Working in the Tidyverse

Trever Yoder

Task 1

Question A

We cannot use `read_csv` because it can only read in comma and tab separated values (ours is ";")

```
#first we need to read in the tidyverse package
library(tidyverse)
```

```
Warning: package 'ggplot2' was built under R version 4.4.2
```

```
Warning: package 'lubridate' was built under R version 4.4.3
```

```
-- 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   3.5.1     v tibble    3.2.1
v lubridate 1.9.4     v tidyr     1.3.1
v purrr     1.0.2
-- 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
```

```
#Let's read it in with read_csv2 since it can handle ; delimited files
data <- read_csv2("data/data.txt", col_names = TRUE)
```

```
i Using "','" as decimal and "'.'" as grouping mark. Use `read_delim()` for more control.
Rows: 2 Columns: 3-- Column specification --------------------------------------------------
Delimiter: ";"
dbl (3): x, y, z
i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
#Print the output
data
```

```
# A tibble: 2 x 3
      x     y     z
  <dbl> <dbl> <dbl>
1     1     2     3
2     5     3     8
```

Question B

```
#Read in the data
data2 <- read_delim("data/data2.txt", col_names = TRUE, delim = "6", col_types = "fdc")

#print the data
data2
```

```
# A tibble: 3 x 3
  x         y z
  <fct> <dbl> <chr>
1 1         2 3
2 5         3 8
3 7         4 2
```

Task 2

Question A

```
#Read in the trailblazer file
trailblazer <- read.csv("data/trailblazer.csv")
```

```r
#Glimpse the data
glimpse(trailblazer)
```

```
Rows: 9
Columns: 11
$ Player      <chr> "Damian Lillard", "CJ McCollum", "Norman Powell", "Robert ~
$ Game1_Home  <int> 20, 24, 14, 8, 20, 5, 11, 2, 7
$ Game2_Home  <int> 19, 28, 16, 6, 9, 5, 18, 8, 11
$ Game3_Away  <int> 12, 20, NA, 0, 4, 8, 12, 5, 5
$ Game4_Home  <int> 20, 25, NA, 3, 17, 10, 17, 8, 9
$ Game5_Home  <int> 25, 14, 12, 9, 14, 9, 5, 3, 8
$ Game6_Away  <int> 14, 25, 14, 6, 13, 6, 19, 8, 8
$ Game7_Away  <int> 20, 20, 22, 0, 7, 0, 17, 7, 4
$ Game8_Away  <int> 26, 21, 23, 6, 6, 7, 15, 0, 0
$ Game9_Home  <int> 4, 27, 25, 19, 10, 0, 16, 2, 7
$ Game10_Home <int> 25, 7, 13, 12, 15, 6, 10, 4, 8
```

Question B

```r
#Pivot the data
trailblazer_longer <- trailblazer |>
  pivot_longer(cols = 2:11,
               names_to = "Location",
               values_to = "Points") |>
  separate_wider_delim(cols = "Location",
                       delim = "_",
                       names = c("Game", "Location"))

#Print first 5 rows
trailblazer_longer |>
  slice(1:5)
```

```
# A tibble: 5 x 4
  Player         Game  Location Points
  <chr>          <chr> <chr>     <int>
1 Damian Lillard Game1 Home         20
2 Damian Lillard Game2 Home         19
3 Damian Lillard Game3 Away         12
4 Damian Lillard Game4 Home         20
5 Damian Lillard Game5 Home         25
```

Question C

On average, Jusuf Nurkic scored more points at home than away during the first 10 games of the season. Below is the code that lead us to this answer!

```
trailblazer_wider <- trailblazer_longer |>

  #Create columns for home and away
  pivot_wider(
  names_from = "Location",
  values_from = "Points") |>

  #group so that the mean is calculated per player
  group_by(Player) |>

  #find means and difference for home vs away
  mutate(mean_home = mean(Home, na.rm = TRUE),
         mean_away = mean(Away, na.rm = TRUE),
         mean_diff = mean_home - mean_away) |>

  #arrange in descending order (ungroup first)
  ungroup() |>
  arrange(desc(mean_diff))

#Print the first row
trailblazer_wider |>
  slice(1:1)
```

```
# A tibble: 1 x 7
  Player       Game   Home  Away mean_home mean_away mean_diff
  <chr>        <chr> <int> <int>     <dbl>     <dbl>     <dbl>
1 Jusuf Nurkic Game1    20    NA      14.2       7.5      6.67
```

Task 3

Question A

1. Meaning of `<NULL>`: There aren't any of this species on these islands, so there are no values so its an empty cell, or "undefined".
2. Meaning of `<dbl [52]>`: There is a vector with 52 numeric (specifically double) elements
3. Meaning of `<list>`: These variables are stored as lists

```r
#read in the palmerpenguins package
library(palmerpenguins)
```

```
Warning: package 'palmerpenguins' was built under R version 4.4.3
```

```r
#run the code provided by colleques
penguins1 <- penguins |>
select(species, island, bill_length_mm) |>
pivot_wider(
names_from = island, values_from = bill_length_mm
)
```

```
Warning: Values from `bill_length_mm` are not uniquely identified; output will contain
list-cols.
* Use `values_fn = list` to suppress this warning.
* Use `values_fn = {summary_fun}` to summarise duplicates.
* Use the following dplyr code to identify duplicates.
  {data} |>
  dplyr::summarise(n = dplyr::n(), .by = c(species, island)) |>
  dplyr::filter(n > 1L)
```

```r
view(penguins1)
```

Question B

```r
#create the desired table
penguins2 <- penguins |>
    group_by(species) |>
  summarise(
    Biscoe = as.double(sum(island %in% "Biscoe", na.rm = TRUE)),
    Dream = as.double(sum(island == "Dream", na.rm = TRUE)), #used == vs %n% for fun
    Torgersen = as.double(sum(island == "Torgersen", na.rm = TRUE)),
    .groups = "keep" #kept the grouping as shown in the desired table
)

#print the table
penguins2
```

```
# A tibble: 3 x 4
# Groups:   species [3]
  species   Biscoe Dream Torgersen
  <fct>      <dbl> <dbl>     <dbl>
1 Adelie        44    56        52
2 Chinstrap      0    68         0
3 Gentoo       124     0         0
```

Task 4

Replacing NA Values

```
penguins_full <- penguins |>
  mutate(bill_length_mm = case_when(
    species == "Adelie" & is.na(bill_length_mm) ~ 26
    ,species == "Gentoo" & is.na(bill_length_mm) ~ 30
    ,TRUE ~ bill_length_mm
  ))


#print first 10 rows of the table in accending order
penguins_full |>
  arrange(bill_length_mm) |>
  slice(1:10)
```

```
# A tibble: 10 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           26            NA                NA          NA
 2 Gentoo  Biscoe              30            NA                NA          NA
 3 Adelie  Dream               32.1          15.5              188        3050
 4 Adelie  Dream               33.1          16.1              178        2900
 5 Adelie  Torgersen           33.5          19                190        3600
 6 Adelie  Dream               34            17.1              185        3400
 7 Adelie  Torgersen           34.1          18.1              193        3475
 8 Adelie  Torgersen           34.4          18.4              184        3325
 9 Adelie  Biscoe              34.5          18.1              187        2900
10 Adelie  Torgersen           34.6          21.1              198        4400
# i 2 more variables: sex <fct>, year <int>
```