CC-6

Deekshya

2025-03-30

####1. 2 pts. Regarding reproducibility, what is the main point of writing your own functions and iterations? It helps reduce copy and paste errors for something we would like to do over and over again. It can also helpful for sanity checks through data simulation.

###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.

F_to_C <- function(fahrenheit_temp) { celsius <- (5 * (fahrenheit_temp - 32) / 9) # Formula to convert Fahrenheit to Celsius return(celsius) # Return the result }

#F_to_C : is a the name of function we are writing on our own to convert F to C

#function(){}, anything inside this is an argument or a input.

#(farenheit_temp) is an input #{celsius <- (5 * (fahrenheit_temp - 32) / 9) } is formula #return (celsius): ensures that when the function is called and #outputs the Celsius value.

###Now to make a loop of this

temps_F <- c(32, 68, 98.6, 212) # List of Fahrenheit temperatures

for (temp in temps_F) { print(F_to_C(temp)) # Calls the function and prints the Celsius value }

for(where my input is){do(something)}

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.

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

```
datum = read.csv("Cities.csv")
```

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.

convert 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 kilometers

earth_radius <- 6378137

Calculate the distance

distance_km <- (earth_radius * c)/1000

```
##function(input){formula}
distance <- function(lat1, lon1, lat2, lon2) {
  # Convert 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 kilometers
  earth_radius <- 6378137  # Radius in meters

  # Calculate the distance in kilometers
  distance_km <- (earth_radius * c) / 1000  # Convert meters to kilometers
  return(distance_km)
}
```

5. 5 pts. 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.

```
auburn <- subset(datum, city == "Auburn")
nyc <- subset(datum, city == "New York")

auburn_lat <- auburn$lat
auburn_lon <- auburn$long
nyc_lat <- nyc$lat
nyc_lon <- nyc$long

distance_to_nyc = distance(auburn_lat, auburn_lon, nyc_lat, nyc_lon)
```

b. The output of your function should be 1367.854 km

```
print(distance_to_nyc)
```

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

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. ## [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

```
# Create an empty vector to store the distances
distances <- c()

other_cities <- subset(datum, city != "Auburn")
# Loop through all the other cities
for (i in 1:nrow(other_cities)) {
  # Get the latitude and longitude of the current city
  city_lat <- other_cities$lat[i]
  city_lon <- other_cities$long[i]

  # Calculate the distance from Auburn to the current city
  distance_to_city <- distance(auburn_lat, auburn_lon, city_lat, city_lon)

  # Store the calculated distance in the 'distances' vector
  distances <- c(distances, distance_to_city)
}
```

```
print(distances[1:9])
```

```
## [1] 1367.8540 3051.8382 1045.5213  916.4138  993.0298 1056.0217 1239.9732
## [8]  162.5121 1036.9900
```

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