**Unexecuted Code**

## Homework #3 - 7 Aug 2018

#Step 1 Create a function (named readStates) to read a CSV file into R

#--------------------------

#creating a default for the url argument

readStates <- function(url =

"http://www2.census.gov/programs-surveys/popest/tables/2010-2011/state/totals/nst-est2011-01.csv"){

# Reading the csv and preventing the strings to be read as factors

df <- read.csv(url(url), stringsAsFactors = F)

#Step 2 Clean the dataframe

#--------------------------

#deleting the first 8 rows and the last 5 columns

df <- df[-1:-8, -6:-10]

#deleting the last 7 rows

df <- df[-52:-58,]

#Creating a vector of column names

colNames <- c("stateName", "Jul2010", "Jul2011", "base2010", "base2011")

#Assigning the column names to the data frame

colnames(df) <- colNames

#resetting the row numbers

rownames(df) <- NULL

#Getting rid of the period before all the state names

df$stateName <- gsub("\\.", "", df$stateName)

#Writing a for loop to go through each of the columns 2 through 5

for (col in 2:5){

#Replacing the comma with nothing

df[,col] <- gsub(",", "", df[,col])

#replacing any spaces with nothing

df[,col] <- gsub(" ", "", df[,col])

#Converting the strings to numbers

df[,col] <- as.numeric(df[,col])

}

# returning the data frame

return(df)

}

#Step 3 Store and explore

#--------------------------

#Assigning the data frame to a variable dfStates

dfStates <- readStates()

#Testing to see if the mean for July2011 is 6,053,834

#Saving as the variable pop.avg

(pop.avg <- mean(dfStates$Jul2011))

#Step 4 Find the state name with the highest population

#--------------------------

#State with the highest population

(dfStates[which.max(dfStates$Jul2011), "stateName"])

#Sorting the data based on July2011 in increasing order

(dfStates.sorted <- dfStates[order(dfStates$Jul2011),])

#Step 5 Explore the distrubution of the states

#--------------------------

# Write a function that takes two parameters. The first is a vector and the second is a number.

#Defining the function

myFunction <- function(vec, num){

#getting the number of points that are less than the given number

numLess <- sum(vec < num)

#retunring the number of points less than the number

return(numLess / length(vec))

}

#testing the function with the play data

vec <- c(1,2,3,4,5)

num <- 2

(myFunction(vec, 2))

#exploring percentage of states with a population below the mean

myFunction(dfStates$Jul2011, pop.avg)

# 2/3s of the states are below the mean

**Console log w/plot**

**Executed code**

> ## Homework #3 - 7 Aug 2018

>

> #Step 1 Create a function (named readStates) to read a CSV file into R

> #--------------------------

>

> #creating a default for the url argument

> readStates <- function(url =

+ "http://www2.census.gov/programs-surveys/popest/tables/2010-2011/state/totals/nst-est2011-01.csv"){

+ # Reading the csv and preventing the strings to be read as factors

+ df <- read.csv(url(url), stringsAsFactors = F)

+

+ #Step 2 Clean the dataframe

+ #--------------------------

+

+ #deleting the first 8 rows and the last 5 columns

+ df <- df[-1:-8, -6:-10]

+ #deleting the last 7 rows

+ df <- df[-52:-58,]

+ #Creating a vector of column names

+ colNames <- c("stateName", "Jul2010", "Jul2011", "base2010", "base2011")

+ #Assigning the column names to the data frame

+ colnames(df) <- colNames

+ #resetting the row numbers

+ rownames(df) <- NULL

+ #Getting rid of the period before all the state names

+ df$stateName <- gsub("\\.", "", df$stateName)

+ #Writing a for loop to go through each of the columns 2 through 5

+ for (col in 2:5){

+ #Replacing the comma with nothing

+ df[,col] <- gsub(",", "", df[,col])

+ #replacing any spaces with nothing

+ df[,col] <- gsub(" ", "", df[,col])

+ #Converting the strings to numbers

+ df[,col] <- as.numeric(df[,col])

+ }

+ # returning the data frame

+ return(df)

+ }

>

>

> #Step 3 Stor and explore

> #--------------------------

>

> #Assigning the data frame to a variable dfStates

> dfStates <- readStates()

>

> #Testing to see if the mean for July2011 is 6,053,834

> #Saving as the variable pop.avg

> (pop.avg <- mean(dfStates$Jul2011))

[1] 6053834

>

>

> #Step 4 Find the state name with the highest population

> #--------------------------

> #State with the highest population

> (dfStates[which.max(dfStates$Jul2011), "stateName"])

[1] "California"

>

> #Sorting the data based on July2011 in increasing order

> (dfStates.sorted <- dfStates[order(dfStates$Jul2011),])

stateName Jul2010 Jul2011 base2010 base2011

51 Wyoming 563626 563626 564554 568158

9 District of Columbia 601723 601723 604912 617996

46 Vermont 625741 625741 625909 626431

35 North Dakota 672591 672591 674629 683932

2 Alaska 710231 710231 714146 722718

42 South Dakota 814180 814180 816598 824082

8 Delaware 897934 897934 899792 907135

27 Montana 989415 989415 990958 998199

40 Rhode Island 1052567 1052567 1052528 1051302

30 New Hampshire 1316470 1316472 1316807 1318194

20 Maine 1328361 1328361 1327379 1328188

12 Hawaii 1360301 1360301 1363359 1374810

13 Idaho 1567582 1567582 1571102 1584985

28 Nebraska 1826341 1826341 1830141 1842641

49 West Virginia 1852994 1852996 1854368 1855364

32 New Mexico 2059179 2059180 2065913 2082224

29 Nevada 2700551 2700551 2704283 2723322

45 Utah 2763885 2763885 2775479 2817222

17 Kansas 2853118 2853118 2859143 2871238

4 Arkansas 2915918 2915921 2921588 2937979

25 Mississippi 2967297 2967297 2970072 2978512

16 Iowa 3046355 3046350 3050202 3062309

7 Connecticut 3574097 3574097 3575498 3580709

37 Oklahoma 3751351 3751354 3760184 3791508

38 Oregon 3831074 3831074 3838332 3871859

18 Kentucky 4339367 4339362 4347223 4369356

19 Louisiana 4533372 4533372 4545343 4574836

41 South Carolina 4625364 4625364 4637106 4679230

1 Alabama 4779736 4779735 4785401 4802740

6 Colorado 5029196 5029196 5047692 5116796

24 Minnesota 5303925 5303925 5310658 5344861

50 Wisconsin 5686986 5686986 5691659 5711767

21 Maryland 5773552 5773552 5785681 5828289

26 Missouri 5988927 5988927 5995715 6010688

43 Tennessee 6346105 6346110 6357436 6403353

3 Arizona 6392017 6392013 6413158 6482505

15 Indiana 6483802 6483800 6490622 6516922

22 Massachusetts 6547629 6547629 6555466 6587536

48 Washington 6724540 6724540 6742950 6830038

47 Virginia 8001024 8001030 8023953 8096604

31 New Jersey 8791894 8791894 8799593 8821155

34 North Carolina 9535483 9535475 9560234 9656401

11 Georgia 9687653 9687660 9712157 9815210

23 Michigan 9883640 9883635 9877143 9876187

36 Ohio 11536504 11536502 11537968 11544951

39 Pennsylvania 12702379 12702379 12717722 12742886

14 Illinois 12830632 12830632 12841980 12869257

10 Florida 18801310 18801311 18838613 19057542

33 New York 19378102 19378104 19395206 19465197

44 Texas 25145561 25145561 25253466 25674681

5 California 37253956 37253956 37338198 37691912

>

>

> #Step 5 Explore the distrubution of the states

> #--------------------------

> # Write a function that takes two parameters. The first is a vector and the second is a number.

>

> #Defining the function

> myFunction <- function(vec, num){

+ #getting the number of points that are less than the given number

+ numLess <- sum(vec < num)

+ #retunring the number of points less than the number

+ return(numLess / length(vec))

+ }

>

> #testing the function with the play data

> vec <- c(1,2,3,4,5)

> num <- 2

> (myFunction(vec, 2))

[1] 0.2

>

> #exploring percentage of states with a population below the mean

> myFunction(dfStates$Jul2011, pop.avg)

[1] 0.6666667

> # 2/3s of the states are below the mean