InClassChallenge6

Alex Berry

2025-03-27

Table of Contents

# QUESTION 1

The main point of writing your own functions is to avoid copy and paste errors when you need to perform the same code on different data. Functions allow you to simplify your code, allowing for better data management and reproducibility. Iterations are also helpful to reduce copy and paste errors when you need to repeat code multiple times, ultimately expediting repetitive work. What both functions and iterations accomplish is the reduction of copy and paste errors, and allows for consistent, and reusable code.

# QUESTION 2

# Writing a function  
function\_name <- function(factor1, factor2) {  
 result <- factor1 + factor2  
 return(result)  
}

# Example  
add\_numbers <- function(a, b) {  
 sum <- a + b  
 return(sum)  
}  
  
add\_numbers(12, 15)

## [1] 27

# Writing a For Loop  
for (i in 1:5) {  
 print(i)  
}

## [1] 1  
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5

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

## [1] 1  
## [1] 2  
## [1] 3  
## [1] 4  
## [1] 5

The code will be written in R Script or R Markdown. The output will be in the console.

# QUESTION 3

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

# QUESTION 4

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)  
}

# QUESTION 5

library(tidyverse)

## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──  
## ✔ dplyr 1.1.4 ✔ readr 2.1.5  
## ✔ forcats 1.0.0 ✔ stringr 1.5.1  
## ✔ ggplot2 3.5.1 ✔ tibble 3.2.1  
## ✔ lubridate 1.9.4 ✔ tidyr 1.3.1  
## ✔ purrr 1.0.2   
## ── Conflicts ────────────────────────────────────────── tidyverse\_conflicts() ──  
## ✖ dplyr::filter() masks stats::filter()  
## ✖ dplyr::lag() masks stats::lag()  
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors

auburn\_lat <- subset(cities.data, city == "Auburn" & state\_id == "AL") %>%  
 pull(lat)  
auburn\_long <- subset(cities.data, city == "Auburn" & state\_id == "AL") %>%  
 pull(long)  
nyc\_lat <- subset(cities.data, city == "New York" & state\_id == "NY") %>%  
 pull(lat)  
nyc\_long <- subset(cities.data, city == "New York" & state\_id == "NY") %>%  
 pull(long)  
  
haversine\_distance(auburn\_lat, auburn\_long, nyc\_lat, nyc\_long)

## [1] 1367.854

# QUESTION 6

head(cities.data, 10)

## city city\_ascii state\_id state\_name county\_fips  
## 1 New York New York NY New York 36081  
## 2 Los Angeles Los Angeles CA California 6037  
## 3 Chicago Chicago IL Illinois 17031  
## 4 Miami Miami FL Florida 12086  
## 5 Houston Houston TX Texas 48201  
## 6 Dallas Dallas TX Texas 48113  
## 7 Philadelphia Philadelphia PA Pennsylvania 42101  
## 8 Atlanta Atlanta GA Georgia 13121  
## 9 Washington Washington DC District of Columbia 11001  
## 10 Boston Boston MA Massachusetts 25025  
## county\_name lat long population density  
## 1 Queens 40.6943 -73.9249 18832416 10943.7  
## 2 Los Angeles 34.1141 -118.4068 11885717 3165.8  
## 3 Cook 41.8375 -87.6866 8489066 4590.3  
## 4 Miami-Dade 25.7840 -80.2101 6113982 4791.1  
## 5 Harris 29.7860 -95.3885 6046392 1386.5  
## 6 Dallas 32.7935 -96.7667 5843632 1477.2  
## 7 Philadelphia 40.0077 -75.1339 5696588 4547.5  
## 8 Fulton 33.7628 -84.4220 5211164 1425.3  
## 9 District of Columbia 38.9047 -77.0163 5146120 4245.2  
## 10 Suffolk 42.3188 -71.0852 4355184 5303.3

auburn\_lat <- subset(cities.data, city == "Auburn" & state\_id == "AL") %>%  
 pull(lat)  
auburn\_long <- subset(cities.data, city == "Auburn" & state\_id == "AL") %>%  
 pull(long)  
  
for (i in 1:nrow(cities.data)) {  
 lat\_city <- cities.data$lat[i]  
 lon\_city <- cities.data$long[i]  
 distances <- haversine\_distance(auburn\_lat, auburn\_long, lat\_city, lon\_city)  
 print(distances)  
}

## [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

cities.data$Distance\_from\_Auburn\_km <- distances  
  
head(cities.data, 10)

## city city\_ascii state\_id state\_name county\_fips  
## 1 New York New York NY New York 36081  
## 2 Los Angeles Los Angeles CA California 6037  
## 3 Chicago Chicago IL Illinois 17031  
## 4 Miami Miami FL Florida 12086  
## 5 Houston Houston TX Texas 48201  
## 6 Dallas Dallas TX Texas 48113  
## 7 Philadelphia Philadelphia PA Pennsylvania 42101  
## 8 Atlanta Atlanta GA Georgia 13121  
## 9 Washington Washington DC District of Columbia 11001  
## 10 Boston Boston MA Massachusetts 25025  
## county\_name lat long population density  
## 1 Queens 40.6943 -73.9249 18832416 10943.7  
## 2 Los Angeles 34.1141 -118.4068 11885717 3165.8  
## 3 Cook 41.8375 -87.6866 8489066 4590.3  
## 4 Miami-Dade 25.7840 -80.2101 6113982 4791.1  
## 5 Harris 29.7860 -95.3885 6046392 1386.5  
## 6 Dallas 32.7935 -96.7667 5843632 1477.2  
## 7 Philadelphia 40.0077 -75.1339 5696588 4547.5  
## 8 Fulton 33.7628 -84.4220 5211164 1425.3  
## 9 District of Columbia 38.9047 -77.0163 5146120 4245.2  
## 10 Suffolk 42.3188 -71.0852 4355184 5303.3  
## Distance\_from\_Auburn\_km  
## 1 0  
## 2 0  
## 3 0  
## 4 0  
## 5 0  
## 6 0  
## 7 0  
## 8 0  
## 9 0  
## 10 0

# Bonus  
auburn\_lat <- subset(cities.data, city == "Auburn" & state\_id == "AL") %>%  
 pull(lat)  
auburn\_long <- subset(cities.data, city == "Auburn" & state\_id == "AL") %>%  
 pull(long)  
  
distance\_df <- data.frame(city1 = character(0), city2 = character(0), distance\_km = numeric(0))  
  
for (i in 1:nrow(cities.data)) {  
 if (cities.data$city[i] != "Auburn" | cities.data$state\_id[i] != "AL") {  
 lat\_city <- cities.data$lat[i]  
 lon\_city <- cities.data$long[i]  
 distance <- haversine\_distance(auburn\_lat, auburn\_long, lat\_city, lon\_city)  
 new\_row <- data.frame(city1 = cities.data$city[i], city2 = "Auburn", distance\_km = distance)  
 distance\_df <- rbind(distance\_df, new\_row)  
 }  
}  
  
head(distance\_df, 10)

## 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  
## 7 Philadelphia Auburn 1239.9732  
## 8 Atlanta Auburn 162.5121  
## 9 Washington Auburn 1036.9900  
## 10 Boston Auburn 1665.6985

# QUESTION 7

[Link to GitHub](https://github.com/alexberry8/In-Class-Coding-Challenges)