Coding_Challenge_6

Anet Filipova, Brynleigh Payne

2025-03-27

Question 1:

They are useful when you need to perform the same code on different data and you want to avoid copy and paste errors. Functions are useful to simplify your code, and make your data management as reproducible as possible.- Noel

Question 2:

To write a function we have to write the name of the function followed by () which specifies the input of the function, then using {} we describe the action or calculation the function will do. Finally, using return() the output is specifed. We can call a function by storing it into an object.

Example:

C_to_F <- function(celsius_temp){ farenheit <- (celsius_temp * (9/5)+32) return(farenheit) }

C_to_F(25)

To write a for loop we start with "for" followed by (i) which represents a character/value that will be iterated from a sequence we specify. After this we have to open {} where we actually specify the action for the iteration of "i". A for- loop works on the basis of code block repetition with multiple iterations and ending when specified.

Example:

for (i in 2:20){ print(i/2) }

Question 3:

```
# Load the data using a relative file path
cities <- read.csv("Cities.csv", na.strings = "na")
```

Question 4:

Creating a function to calculate the distance between two pairs of coordinates based on the Haversine formula

```r
KM <- function(lat1, lon1, lat2, lon2){ # function(lat1, lon1, lat2, lon2) will be the input of the fun
rad.lat1 <- lat1 * pi/180 # convert to radians
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:

Calculating the distance between Auburn and New York

```r
# a. Subsetting Auburn and New York
auburn <- subset(cities, city == "Auburn") # subsetting Auburn
nyc <- subset(cities, city == "New York") # subsetting New York

# b. Calculating the distance between the two cities
KM(lat1 = auburn$lat, lon1 = auburn$long, lat2 = nyc$lat, lon2 = nyc$long)
```

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

Question 6:

Creating a for loop to calculate the distance between Auburn and all other cities

```r
city_name <- NULL # Step 1. Set an R object to NULL
nm <- unique(cities$city) # Step 2. Using unique () function to put each city into an onject called nm
for (i in seq_along(nm)) { # Step 3. Setting the for loop with iteration values corresponding to each c
city_nm <- subset(cities, city == nm[i]) # Step 4. Subsetting i with each city name
d_i <- data.frame(KM(lat1 = auburn$lat, lat2 = city_nm$lat, lon1 = city_nm$long, lon2 = auburn$long), "
                  nm[i]) # Step 5. Save the result of your for loop into a dataframe each iteration
city_name <- rbind.data.frame(city_name, d_i) # Step 6. Append one row of the dataframe to the null obj
}
```

Question 7:

Link to GitHub