Codingchallenge6

Temitope Folorunso & Nneka Iduu

2025-03-27

Contents

Loops and iterations – 25 pts **1**

Question 1 1

Question 2 1

Question 3 2

Question 4 2

Question 5 3

Question 6 3

link to the github 4

Loops and iterations – 25 pts

This assignment will help with practicing on writing and executing for loops and writing functions. It will also involve some more practice with GitHub.

Question 1

1. 2 pts. Regarding reproducibility, what is the main point of writing your own functions and iterations?

The main purpose of writing our own functions and iterations is to enable automation, which prevents mistakes from copying and pasting. This allows us to repeat processes on different datasets, making the code more efficient and consistent, thus ensuring reproducible data management.

Question 2

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.

How to write a function in R;

- Name the function and assign it using the backward arrow (<-).
- Use the function() keyword to declare the function.
- Inside the parentheses (), specify the input variables (separated by commas if there are multiple).
- Open curly brackets {} and write the code the function will execute.
- Use the return() statement to return the result from the function.

Syntax for function;

my_function <-function(x, y) { result <- x + y return(result) } result #call result if needed – optional

How it works

The function with name, my_function is created and given inputs x and y. Inside the curly brackets {}, it adds the inputs together (x + y) as the specified operation and then returns the result of that operation.

How to write a for loop in R;

- Define the loop using the 'for' keyword.
- Specify the loop variable and the sequence it will iterate over (i in 1:n)
- Use curly brackets {} to define the code that will be executed in each iteration.

Syntax for for loop;

for (i in 1:5) {
print(i)
}

How it works; The for loop iterates over a sequence (in this case, numbers 1 to 5). On each iteration, it assigns the current number to i and prints it. The loop runs 5 times, printing the values 1 through 5.

This dataset contains the population and coordinates (latitude and longitude) of the 40 most populous cities in the US, along with Auburn, AL. Your task is to create a function that calculates the distance between Auburn and each other city using the Haversine formula. To do this, you'll write a for loop that goes through each city in the dataset and computes the distance from Auburn. Detailed steps are provided below.

Question 3

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

Question 4

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.

```
DistanCord <- 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

# 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 kilometers
earth_radius <- 6378137

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

return(distance_km)
}
```

Question 5

Using your function, compute the distance between Auburn, AL and New York City a. Subset/filter the Cities.csv data to include only the latitude and longitude values you need and input as input to your function.

```
## Subset the city first
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
lat1 <-data$lat[data$city =="Auburn"]
lon1 <- data$long[data$city == "Auburn"]
lat2 <-data$lat[data$city =="New York"]
lon2 <- data$long[data$city == "New York"]

DistanCord(lat1,lon1,lat2,lon2)
```

```
## [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.

```
  # Initialize an empty vector
# Initialize a new vector to store distances
distances <- numeric(ncol(data))

# Loop through each city and compute distance to Auburn
for (i in 1:nrow(data)) {
  lat2 <- data$lat[i]
```

```
  lon2 <- data$long[i]
  distances[i] <- DistanCord(lat1, lon1, lat2, lon2)
}


# Add the result to your cities data frame
data$Distance_km <- distances
data$City2 <- "Auburn"
Distance.all <- data.frame(distances)

Data1<- data[, c("city","City2", "Distance_km")]
Data1 <- rename(Data1, City1 = city)
head(Data1)
```

```
##         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 the github

Link to my code_challenge6