Homework 3: Programming with Tidyverse and Base R

Keshav Ramesh

```
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   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     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
```

```
library(palmerpenguins)
```

Task 1

Question 1

```
## import data
read_csv("Data/data.txt")
```

```
Rows: 2 Columns: 1
-- Column specification --------------------------------------------------------
Delimiter: ","
chr (1): 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.
```

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

The reason that we cannot use read_csv is that the data is delimited by something other than a comma or a tab.

```
#import data and assign

data = read_csv2("Data/data.txt")
```

```
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.
```

```
data
```

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

Question 2

```r
data2 = read_delim("Data/data2.txt",
  delim = "6",
  col_types = cols(
    x = col_factor(),
    y = col_double(),
    z = col_character()
  )

)


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 1

```
trailblazer <- read_csv("Data/trailblazer.csv")
```

```
Rows: 9 Columns: 11
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (1): Player
dbl (10): Game1_Home, Game2_Home, Game3_Away, Game4_Home, Game5_Home, Game6_...

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.
```

```
glimpse(trailblazer)
```

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

```
## checking data
head(trailblazer)
```

```
# A tibble: 6 x 11
  Player       Game1_Home Game2_Home Game3_Away Game4_Home Game5_Home Game6_Away
  <chr>             <dbl>      <dbl>      <dbl>      <dbl>      <dbl>      <dbl>
1 Damian Lill~         20         19         12         20         25         14
2 CJ McCollum          24         28         20         25         14         25
```

```
3 Norman Powe~         14         16         NA         NA         12         14
4 Robert Covi~          8          6          0          3          9          6
5 Jusuf Nurkic         20          9          4         17         14         13
6 Cody Zeller           5          5          8         10          9          6
# i 4 more variables: Game7_Away <dbl>, Game8_Away <dbl>, Game9_Home <dbl>,
#   Game10_Home <dbl>
```

Question 2

```
trailblazer_longer = trailblazer %>%
  pivot_longer(!Player, names_to = "Game_Location", values_to = "Points") %>% # make columns
  separate(Game_Location, into = c("Game", "Location"), sep = "_") ## separate columns

head(trailblazer_longer, 5)
```

```
# A tibble: 5 x 4
  Player         Game  Location Points
  <chr>          <chr> <chr>     <dbl>
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 3

```
trailblazer_points = trailblazer_longer %>%
  pivot_wider(names_from = Location,
              values_from = Points) %>% # make columns
  group_by(Player) %>% # group for summarize function
  summarise(
    mean_home = mean(Home, na.rm = TRUE),
    mean_away = mean(Away, na.rm = TRUE),
    difference = mean_home - mean_away
  ) %>%
  arrange(desc(difference)) # arrange

trailblazer_points
```

```
# A tibble: 9 x 4
  Player           mean_home mean_away difference
  <chr>                <dbl>     <dbl>      <dbl>
1 Jusuf Nurkic         14.2       7.5       6.67
2 Robert Covington      9.5       3         6.5
3 Nassir Little         8.33      4.25      4.08
4 Damian Lillard       18.8      18         0.833
5 Cody Zeller           5.83      5.25      0.583
6 Larry Nance Jr        4.5       5        -0.5
7 CJ McCollum          20.8      21.5      -0.667
8 Anfernee Simons      12.8      15.8      -2.92
9 Norman Powell        16        19.7      -3.67
```

Task 3

Question 1

```
## checking question

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)
```

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

“” = there is no data for that combination of species and island

“<dbl [52]>” = there are 52 values with a class of “double”

“” = indicates that the column contains a list-column instead of a value

Question 2

```
penguins %>%
  count(species, island) %>% # count combinations
  pivot_wider(
    names_from = island, values_from = n, values_fill = 0) ## pivot to match
```

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

Task 4

Question 1

```
## Checking original dataset

penguins
```

```
# A tibble: 344 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
 7 Adelie  Torgersen           38.9          17.8               181        3625
 8 Adelie  Torgersen           39.2          19.6               195        4675
 9 Adelie  Torgersen           34.1          18.1               193        3475
10 Adelie  Torgersen           42            20.2               190        4250
# i 334 more rows
# i 2 more variables: sex <fct>, year <int>
```

```
penguins_fixed = penguins %>%
  mutate( ## column access
    bill_length_mm = case_when( ## ifelse
      is.na(bill_length_mm) & species == "Adelie" ~ 26,
      is.na(bill_length_mm) & species == "Gentoo" ~ 30,
      TRUE ~ bill_length_mm ## else case
    )
  ) %>%
  arrange(bill_length_mm)

head(penguins_fixed, 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>
```