CodingChallenge6

Mamata K C

2025-03-27

Contents

Loops and iterations – 25 pts

PLEASE READ THIS BEFORE CONTINUING

This assignment will help you practice writing and executing for loops and writing functions. It will also involve some more practice with GitHub. You may collaborate with a partner to enhance your learning experience. Please ensure the following:

• Collaboration: If you work with a partner, include both names on the final submission by editing the YAML header. • Submission: Only one person should submit the assignment to Canvas in a Word document or .pdf file generated through R markdown. Additionally, you should provide a link to your GitHub, where the assignment should be viewable by rendering it as a GitHub-flavored markdown file. • Setup: It is also assumed you already have a GitHub repository for this class. • Time: This should take you no longer than the class period to complete.

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

- We can compute the same calculations R using the mathematical formula and copy pasting the formula multiple times and just changing the value of the variable. However, in case of complicated calculations, we might do mistakes during copying and pasting process and there occurs problem in reproducibility. Further, it is also a long and tedious process for a large set of data. Thus, creating our own function to do the calculations and use iterations to do same calculation multiple times for different data helps in reproducing our analysis.

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.

Function

- to write a function, we name the function first and use function() function where we write the input data that we want to change. Then we write the formula inside the curly brackets, where we can also save the name of the formula and return the name of the formula inside the curly brackets as output.

for example: NameFunction <- function(inputdata){ NameFormula <- (formula) return(NameFormula) }

The above code will take the input data and perform the calculation using the formula we write down and give us the output.

for loop

- for loop sets up iteration
- while running for loop, it takes every data from the input data and runs the code that we give it inside curly brackets until all the input values are covered and gives us the output.

for example: for (i in 1:100) { result <- NameFunction(i) print(result) }

- The above code takes all numbers from 1 to 100, performs the calculation using the function that we created above for all of them and gives output.

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.

```
cities <- read.csv("Cities.csv") #  loading file using relative file path
cities
```

```
##             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
## 11       Phoenix       Phoenix       AZ              Arizona        4013
## 12       Detroit       Detroit       MI             Michigan       26163
## 13       Seattle       Seattle       WA           Washington       53033
## 14 San Francisco San Francisco       CA           California        6075
## 15     San Diego     San Diego       CA           California        6073
## 16   Minneapolis   Minneapolis       MN            Minnesota       27053
## 17         Tampa         Tampa       FL              Florida       12057
## 18      Brooklyn      Brooklyn       NY             New York       36047
## 19        Denver        Denver       CO             Colorado        8031
## 20        Queens        Queens       NY             New York       36081
## 21     Riverside     Riverside       CA           California        6065
## 22     Las Vegas     Las Vegas       NV               Nevada       32003
## 23     Baltimore     Baltimore       MD             Maryland       24510
## 24     St. Louis     St. Louis       MO             Missouri       29510
## 25      Portland      Portland       OR               Oregon       41051
## 26   San Antonio   San Antonio       TX                Texas       48029
## 27    Sacramento    Sacramento       CA           California        6067
## 28        Austin        Austin       TX                Texas       48453
## 29       Orlando       Orlando       FL              Florida       12095
## 30      San Juan      San Juan       PR          Puerto Rico       72127
## 31      San Jose      San Jose       CA           California        6085
## 32  Indianapolis  Indianapolis       IN              Indiana       18097
## 33    Pittsburgh    Pittsburgh       PA         Pennsylvania       42003
## 34    Cincinnati    Cincinnati       OH                 Ohio       39061
## 35     Manhattan     Manhattan       NY             New York       36061
## 36   Kansas City   Kansas City       MO             Missouri       29095
## 37     Cleveland     Cleveland       OH                 Ohio       39035
## 38      Columbus      Columbus       OH                 Ohio       39049
## 39         Bronx         Bronx       NY             New York       36005
## 40        Auburn        Auburn       AL              Alabama        1081
##            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
## 11             Maricopa 33.5722 -112.0892    4065338  1210.3
## 12                Wayne 42.3834  -83.1024    3716929  1771.8
## 13                 King 47.6211 -122.3244    3555253  3408.0
## 14        San Francisco 37.7558 -122.4449    3364862  6914.5
## 15            San Diego 32.8313 -117.1222    3057778  1640.0
## 16             Hennepin 44.9635  -93.2678    2906807  3052.0
## 17         Hillsborough 27.9945  -82.4447    2906035  1336.6
## 18                Kings 40.6501  -73.9496    2736074 15200.5
## 19               Denver 39.7620 -104.8758    2691349  1800.3
## 20               Queens 40.7498  -73.7976    2405464  8503.7
## 21            Riverside 33.9381 -117.3949    2288508  1504.7
## 22                Clark 36.2333 -115.2654    2256509  1771.5
## 23            Baltimore 39.3051  -76.6144    2189589  2753.1
## 24            St. Louis 38.6359  -90.2451    2127843  1833.6
## 25            Multnomah 45.5371 -122.6500    2084045  1859.0
## 26                Bexar 29.4632  -98.5238    2069843  1129.0
## 27           Sacramento 38.5677 -121.4685    1962998  2054.1
## 28               Travis 30.3005  -97.7522    1915031  1145.0
## 29               Orange 28.4773  -81.3370    1913597  1082.0
## 30             San Juan 18.3985  -66.0610    1809800  3110.1
## 31          Santa Clara 37.3012 -121.8480    1771563  2148.3
## 32               Marion 39.7771  -86.1458    1740984   943.3
## 33            Allegheny 40.4397  -79.9763    1712828  2116.9
## 34              Hamilton 39.1413  -84.5060    1704916  1534.2
## 35              New York 40.7834  -73.9662    1694263 28653.9
## 36               Jackson 39.1238  -94.5541    1686807   623.6
## 37              Cuyahoga 41.4764  -81.6805    1679247  1825.4
## 38              Franklin 39.9862  -82.9855    1578153  1585.7
## 39                 Bronx 40.8501  -73.8662    1472654 13356.3
## 40                   Lee 32.6087  -85.4903     108327   486.5
```

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

```r
# writing a function to calculate distance between two coordinates

haversine_distance <- function(lat1, lon1, lat2, lon2) {
  rad.lat1 <- lat1 * pi/180 #converting to radians
  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))  #haversine formula
  earth_radius <- 6378137 #earth's radius in kilometers
  distance_km <- (earth_radius * c)/1000 #calculating the distance
  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.

```r
# Subsetting required values of latitude and longitude values

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

# Using the function to compute distance between Auburn and New York City.
DistanceAuburnNYC <- haversine_distance(lat1, lon1, lat2, lon2)
print(DistanceAuburnNYC)
```

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

b. The output of your function should be 1367.854 km

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

```r
lat2 <- cities$lat[cities$city!="Auburn"]  #subsetting data for lattitudes for all cities except Auburn
lon2 <- cities$long[cities$city!="Auburn"] #subsetting data for longitudes for all cities except Auburn

#for loop to calculate the distance between all other cities and Auburn
```

```r
for (i in seq_along(lat2)){
  result <- haversine_distance(lat1, lon1, lat2[i], lon2[i])
  print(result)
}
```

```
## [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
```

Bonus point if you can have the output of each iteration append a new row to a dataframe, generating a new column of data. In other words, the loop should create a dataframe with three columns called city1, city2, and distance_km, as shown below. The first six rows of the dataframe are shown below.

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

```
CitiesDistance = NULL #creating a NULL object

#for loop to calculate the distance between all other cities and Auburn
for (i in seq_along(lat2)) {
  result <- haversine_distance(lat1, lon1, lat2[i], lon2[i])
  City1 <- cities$city[i]   #creating a vector City1 with all other cities
  City2 <- "Auburn"   #creating a vector City2 with city Auburn
  CombinedData <- data.frame(City1 = City1, City2 = City2, Distance_km = result)
  CitiesDistance <- rbind.data.frame(CitiesDistance, CombinedData)
}


CitiesDistance
```

```
##            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
## 11       Phoenix Auburn   2476.2552
## 12       Detroit Auburn   1108.2288
## 13       Seattle Auburn   3507.9589
## 14 San Francisco Auburn   3388.3656
## 15     San Diego Auburn   2951.3816
## 16   Minneapolis Auburn   1530.2000
## 17         Tampa Auburn    591.1181
## 18      Brooklyn Auburn   1363.2072
## 19        Denver Auburn   1909.7897
## 20        Queens Auburn   1380.1382
## 21     Riverside Auburn   2961.1199
## 22     Las Vegas Auburn   2752.8142
## 23     Baltimore Auburn   1092.2595
## 24     St. Louis Auburn    796.7541
## 25      Portland Auburn   3479.5376
```

```
## 26   San Antonio Auburn   1290.5492
## 27    Sacramento Auburn   3301.9923
## 28        Austin Auburn   1191.6657
## 29       Orlando Auburn    608.2035
## 30      San Juan Auburn   2504.6312
## 31      San Jose Auburn   3337.2781
## 32  Indianapolis Auburn    800.1452
## 33    Pittsburgh Auburn   1001.0879
## 34    Cincinnati Auburn    732.5906
## 35     Manhattan Auburn   1371.1633
## 36   Kansas City Auburn   1091.8970
## 37     Cleveland Auburn   1043.2727
## 38      Columbus Auburn    851.3423
## 39         Bronx Auburn   1382.3721
```

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

Link to my GitHub