CodingChallenge6_Iterations&Functions_SK

Shakiba Kazemian

2025-03-27

Q1: Regarding reproducibility, what is the main point of writing your own functions and iterations?

Writing our own functions and iterations is essential for reproducibility because it allows us to automate repetitive tasks in a consistent and error-free manner. Functions make our code modular, cleaner, and easier to understand, which is especially important when working with complex analyses or large datasets. Iterations, such as `for` loops or `map()` functions, ensure that the same logic is applied uniformly across multiple inputs, reducing the risk of human error from copy-pasting. Together, functions and iterations improve the transparency, scalability, and repeatability of our code—key components of reproducible research that allow others (or our future self!!) to re-run and verify results reliably.

Q2: 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.

In R, a **function** is written using the `function()` keyword and is used to perform a specific task on input values. You define a function using the format: `function_name <- function(input1, input2) { code; return(result) }`. The code inside the curly brackets runs when the function is called, and the `return()` statement sends back the result or output. A **for loop** is used to repeat code multiple times. It follows the format: `for (i in 1:5) { code }`, where `i` takes on each value in the sequence, and the code inside the loop runs each time. Both functions and loops help avoid repeating code and make your analysis cleaner and more efficient.

Q3: Reading the data using a relative file path

```
cities <- read.csv("Cities.csv")
```

Q4: wrap all that code into a function called haversine_distance() that takes in four arguments: lat1, lon1, lat2, and lon2, and returns distance_km.

```
haversine_distance <- function(lat1, lon1, lat2, lon2) {
  # Convert degrees 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))

  # Earth's radius in meters
  earth_radius <- 6378137

  # Calculate distance in kilometers
  distance_km <- (earth_radius * c) / 1000

  return(distance_km)
}
```

Q5: Filter to get Auburn, AL and New York City

a. Extracting the latitude and longitude for both cities

```
auburn <- cities[cities$city == "Auburn" & cities$state_id == "AL", ]
nyc <- cities[cities$city == "New York" & cities$state_id == "NY", ]
```

b. Using the function to calculate the distance

```
# Define the Haversine function
haversine_distance <- function(lat1, lon1, lat2, lon2) {
  rad.lat1 <- lat1 * pi / 180
  rad.lon1 <- lon1 * pi / 180
  rad.lat2 <- lat2 * pi / 180
  rad.lon2 <- lon2 * pi / 180

  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))
  earth_radius <- 6378137
  distance_km <- (earth_radius * c) / 1000
  return(distance_km)
}
```

Using the latitude and longitude values from the two cities:

```
distance <- haversine_distance(
  lat1 = auburn$lat,
  lon1 = auburn$long,
  lat2 = nyc$lat,
  lon2 = nyc$long
)

distance  # Should output: 1367.854
```

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

Q6: using the haversine_distance() function inside a for loop to:

- Compare every other city to Auburn, AL
- Calculate the distance
- Append the result into a tidy data.frame with columns: City1, City2, and Distance_km

```
# Filter Auburn info
auburn <- cities[cities$city == "Auburn" & cities$state_id == "AL", ]

# Create an empty dataframe to hold results
distance_df <- data.frame(City1 = character(),
                          City2 = character(),
                          Distance_km = numeric(),
                          stringsAsFactors = FALSE)

# Loop through each city in the dataset
for (i in 1:nrow(cities)) {
  current_city <- cities[i, ]

  # Skip if the current city *is* Auburn
  if (current_city$city == "Auburn" & current_city$state_id == "AL") {
    next
  }

  # Calculate the distance
  distance <- haversine_distance(
    lat1 = current_city$lat,
    lon1 = current_city$long,
    lat2 = auburn$lat,
    lon2 = auburn$long
  )

  # Append the result to the dataframe
  distance_df <- rbind(distance_df, data.frame(
    City1 = current_city$city,
    City2 = "Auburn",
    Distance_km = distance
  ))
}
```

view the output:

```
# View first few rows
head(distance_df)
```

```
##         City1  City2 Distance_km
## 1    New York Auburn   1367.8540
## 2 Los Angeles Auburn   3051.8382
## 3     Chicago Auburn   1045.5213
## 4       Miami Auburn    916.4138
## 5     Houston Auburn    993.0298
## 6      Dallas Auburn   1056.0217
```

Link to my GitHub

Click here to view my submission on GitHub