Coding Challenge 6

Theresa Quintana

2025-03-27

You can find the corresponding documents to this assignment on my GitHub: Theresa's PLPA 6820 Github Access

#Question 1 (2 pts). Regarding reproducibility, what is the main point of writing your own functions and iterations? Writing your own helps reduce manual steps (e.g., copy/paste) and therefore reduces errors by helping you automate processes that others later (or future you!!) can run easier for reproducibility.

#Question 2 (2 pts). In your own words, describe how to write a function and a for loop in R and how they work. Give me specifics like syntax, where to write code, and how the results are returned. Writing a function - Start with naming the variable (what the function will go into). Next, you use the keyword function and define the inputs that will go into the () (e.g., color <- function(blue, green, etc)). After this, you use {} to write the code of the function within. The function is then a set of instructions you can give input to and it will run calculations and give you the output automatically.

Writing a "for" loop - "For" loops will help us repeat a function task over and over for all the points in our data we are interested in. To write it, we use for (i in a:z) where i is the changing variable and a:z is the data range.We then use the {} to insert the function to run through the data range.

You write these in r code chunks. Functions will save to an object in the environment and results can be returned/print to the console or you can save/store them. For loops, you have to save/store the results in a data frame or vector or print to the console (the loop itself will just run otherwise).

#Question 3 (2 pts). Read in the Cities.csv file from Canvas using a relative file path.

```
cities <- read.csv("Cities.csv", na.strings = "na")
```

#Question 4 (6 pts). Write a function to calculate the distance between two pairs of coordinates based on the Haversine formula (see below). The input into the function should be lat1, lon1, lat2, and lon2. The function should return the object distance_km. All the code below needs to go into the function.

```
distanceA_to_B <- function(lat1, lon1, lat2, lon2){
  #covert to radians
  rad.lat1 <- lat1 * pi/180
  rad.lon1 <- lon1 * pi/180
  rad.lat2 <- lat2 * pi/180
  rad.lon2 <- lon2 * pi/180
  #Haversine formula
  delta_lat <- rad.lat2 - rad.lat1
  delta_lon <- rad.lon2 - rad.lon1
  a <- sin(delta_lat / 2)^2 + cos(rad.lat1) * cos(rad.lat2) * sin(delta_lon / 2)^2
  c <- 2 * asin(sqrt(a))
  #in km
  earth_radius <- 6378137
  #distance calculation
```

```
  distance_km <- (earth_radius * c)/1000
  return(distance_km)
}
```

#Question 5 (5 pts). Using your function, compute the distance between Auburn, AL and New York City. Subset/filter the Cities.csv data to include only the latitude and longitude values you need and input as input to your function. The output of your function should be 1367.854 km

```
#subset cities dataframe so that only Auburn and New York are left with there respective long/lat value
cities_2 <- subset(cities,
                   (city == "Auburn" & state_name == "Alabama") |
                     (city == "New York" & state_name == "New York"),
                   select = c(city, state_name, lat, long))

#define lon/lat inputs
lat1 <- cities_2$lat[cities_2$city == "Auburn"]
lon1 <- cities_2$long[cities_2$city == "Auburn"]
lat2 <- cities_2$lat[cities_2$city == "New York"]
lon2 <- cities_2$long[cities_2$city == "New York"]

#find the distance between Auburn and New York using the function
Auburn_to_NewYork <- distanceA_to_B(lat1, lon1, lat2, lon2)
Auburn_to_NewYork #in km
```

```
## [1] 1367.854
```

#Question 6 (6 pts). Now, use your function within a for loop to calculate the distance between all other cities in the data. The output of the first 9 iterations is shown below.

```
#define auburn lon/lat inputs (this was done above but just repeating)
#(could also put this in the loop (does it matter where?))
lat1 <- cities_2$lat[cities_2$city == "Auburn"] #could also take from cities dataframe
lon1 <- cities_2$long[cities_2$city == "Auburn"]


for (i in 1:nrow(cities)){
  lat2 <- cities$lat[i] #changing lat2 from previous chunk (set to NY only)
  lon2 <- cities$long[i] #changing lon2 from previous chunk (set to NY only)
  result <- distanceA_to_B(lat1, lon1, lat2, lon2)
  print(result)
}
```

```
## [1] 1367.854
## [1] 3051.838
## [1] 1045.521
## [1] 916.4138
## [1] 993.0298
## [1] 1056.022
## [1] 1239.973
## [1] 162.5121
## [1] 1036.99
## [1] 1665.699
```

```
## [1] 2476.255
## [1] 1108.229
## [1] 3507.959
## [1] 3388.366
## [1] 2951.382
## [1] 1530.2
## [1] 591.1181
## [1] 1363.207
## [1] 1909.79
## [1] 1380.138
## [1] 2961.12
## [1] 2752.814
## [1] 1092.259
## [1] 796.7541
## [1] 3479.538
## [1] 1290.549
## [1] 3301.992
## [1] 1191.666
## [1] 608.2035
## [1] 2504.631
## [1] 3337.278
## [1] 800.1452
## [1] 1001.088
## [1] 732.5906
## [1] 1371.163
## [1] 1091.897
## [1] 1043.273
## [1] 851.3423
## [1] 1382.372
## [1] 0
```

#Question Bonus - if you can have the output of each iteration append a new row to a dataframe, generating a new column of data. In other words, the loop should create a dataframe with three columns called city1, city2, and distance_km, as shown below. The first six rows of the dataframe are shown below.

```
cities_distances <- NULL #empty df
for (i in 1:nrow(cities)) { #run the loop, lon/lat1 already defined in chunks above
   lat2 <- cities$lat[i]
   lon2 <- cities$long[i]
   distance <- distanceA_to_B(lat1, lon1, lat2, lon2)

   result_i <- data.frame(
    cityA = "Auburn", #columns for the df, City A (auburn), B(rotating cities), distance between
    cityB = cities$city[i],
    distance_km = distance
  )
  cities_distances <- rbind(cities_distances, result_i)
}
```

Question 7 (2 pts). Commit and push a gfm .md file to GitHub inside a directory called Coding Challenge 6. Provide me a link to your github written as a clickable link in your .pdf or .docx