Assignment\_6

Logan Luchs

2025-03-27

Table of Contents

[Link to my Github](https://github.com/Logz1n/Reproducibility_Class)

#1 Writing your own functions and iterations allows other people to use your code for other things. Therefore aiding reproducibility of your code.

#2 for loops start with an initialization such as for (i in X){, then proceed with the function until everything in x has been looped over. A function is placed after the open bracket. For example for(i in x) {logx <- log(i). To return the results print() can be used. and example would be print(logx). After that the function is closed with the closed bracket }. The total code for the example would look like this: ##for(i in x){ ##logx <- log(i) ##print(logx) ##}

# Loading libraries and color pallettes

#Loading Libraries  
library(ggplot2)   
library(knitr)  
library(readr)  
library(ggpubr)  
library(dplyr)  
library(tidyverse)  
library(markdown)  
library(drc)  
  
#Colorblind pallette  
cbbPalette <- c("#56B4E9", "#009E73", "#F0E442",  
"#000000", "#D55E00", "#CC79A7", "#E69F00","#0072B2" ) #loading a color pallette

#3. Reading in Data

Cities <- read.csv("Cities.csv", na = "na" ) #loading in the data so that R understands na is na so the column is numeric  
  
AuburnLatLong <- Cities %>%  
 subset(city == "Auburn")  
#subsetting to get data for the second city  
Cities2 <- Cities %>%  
 mutate(city1 = city, city2 = "Auburn", lat2 = AuburnLatLong$lat, lon2 = AuburnLatLong$long)  
#Create a column with the second city metrics, since only auburn is wanted

#4.The Haversine function

Haversine <- 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  
# Calculate the distance  
distance\_km <- (earth\_radius \* c)/1000  
return(distance\_km)  
}

#5. Distance from New York to Auburn

AUNY <- Cities2 %>%  
 dplyr::select(city1, city2, lat, long, lat2,lon2) %>%  
 subset(city1 == c("New York"))  
#subsetting so only the distance from new york to auburn is calculated  
   
Haversine(AUNY$lat, AUNY$long, AUNY$lat2, AUNY$lon2)

## [1] 1367.854

#the haversine function above, using the subsetted data to calculate the distance

#6 and Bonus. Distances from Auburn to cities

distances\_km <- NULL #creates a null object so tidyverse will work  
AUDistance <- Cities2 %>%  
 group\_by(city) %>% #grouping by city  
 nest() %>% #nesting the data  
 mutate(distances\_km= map(data, ~Haversine(.$lat,.$long,.$lat2,.$lon2))) %>% #creates a new column after mapping the data and doing the haversine function  
 unnest(c(data,distances\_km)) %>% #unnests data and distances  
 ungroup(city) %>% #ungroups city so we can use the select function  
 dplyr::select(city1, city2, distances\_km) #selects three columns we want in the resulting dataframe (removes excess garbage)  
 AUDistance

## # A tibble: 40 × 3  
## city1 city2 distances\_km  
## <chr> <chr> <dbl>  
## 1 New York Auburn 1368.  
## 2 Los Angeles Auburn 3052.  
## 3 Chicago Auburn 1046.  
## 4 Miami Auburn 916.  
## 5 Houston Auburn 993.  
## 6 Dallas Auburn 1056.  
## 7 Philadelphia Auburn 1240.  
## 8 Atlanta Auburn 163.  
## 9 Washington Auburn 1037.  
## 10 Boston Auburn 1666.  
## # ℹ 30 more rows