AE 06: Joining country populations with continents

Goal

Our ultimate goal in this application exercise is to create a bar plot of total populations of continents, where the input data are:

1. Countries and populations
2. Countries and continents

```
library(tidyverse) # for data wrangling and visualization
library(scales)    # for pretty axis breaks
```

Data

Countries and populations

These data come from The World Bank and reflect population counts as of 2022.

```
population <- read_csv("https://sta199-s24.github.io/data/world-pop-2022.csv")
```

Let's take a look at the data.

```
population
```

```
# A tibble: 217 x 3
   country              year population
   <chr>               <dbl>      <dbl>
 1 Afghanistan          2022    41129.
 2 Albania              2022     2778.
```

```
 3 Algeria              2022    44903.
 4 American Samoa       2022       44.3
 5 Andorra              2022       79.8
 6 Angola               2022    35589.
 7 Antigua and Barbuda  2022       93.8
 8 Argentina            2022    46235.
 9 Armenia              2022     2780.
10 Aruba                2022      106.
# i 207 more rows
```

Continents

These data come from Our World in Data.

```
continents <- read_csv("https://sta199-s24.github.io/data/continents.csv")
```

Let's take a look at the data.

```
continents
```

```
# A tibble: 285 x 4
   entity                code      year continent
   <chr>                 <chr>    <dbl> <chr>
 1 Abkhazia              OWID_ABK  2015 Asia
 2 Afghanistan           AFG       2015 Asia
 3 Akrotiri and Dhekelia OWID_AKD  2015 Asia
 4 Aland Islands         ALA       2015 Europe
 5 Albania               ALB       2015 Europe
 6 Algeria               DZA       2015 Africa
 7 American Samoa        ASM       2015 Oceania
 8 Andorra               AND       2015 Europe
 9 Angola                AGO       2015 Africa
10 Anguilla              AIA       2015 North America
# i 275 more rows
```

Exercises

- **Think out loud:**
  - Which variable(s) will we use to join the `population` and `continents` data frames?

Add response here.

  - We want to create a new data frame that keeps all rows and columns from `population` and brings in the corresponding information from `continents`. Which join function should we use?

Add response here.

- **Demo:** Join the two data frames and name assign the joined data frame to a new data frame `population_continents`.

```
# add code here
```

- **Demo:** Take a look at the newly created `population_continent` data frame. There are some countries that were not in `continents`. First, identify which countries these are (they will have `NA` values for `continent`).

```
# add code here
```

- **Demo:** All of these countries are actually in the `continents` data frame, but under different names. So, let's clean that data first by updating the country names in the `population` data frame in a way they will match the `continents` data frame, and then joining them, using a `case_when()` statement in `mutate()`. At the end, check that all countries now have continent information.

```
# add code here
```

- **Think out loud:** Which continent do you think has the highest population? Which do you think has the second highest? The lowest?

Add your response here.

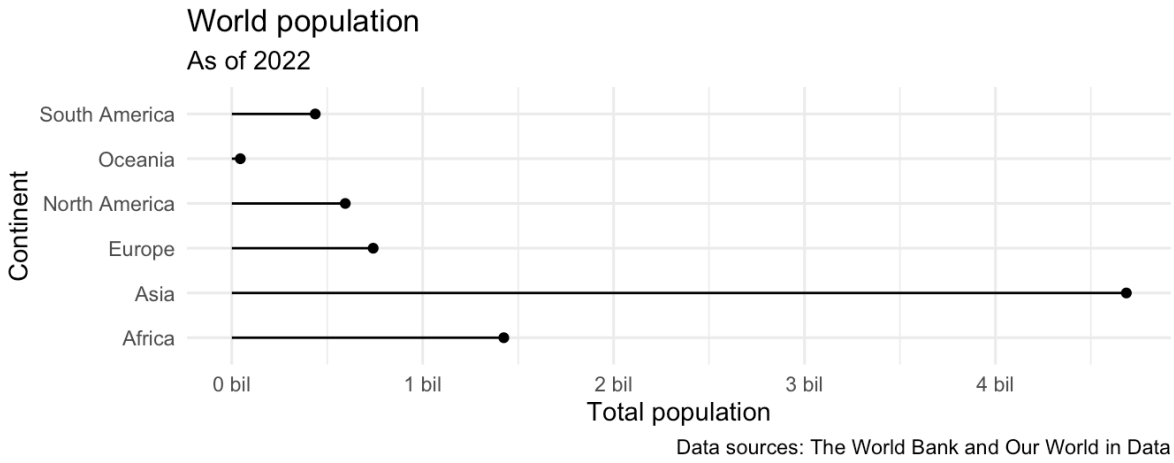
- **Demo:** Create a new data frame called `population_summary` that contains a row for each continent and a column for the total population for that continent, in descending order of population. Note that the function for calculating totals in R is `sum()`.

```
# add code here
```

- **Your turn:** Make a bar plot with total population on the y-axis and continent on the x-axis, where the height of each bar represents the total population in that continent.

```
# add code here
```

- **Your turn:** Recreate the following plot, which is commonly referred to as a **lollipop plot**. Hint: Start with the points, then try adding the `segment`s, then add axis labels and `caption`, and finally, as a stretch goal, update the x scale (which will require a function we haven't introduced in lectures or labs yet!).




```
# add code here
```

- **Think out loud:** What additional improvements would you like to make to this plot.

Add your response here.