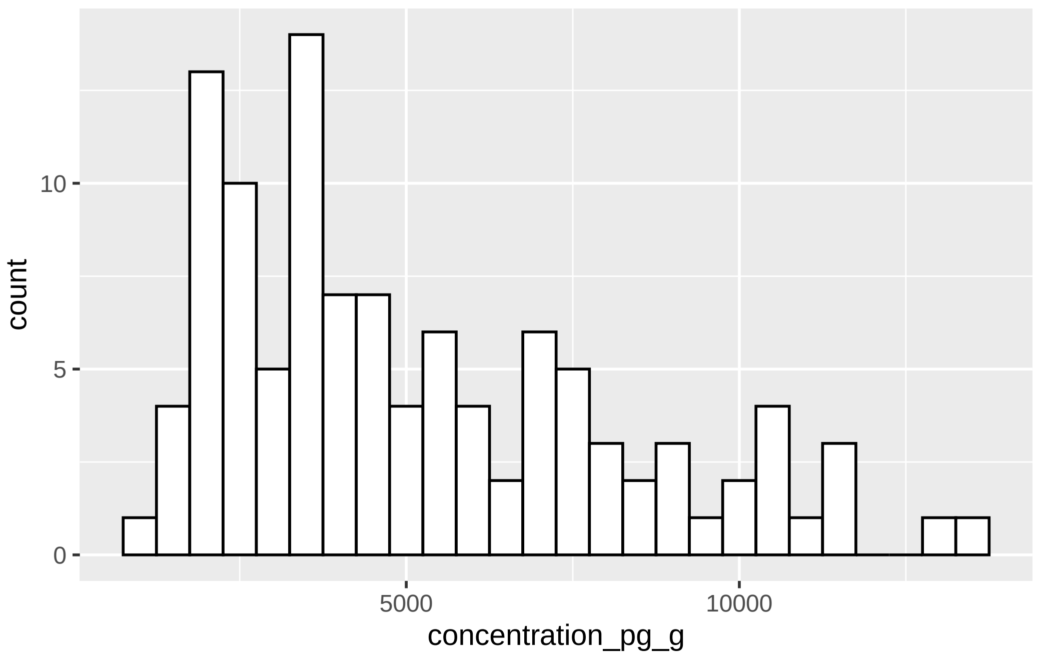
Each row is an individual sample of pika feces.

```
head(nwt_pikas)
```

```
# A tibble: 6 x 8
  date       site      station utm_easting utm_northing sex   concentration_pg_g
  <date>     <fct>     <fct>         <dbl>        <dbl> <fct>              <dbl>
1 2018-06-08 Cable Ga~ Cable ~      451373      4432963 male              11563.
2 2018-06-08 Cable Ga~ Cable ~      451411      4432985 male              10629.
3 2018-06-08 Cable Ga~ Cable ~      451462      4432991 male              10924.
4 2018-06-13 West Kno~ West K~      449317      4434093 male              10414.
5 2018-06-13 West Kno~ West K~      449342      4434141 male              13531.
6 2018-06-13 West Kno~ West K~      449323      4434273 <NA>               7799.
# i 1 more variable: elev_m <dbl>
```

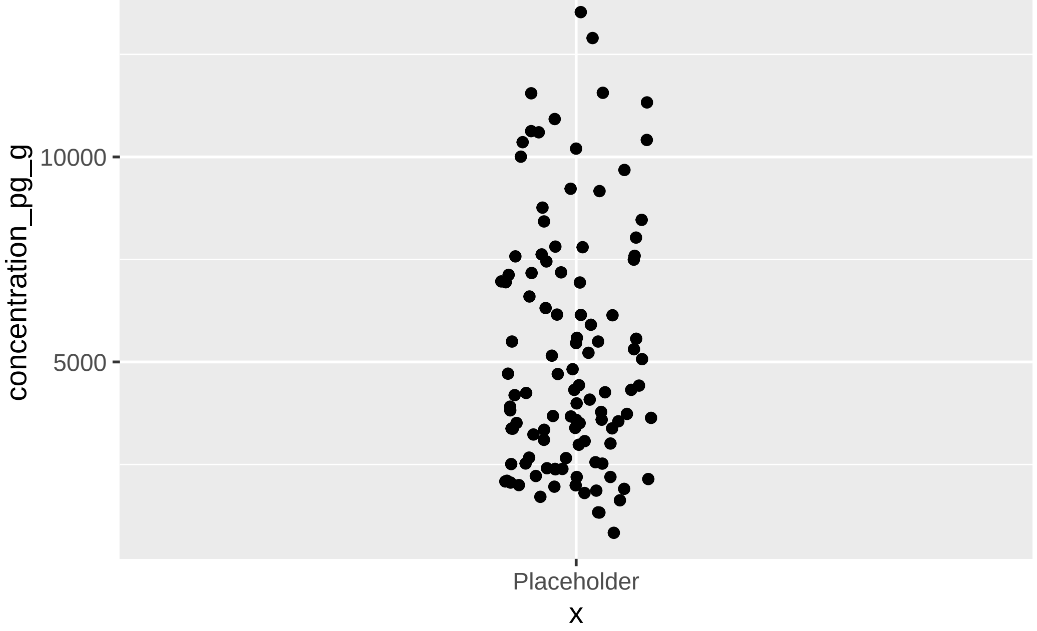
Categorize stress

```
# 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")
```



Wide distribution of stress but more on the lower end with a smaller number of individuals on the higher end.

```
# 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)
```



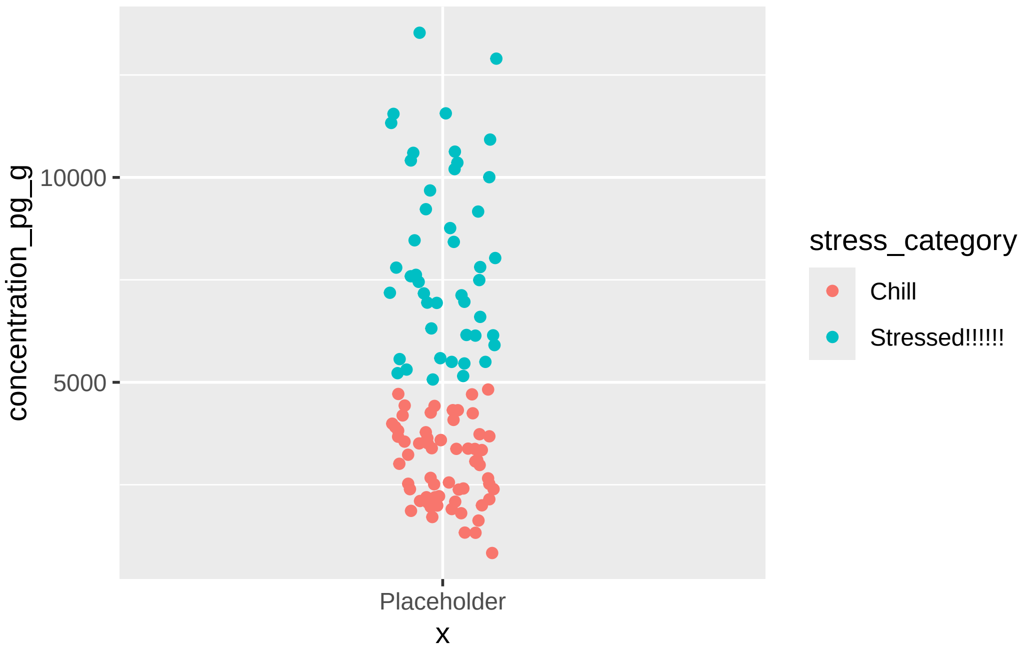
```
nwt_pikas_categ <- nwt_pikas %>%
  # Call the new column stress_category
  mutate(stress_category = case_when(
    # When the value is > 5000, make the new column's value "Stressed!!!!"
    concentration_pg_g > 5000 ~ "Stressed!!!!!!",
    # Otherwise, make the new column's value "Chill"
    .default = "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: Remake the scatterplot, but color the points by the new stress category

```
# 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)
```



Another category for our pikas– time of year

```
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>
```

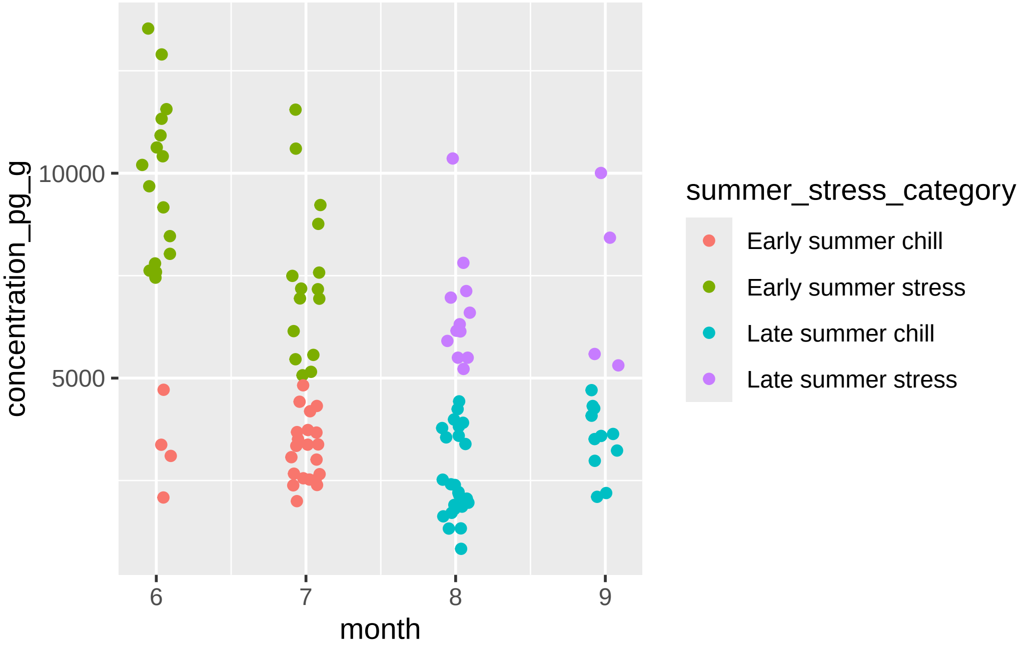
Creating stress categories for summer

```
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. DIY a for loop and an if else statement / case when

Q2.1 What dataset are you using?

```
library(palmerpenguins)
```

```
Attaching package: 'palmerpenguins'

The following objects are masked from 'package:datasets':

    penguins, penguins_raw
```

We will be using the penguins dataset.

Q2.2 Write a couple sentences describing what you want to do with the for loop

```
head(penguins)
```

```
# A tibble: 6 x 8
  species island    bill_length_mm bill_depth_mm flipper_length_mm body_mass_g
  <fct>   <fct>              <dbl>         <dbl>             <int>       <int>
1 Adelie  Torgersen           39.1          18.7               181        3750
2 Adelie  Torgersen           39.5          17.4               186        3800
3 Adelie  Torgersen           40.3          18                 195        3250
4 Adelie  Torgersen           NA            NA                  NA          NA
5 Adelie  Torgersen           36.7          19.3               193        3450
6 Adelie  Torgersen           39.3          20.6               190        3650
# i 2 more variables: sex <fct>, year <int>
```

Average for each bill length, bill depth, flipper length and body mass per species.

Q2.3 Apply the for loop to this dataset

```
#This loops through the columns 3 through 6
for (i in 3:6) {
  #This prints Mean for column and the column name for whatever "i" column you are looping th
  print(paste("Mean for column", colnames(penguins)[i]))
  #This calculates and prints the average for that column ignoring NAs
  print(mean(penguins[[i]], na.rm = TRUE))
}
```

```
[1] "Mean for column bill_length_mm"
[1] 43.92193
[1] "Mean for column bill_depth_mm"
[1] 17.15117
[1] "Mean for column flipper_length_mm"
[1] 200.9152
[1] "Mean for column body_mass_g"
[1] 4201.754
```

Q2.4 Write a couple sentences describing what you want to do with the if else/case_when

```
view(penguins)
```

We are interested in penguin distribution around the varying island. We will perform an ifelse to gather the number of individuals present from per island and categorize for level of threat.

```
#Creating object called island_counts
island_counts <- penguins %>%
  #grouping by island and year
  group_by(island, year) %>%
  #number of observations will equal penguin count
  summarise(penguin_count = n())
```

```
`summarise()` has grouped output by 'island'. You can override using the
`.groups` argument.
```

```
#create island status object
island_status <- island_counts %>%
  #mutate via if else to create a new column called status
  mutate(status = ifelse(penguin_count < 50, "Threatened", "Stable"))

print(island_status)
```

```
# A tibble: 9 x 4
# Groups:   island [3]
  island     year penguin_count status
  <fct>     <int>         <int> <chr>
1 Biscoe     2007            44 Threatened
2 Biscoe     2008            64 Stable
3 Biscoe     2009            60 Stable
4 Dream      2007            46 Threatened
5 Dream      2008            34 Threatened
6 Dream      2009            44 Threatened
7 Torgersen  2007            20 Threatened
8 Torgersen  2008            16 Threatened
9 Torgersen  2009            16 Threatened
```

We estimated 50 as a stable population, however, we don't know this metric and this could vary depending on the resources and carrying capacity of the specific island. But this gives a good baseline to work off of. Here we see how stability and population size have changed over time.