Assignmnt1\_of\_Session6

Sanjeev Tiwari

October 13, 2017

# Question - 1: Function to check prime numbers  
# ---------Function Definition --------------------------------------------------------------  
fn.checkprime<-function(vecNum){  
 retVal<-seq(1:length(vecNum)) # Create a dummy vector to capture the result to be returned  
 intI<-1 #Declare and initilize a loop variable  
 while(intI <= length(vecNum)) { #Set the while loop to iterate for all the elements of vector  
 intInput<-vecNum[intI] #Access the vector element corresponding to iteration cycle  
 if(!is.numeric(intInput)){ # Validation check for non numeric values & displays a conversion warning  
 warning('The character values being converted to numeric', call. = TRUE, immediate. = FALSE)  
 intInput<-as.numeric(intInput) #Convert to numeric  
 if(is.na(intInput)){ #Validation check for failure in conversion & throws the exception   
 stop('Invalid input. Enter a numeric value.', call. = TRUE)  
 }  
 }  
   
 intJ<-2  
 while(intJ < intInput){ #While loop condition to test the vector element value  
 if(intInput %% intJ == 0){ # checking if the remainder is zero  
 break # breaks the loop if remainder 0 is obtained, result can be concluded and control jumps to pick next element value  
 } else {  
 intJ<-intJ+1 #incrementing the dinominator value for checking the division result  
 }  
 }  
 if(intJ==intInput){ # checking If the input number is same as the dinnominator  
 #print(cat(intInput,'Prime number', sep = ' -->', fill = FALSE))  
 retVal[intI]<-paste(intInput,'Prime number\n', sep = '-->') # Capturing/adding the conclusion to a vector  
 } else {   
 #print(cat(intInput,'Not a prime number', sep = ' -->', fill = FALSE))  
 retVal[intI]<-paste(intInput,'Not a prime number\n', sep = ' -->') # Capturing/adding the conclusion to a vector  
 }  
 intI<-intI+1 #To incrementing the index value to refer to next element of input vector  
 }  
 return(retVal) # Return the result as vector  
}  
  
# Demo examples to call the function  
#--------------------------------------------  
fn.checkprime(c('103','82','179'))

## Warning in fn.checkprime(c("103", "82", "179")): The character values being  
## converted to numeric  
  
## Warning in fn.checkprime(c("103", "82", "179")): The character values being  
## converted to numeric  
  
## Warning in fn.checkprime(c("103", "82", "179")): The character values being  
## converted to numeric

## [1] "103-->Prime number\n" "82 -->Not a prime number\n"  
## [3] "179-->Prime number\n"

fn.checkprime(c(103,82,179))

## [1] "103-->Prime number\n" "82 -->Not a prime number\n"  
## [3] "179-->Prime number\n"

fn.checkprime(103)

## [1] "103-->Prime number\n"

fn.checkprime(82)

## [1] "82 -->Not a prime number\n"

fn.checkprime(179)

## [1] "179-->Prime number\n"

# commented to enable compilation of R-Markdown output  
# fn.checkprime('a')   
# fn.checkprime(c('a','b','c'))

# question - 2:  
# install.packages('stringr') ## Prerequisites  
# -----------------------------------------------------------------------  
fn.detectchar<-function(srcString,schString,blnTogether){  
 library('stringr') # Loading the stringr parckage  
   
 intSchCount = 0 # Initializing program variable - capture count of match cases  
 intIndexSrc = 1 #used a index to refer to source string vector elements  
 schChar<-schString[1] #picking the first search pattern from the vector  
   
 while(length(srcString) >= intIndexSrc ){ #Iterate for all elements of source string vector  
   
 #cat('Checking element: ',trimws(paste(intIndexSrc,'(',srcString[intIndexSrc],')'),which = 'both'),'\n')  
   
 #Displays the header title for the match operation being performed  
 cat('Checking element: ',intIndexSrc,trimws('(',which='both'),srcString[intIndexSrc],trimws(')',which = 'both'),'\n')  
 for(schChar in schString){ #iterating for each element of search pattern vector  
 #Checking if there is a match, note, the use of blnFlag argumnent--> reserved for further enhancements  
 if(TRUE == blnTogether && TRUE == (str\_detect(srcString[intIndexSrc], schChar ))){  
 cat(schChar, ' - search pattern APPEARS in given string:', '\n') #displaying the match result as formatted string mat  
 intSchCount<-intSchCount+1 #Capturing the count of match cases  
 }else if(blnTogether == FALSE){ #else case reserved for future implementation  
 cat(schChar, ' To be implemented...', '\n')  
 } else { # if the match is not found, displays the result as formatted string  
 cat(schChar, ' - search pattern DOES NOT APPEAR in string:', '\n')  
 }  
 }  
 #Investigating the total match cases out of provisionedand and displaying the conclusion for each element of source vector  
 if(intSchCount == length(schString)){  
 cat('ALL CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - ',trimws(srcString[intIndexSrc],which='both'),'\n')  
 }else if(intSchCount > 0){  
 cat('NOT ALL, BUT ', intSchCount , 'CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - ',trimws(srcString[intIndexSrc],which='both'),'\n')  
 }else{  
 cat('NONE OF THE CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - ',trimws(srcString[intIndexSrc],which='both'),'\n')   
 }  
 cat('\n') #adding a new line as separator  
 intSchCount<-0 #resetting the count for capturing the summary of next element/match case  
 intIndexSrc<-intIndexSrc+1 #increase the index for referring to next element   
 }  
}  
  
# Demo examples to call the function  
#--------------------------------------------  
words<-c("above", "unit", "Under", "argument")  
schStr<-c('a','u','o')  
fn.detectchar(srcString = words, schString = schStr, blnTogether = T)

##   
## Attaching package: 'stringr'

## The following object is masked \_by\_ '.GlobalEnv':  
##   
## words

## Checking element: 1 ( above )   
## a - search pattern APPEARS in given string:   
## u - search pattern DOES NOT APPEAR in string:   
## o - search pattern APPEARS in given string:   
## NOT ALL, BUT 2 CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - above   
##   
## Checking element: 2 ( unit )   
## a - search pattern DOES NOT APPEAR in string:   
## u - search pattern APPEARS in given string:   
## o - search pattern DOES NOT APPEAR in string:   
## NOT ALL, BUT 1 CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - unit   
##   
## Checking element: 3 ( Under )   
## a - search pattern DOES NOT APPEAR in string:   
## u - search pattern DOES NOT APPEAR in string:   
## o - search pattern DOES NOT APPEAR in string:   
## NONE OF THE CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - Under   
##   
## Checking element: 4 ( argument )   
## a - search pattern APPEARS in given string:   
## u - search pattern APPEARS in given string:   
## o - search pattern DOES NOT APPEAR in string:   
## NOT ALL, BUT 2 CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - argument

words<-c("above", "unit", "Under", "argument")  
schStr<-c('abo','un','ooo')  
fn.detectchar(srcString = words, schString = schStr, blnTogether = T)

## Checking element: 1 ( above )   
## abo - search pattern APPEARS in given string:   
## un - search pattern DOES NOT APPEAR in string:   
## ooo - search pattern DOES NOT APPEAR in string:   
## NOT ALL, BUT 1 CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - above   
##   
## Checking element: 2 ( unit )   
## abo - search pattern DOES NOT APPEAR in string:   
## un - search pattern APPEARS in given string:   
## ooo - search pattern DOES NOT APPEAR in string:   
## NOT ALL, BUT 1 CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - unit   
##   
## Checking element: 3 ( Under )   
## abo - search pattern DOES NOT APPEAR in string:   
## un - search pattern DOES NOT APPEAR in string:   
## ooo - search pattern DOES NOT APPEAR in string:   
## NONE OF THE CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - Under   
##   
## Checking element: 4 ( argument )   
## abo - search pattern DOES NOT APPEAR in string:   
## un - search pattern DOES NOT APPEAR in string:   
## ooo - search pattern DOES NOT APPEAR in string:   
## NONE OF THE CHARACTERS IN SEARCH PATTERN APPEAR IN STRING - argument

# question - 3:  
# ---------Function Definition --------------------------------------------------------------  
fn.checkBMI<-function(weight, height){  
 return(weight/(height^2))\*703  
}  
  
# -------------------------Demo Call's----------------------------------------------  
fn.checkBMI(50, 1.75)

## [1] 16.32653

fn.checkBMI(72, 1.65)

## [1] 26.44628

# question - 4:  
# ---------Function Definition --------------------------------------------------------------  
fn.sumofcubes<-function(x){ # function definition  
 #validations - if the input is <1 or a non numeric value, returns zero  
 if(x<=0 || !is.numeric(x)) return(0)   
 else{ #performing a recursive call to return the sum of cubes as described in the problem statement  
 return(x^3 + fn.sumofcubes(x-1))  
 }  
}  
   
# -------------------------Demo Call's----------------------------------------------  
fn.sumofcubes(0)

## [1] 0

fn.sumofcubes('0')

## [1] 0

fn.sumofcubes(6)

## [1] 441

fn.sumofcubes(3)

## [1] 36

fn.sumofcubes(2)

## [1] 9

# question - 5:  
# ---------Function Definition --------------------------------------------------------------  
fn.Mode<-function(x){ # function definition  
 #validations - if the input is <1 or a non numeric value, returns zero  
 if(length(x)<=1 || !is.numeric(x)) return('Insufficient or invalid arguments supplied!!!')   
 else{   
 retVal<-unique(x) #checking the uniqueness  
   
 # matches the unique set to value within the supplied vector to return the a vector of frequency or occurence  
 retVal<-tabulate(match(x, retVal))   
   
 #Returns the value from the vector where the vector element holds highest value (e of which.max())   
 return(retVal[which.max(retVal)])  
 }  
}  
  
# -------------------------Demo Call's----------------------------------------------  
x = c(2,3,3,4,4,4,4,4,5,6,7,9,10)  
fn.Mode(x)

## [1] 5

# Question - 6:  
# ---------Function Definition --------------------------------------------------------------  
## Already created a functiona in response to question-1, please refer to the demo call section   
#for the given vector  
  
# -------------------------Demo Call's----------------------------------------------  
x <- c(2,2,3,3,4,5,7,11,15,19,24,29)  
fn.checkprime(x)

## [1] "2-->Prime number\n" "2-->Prime number\n"   
## [3] "3-->Prime number\n" "3-->Prime number\n"   
## [5] "4 -->Not a prime number\n" "5-->Prime number\n"   
## [7] "7-->Prime number\n" "11-->Prime number\n"   
## [9] "15 -->Not a prime number\n" "19-->Prime number\n"   
## [11] "24 -->Not a prime number\n" "29-->Prime number\n"

# Question - 7:  
# ---------Function Definition --------------------------------------------------------------  
# Steps to create a package are -   
# 1. Create a folder to the package files/skeleton files and R command files   
# and set it a home directory using the setwd() command  
# 2. Download and install RTools component - required for compiling the package  
# 3. As part of installation, edit the PATH environmental variable to setup RTools commands in environment  
# 4. Create a function / set of functions thats needs to be enclosed within a package  
# 5. Create a package skeleton using the command - package.skeleton() - this command generates the skeleton   
# structure for the package to be created and published. it contains Description, namespace files in additions   
# to R, man and data directories   
# 6. Edit the description and help files with details  
# 7. Build the package using the command - R CMD build <package name>  
# 8. Install the package using the command - R CMD INSTALL <Package>  
# 9. Refresh/Reload RStudio and locate the package under the packages menu  
# 10. Load the package using library() command and exdperiment around functions included in the package  
  
# The package mentioned in the assignment question will be compiled and uploaded to the git url.   
  
print('installation of RTools for compilation of the package - The package mentioned in the assignment question will be compiled and uploaded to the git url. ')

## [1] "installation of RTools for compilation of the package - The package mentioned in the assignment question will be compiled and uploaded to the git url. "

# Question - 8:  
# inspall.packages('dplyr') # Prerequisite for the fucntion  
# ---------Function Definition --------------------------------------------------------------  
# Need to solve the questions with the help of DATA FRAMES  
  
# 8(a)  
# dsBoys<-read.csv(file.choose(), header = TRUE);head(dsBoys) #performed with data file Boys-top100.csv (included in repository)  
# dsGirls<-read.csv(file.choose(), header = TRUE);head(dsGirls) #performed with data file Girls-top100.csv (included in repository)  
  
dfStu<-data.frame(roll\_no = c(3,1,2,5,4,7,8), names = c('peter', 'jack', 'david', 'james', 'john','Sanjeev', 'Anshul'));dfStu

## roll\_no names  
## 1 3 peter  
## 2 1 jack  
## 3 2 david  
## 4 5 james  
## 5 4 john  
## 6 7 Sanjeev  
## 7 8 Anshul

dfMarks<-data.frame(roll\_no = c(4,12,13,6,15), maths = c(89,92,76,67,90), science = c(98,92,88,91,92));dfMarks

## roll\_no maths science  
## 1 4 89 98  
## 2 12 92 92  
## 3 13 76 88  
## 4 6 67 91  
## 5 15 90 92

# 8(b) Exampe of inner\equi join  
jnInner<-merge(dfStu, dfMarks);jnInner

## roll\_no names maths science  
## 1 4 john 89 98

# (8c) Exampe of left outer join  
jnLouter <- merge(dfStu, dfMarks, by = 'roll\_no', all.x=TRUE);jnLouter

## roll\_no names maths science  
## 1 1 jack NA NA  
## 2 2 david NA NA  
## 3 3 peter NA NA  
## 4 4 john 89 98  
## 5 5 james NA NA  
## 6 7 Sanjeev NA NA  
## 7 8 Anshul NA NA

# (8d) Exampe of right outer join  
jnRouter <- merge(dfStu, dfMarks, by = 'roll\_no', all.y=TRUE);jnRouter

## roll\_no names maths science  
## 1 4 john 89 98  
## 2 6 <NA> 67 91  
## 3 12 <NA> 92 92  
## 4 13 <NA> 76 88  
## 5 15 <NA> 90 92

# 8(e) Exampe of outer join  
jnFouter<-merge(dfStu, dfMarks, by = 'roll\_no', all = TRUE);jnFouter

## roll\_no names maths science  
## 1 1 jack NA NA  
## 2 2 david NA NA  
## 3 3 peter NA NA  
## 4 4 john 89 98  
## 5 5 james NA NA  
## 6 6 <NA> 67 91  
## 7 7 Sanjeev NA NA  
## 8 8 Anshul NA NA  
## 9 12 <NA> 92 92  
## 10 13 <NA> 76 88  
## 11 15 <NA> 90 92

# Exampe of cross join  
jnCross<-merge(dfStu, dfMarks, by = NULL);jnCross

## roll\_no.x names roll\_no.y maths science  
## 1 3 peter 4 89 98  
## 2 1 jack 4 89 98  
## 3 2 david 4 89 98  
## 4 5 james 4 89 98  
## 5 4 john 4 89 98  
## 6 7 Sanjeev 4 89 98  
## 7 8 Anshul 4 89 98  
## 8 3 peter 12 92 92  
## 9 1 jack 12 92 92  
## 10 2 david 12 92 92  
## 11 5 james 12 92 92  
## 12 4 john 12 92 92  
## 13 7 Sanjeev 12 92 92  
## 14 8 Anshul 12 92 92  
## 15 3 peter 13 76 88  
## 16 1 jack 13 76 88  
## 17 2 david 13 76 88  
## 18 5 james 13 76 88  
## 19 4 john 13 76 88  
## 20 7 Sanjeev 13 76 88  
## 21 8 Anshul 13 76 88  
## 22 3 peter 6 67 91  
## 23 1 jack 6 67 91  
## 24 2 david 6 67 91  
## 25 5 james 6 67 91  
## 26 4 john 6 67 91  
## 27 7 Sanjeev 6 67 91  
## 28 8 Anshul 6 67 91  
## 29 3 peter 15 90 92  
## 30 1 jack 15 90 92  
## 31 2 david 15 90 92  
## 32 5 james 15 90 92  
## 33 4 john 15 90 92  
## 34 7 Sanjeev 15 90 92  
## 35 8 Anshul 15 90 92

# 8(f) - Filter operation  
library(dplyr) # using dplyr package to call filter function

## Warning: package 'dplyr' was built under R version 3.4.2

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

# using filter() to return all rows where is.na() is TRUE  
filter(jnFouter,is.na(jnFouter$maths))

## Warning: package 'bindrcpp' was built under R version 3.4.2

## roll\_no names maths science  
## 1 1 jack NA NA  
## 2 2 david NA NA  
## 3 3 peter NA NA  
## 4 5 james NA NA  
## 5 7 Sanjeev NA NA  
## 6 8 Anshul NA NA

# 8(g) - group by, sum and average operations  
# example of group by  
group\_by(jnFouter,jnFouter$maths, add = FALSE)

## # A tibble: 11 x 5  
## # Groups: jnFouter$maths [6]  
## roll\_no names maths science `jnFouter$maths`  
## <dbl> <fctr> <dbl> <dbl> <dbl>  
## 1 1 jack NA NA NA  
## 2 2 david NA NA NA  
## 3 3 peter NA NA NA  
## 4 4 john 89 98 89  
## 5 5 james NA NA NA  
## 6 6 <NA> 67 91 67  
## 7 7 Sanjeev NA NA NA  
## 8 8 Anshul NA NA NA  
## 9 12 <NA> 92 92 92  
## 10 13 <NA> 76 88 76  
## 11 15 <NA> 90 92 90

# example of sum  
summ<-sum(jnFouter$maths);summ

## [1] NA

sums<-sum(jnFouter$science);sums

## [1] NA

# example of average  
avgm<-mean(jnFouter$maths);avgm

## [1] NA

avgs<-mean(jnFouter$science);avgs

## [1] NA

# example of summary grouping  
group\_by(jnFouter,jnFouter$maths, add = TRUE) %>%   
 summarise(sum = sum(jnFouter$maths),average = mean(jnFouter$maths))

## # A tibble: 6 x 3  
## `jnFouter$maths` sum average  
## <dbl> <dbl> <dbl>  
## 1 67 NA NA  
## 2 76 NA NA  
## 3 89 NA NA  
## 4 90 NA NA  
## 5 92 NA NA  
## 6 NA NA NA

# 8(h) -- need more clarity on the problem statement  
# 8(i) -- need more clarity on the problem statement  
# 8(j) -- need more clarity on the problem statement

#Question - 9:  
# inspall.packages('dplyr') # Prerequisite for the fucntion  
# vignette("programming") # displays documentations "Programming with dplyr Package"  
# ---------Function Definition --------------------------------------------------------------  
  
# Creating the data variables  
StNames<-c('Sanjeev', 'Vishwanath', 'Mohit', 'Anbu', 'Prabhat', 'Siddharth', 'Satyam', 'Syandeep', 'Yogesh')  
StdID<-seq(1:length(StNames));StdID

## [1] 1 2 3 4 5 6 7 8 9

Class<-c(2,2,3,2,3,3,5,6,5);length(Class)

## [1] 9

Section<-c(1,1,1,2,2,1,1,3,2);length(Section)

## [1] 9

#Create the data frame using the data variables  
dfStudents<-data.frame(StNames,StdID,Class,Section);dfStudents

## StNames StdID Class Section  
## 1 Sanjeev 1 2 1  
## 2 Vishwanath 2 2 1  
## 3 Mohit 3 3 1  
## 4 Anbu 4 2 2  
## 5 Prabhat 5 3 2  
## 6 Siddharth 6 3 1  
## 7 Satyam 7 5 1  
## 8 Syandeep 8 6 3  
## 9 Yogesh 9 5 2

# Displaying the unique data based on cloumns - Class and Section  
distinct(.data = dfStudents, Class, Section, keep\_all = TRUE)

## Class Section  
## 1 2 1  
## 2 3 1  
## 3 2 2  
## 4 3 2  
## 5 5 1  
## 6 6 3  
## 7 5 2

# Displaying the unique data based on cloumns - Class   
distinct(.data = dfStudents, Class, keep\_all = TRUE)

## Class  
## 1 2  
## 2 3  
## 3 5  
## 4 6

library(dplyr) #using the library for performing grouping and summary operations  
# Grouping the data set values on two variables - Class and Section and displaying the   
# count of StdID values as summary  
dfStudents %>% group\_by(Class) %>% summarise(Summary = n\_distinct(StdID))

## # A tibble: 4 x 2  
## Class Summary  
## <dbl> <int>  
## 1 2 3  
## 2 3 3  
## 3 5 2  
## 4 6 1

dfStudents %>% group\_by(Class, Section) %>% summarise(Summary = n\_distinct(StdID))

## # A tibble: 7 x 3  
## # Groups: Class [?]  
## Class Section Summary  
## <dbl> <dbl> <int>  
## 1 2 1 2  
## 2 2 2 1  
## 3 3 1 2  
## 4 3 2 1  
## 5 5 1 1  
## 6 5 2 1  
## 7 6 3 1

#function to perform summary operation on a data frame  
fn.countSumm<-function(df,...) { #using ... to signify acceptance of any number of arguments  
 # capturing all the arguments in ... as list of formulas  
 group\_var<-quos(...)   
   
 #Performing a groupby on data frame and summarising by of number of rows  
 return(df %>% group\_by(!!!group\_var) %>% summarise(Summary = n()))  
}  
# Calling the function example  
fn.countSumm(dfStudents,Class,Section)

## # A tibble: 7 x 3  
## # Groups: Class [?]  
## Class Section Summary  
## <dbl> <dbl> <int>  
## 1 2 1 2  
## 2 2 2 1  
## 3 3 1 2  
## 4 3 2 1  
## 5 5 1 1  
## 6 5 2 1  
## 7 6 3 1

#function to detect unique & duplicate rows from data frame  
fn.checkRows<-function(df,...,unqFlag, blnLevels=FALSE) {   
 if(blnLevels==TRUE){ # check flag to displaying the levels  
 # using the if else construct to return the result   
 return(ifelse(unqFlag==T, df[!duplicated(df[...]),], df[duplicated(df[...]),]))  
 }else{  
 if(unqFlag==T){  
 return(df[!duplicated(df[...]),]) #Returns the unique rows  
 }else{  
 return(df[duplicated(df[...]),]) #Returns the duplicate rows  
 }  
 }  
}  
# Demo example for using the function  
# ------------------------------------------  
# Calling without specifying the optional parameter  
fn.checkRows(dfStudents,c(3,4),unqFlag = FALSE) # Display Duplicates

## StNames StdID Class Section  
## 2 Vishwanath 2 2 1  
## 6 Siddharth 6 3 1

fn.checkRows(dfStudents,c(3,4),unqFlag = TRUE) # display Unique rows

## StNames StdID Class Section  
## 1 Sanjeev 1 2 1  
## 3 Mohit 3 3 1  
## 4 Anbu 4 2 2  
## 5 Prabhat 5 3 2  
## 7 Satyam 7 5 1  
## 8 Syandeep 8 6 3  
## 9 Yogesh 9 5 2

# Calling specifying the optional parameter  
fn.checkRows(dfStudents,c(3,4),unqFlag = FALSE,blnLevels = TRUE) # Display Duplicates

## [[1]]  
## [1] Vishwanath Siddharth   
## 9 Levels: Anbu Mohit Prabhat Sanjeev Satyam Siddharth ... Yogesh

fn.checkRows(dfStudents,c(3,4),unqFlag = TRUE,blnLevels = TRUE) # display Unique rows

## [[1]]  
## [1] Sanjeev Mohit Anbu Prabhat Satyam Syandeep Yogesh   
## 9 Levels: Anbu Mohit Prabhat Sanjeev Satyam Siddharth ... Yogesh

# Question - 10:  
# ---------Function Definition --------------------------------------------------------------  
# 10(a) - Function to check the dF for complete cases or checking presence of NA in specified columns  
fn.checkNullDS<-function(df, blnChkDS=TRUE, ...=NULL){  
   
 if(blnChkDS==T){ # check if data set check for complete cases required  
 #retVal<-(!complete.cases(df)) # check for complete cares and return the result  
   
 # (!complete.cases(jnFouter))  
 retVal<-df[!complete.cases(df),] # check for complete cares and return the rows   
 }else{  
 #group\_vars<-quos(...) # format the arguments as list of formulas, not required for is.na()  
 retVal<-df[is.na(df[...]),] # return the s where NA is present  
 }  
 return(retVal)  
}  
# Demo example for using the function  
# ------------------------------------------  
# Calling the function - example  
fn.checkNullDS(jnFouter, blnChkDS=TRUE)

## roll\_no names maths science  
## 1 1 jack NA NA  
## 2 2 david NA NA  
## 3 3 peter NA NA  
## 5 5 james NA NA  
## 6 6 <NA> 67 91  
## 7 7 Sanjeev NA NA  
## 8 8 Anshul NA NA  
## 9 12 <NA> 92 92  
## 10 13 <NA> 76 88  
## 11 15 <NA> 90 92

fn.checkNullDS(jnFouter, blnChkDS=FALSE, c('names','maths'))

## roll\_no names maths science  
## 6 6 <NA> 67 91  
## 9 12 <NA> 92 92  
## 10 13 <NA> 76 88  
## 11 15 <NA> 90 92  
## NA NA <NA> NA NA  
## NA.1 NA <NA> NA NA  
## NA.2 NA <NA> NA NA  
## NA.3 NA <NA> NA NA  
## NA.4 NA <NA> NA NA  
## NA.5 NA <NA> NA NA

#10(b) - Need more clarity on problem statement  
# Demo example for using the function  
# ------------------------------------------

#Question - 11  
# ---------Function Definition --------------------------------------------------------------  
# 11(a) - remove duplicates  
poise <- function(x){  
 viz <- x[duplicated(x)]  
 print(viz)  
 return(x)  
}  
x <- c(8,9,9,7,5,4,4,3,2,6,6,2,1)  
poise(x)

## [1] 9 4 6 2

## [1] 8 9 9 7 5 4 4 3 2 6 6 2 1

# 11(b)  
Uni<-function(x){  
   
 #sun<-(unique(x)  
 sun<-n\_distinct(x)  
 return(sun)  
}  
x <- c(8,9,9,7,5,4,4,3,2,6,6,2,1)  
Uni(x)

## [1] 9

# 11(c)  
clip <- function(){  
 j <- "Planet"  
 w <- "World"  
 paste(j,w, sep="::")  
}  
clip()

## [1] "Planet::World"

# 11(d)  
Melt <- function(){  
 mat <- matrix(c(1:10), nrow=5,ncol=2);  
 mat  
 cat("Sum column wise :", apply(mat, 2, sum), "\n")  
 cat("Sum row wise :", apply(mat, 1, sum))  
}  
Melt()

## Sum column wise : 15 40   
## Sum row wise : 7 9 11 13 15

# 11(e) -- Need to explore hdfs   
# 11(f) -- Need to explore hdfs

#Question - 12  
# ---------Function Definition --------------------------------------------------------------  
seat <- function(){  
   
 #12(a) -- Please refer the example from preevious questions  
 #12(b) -- Need to explore hdfs  
   
 seat<-Seatbelts;  
   
 # 12(c) Rename the column names of a matrix object  
 colnames(seat) <- c("driverkilled", "Drivers", "Front",  
 "Rear", "KM/S", "PetrolPrice",  
 "vankill", "LAW")  
 print(seat)  
   
 # 12(d) Example of dropping a column name  
 seat <- seat[,(names(seat)) %in% c("law", "vankill")]  
 print(seat)  
   
 # 12(e) Examples of handling NA values  
 x <- c(NA, NaN)  
 print(is.na(x))  
 print(is.nan(x))  
 print(class(NA))  
 print(class(NaN))  
 print(class(NULL))  
   
 # 12(f)  
 vec <- c(1,2,3,4,5)  
 if(is.numeric(vec)){  
 print(TRUE)  
 }  
 ## g -- Compute number of unique combinations in a data frame grouped by certain columns  
 library(dplyr)  
 Orange  
 distinct(Orange, Tree)  
   
}  
# Calling the function  
seat()

## driverkilled Drivers Front Rear KM/S PetrolPrice vankill LAW  
## Jan 1969 107 1687 867 269 9059 0.10297181 12 0  
## Feb 1969 97 1508 825 265 7685 0.10236300 6 0  
## Mar 1969 102 1507 806 319 9963 0.10206249 12 0  
## Apr 1969 87 1385 814 407 10955 0.10087330 8 0  
## May 1969 119 1632 991 454 11823 0.10101967 10 0  
## Jun 1969 106 1511 945 427 12391 0.10058119 13 0  
## Jul 1969 110 1559 1004 522 13460 0.10377398 11 0  
## Aug 1969 106 1630 1091 536 14055 0.10407640 6 0  
## Sep 1969 107 1579 958 405 12106 0.10377398 10 0  
## Oct 1969 134 1653 850 437 11372 0.10302640 16 0  
## Nov 1969 147 2152 1109 434 9834 0.10273011 13 0  
## Dec 1969 180 2148 1113 437 9267 0.10199719 14 0  
## Jan 1970 125 1752 925 316 9130 0.10127456 14 0  
## Feb 1970 134 1765 903 311 8933 0.10070398 6 0  
## Mar 1970 110 1717 1006 351 11000 0.10013961 8 0  
## Apr 1970 102 1558 892 362 10733 0.09862110 11 0  
## May 1970 103 1575 990 486 12912 0.09834929 7 0  
## Jun 1970 111 1520 866 429 12926 0.09808018 13 0  
## Jul 1970 120 1805 1095 551 13990 0.09727921 13 0  
## Aug 1970 129 1800 1204 646 14926 0.09741062 11 0  
## Sep 1970 122 1719 1029 456 12900 0.09742524 11 0  
## Oct 1970 183 2008 1147 475 12034 0.09638063 14 0  
## Nov 1970 169 2242 1171 456 10643 0.09573896 16 0  
## Dec 1970 190 2478 1299 468 10742 0.09510631 14 0  
## Jan 1971 134 2030 944 356 10266 0.09673597 17 0  
## Feb 1971 108 1655 874 271 10281 0.09610922 16 0  
## Mar 1971 104 1693 840 354 11527 0.09536725 15 0  
## Apr 1971 117 1623 893 427 12281 0.09470959 13 0  
## May 1971 157 1805 1007 465 13587 0.09411762 13 0  
## Jun 1971 148 1746 973 440 13049 0.09353215 15 0  
## Jul 1971 130 1795 1097 539 16055 0.09295405 12 0  
## Aug 1971 140 1926 1194 646 15220 0.09283979 6 0  
## Sep 1971 136 1619 988 457 13824 0.09272474 9 0  
## Oct 1971 140 1992 1077 446 12729 0.09226965 13 0  
## Nov 1971 187 2233 1045 402 11467 0.09170669 14 0  
## Dec 1971 150 2192 1115 441 11351 0.09126207 15 0  
## Jan 1972 159 2080 1005 359 10803 0.09071160 14 0  
## Feb 1972 143 1768 857 334 10548 0.09027633 3 0  
## Mar 1972 114 1835 879 312 12368 0.08995192 12 0  
## Apr 1972 127 1569 887 427 13311 0.08909964 13 0  
## May 1972 159 1976 1075 434 13885 0.08867919 12 0  
## Jun 1972 156 1853 1121 486 14088 0.08815929 8 0  
## Jul 1972 138 1965 1190 569 16932 0.08890206 8 0  
## Aug 1972 120 1689 1058 523 16164 0.08818133 15 0  
## Sep 1972 117 1778 939 418 14883 0.08894029 8 0  
## Oct 1972 170 1976 1074 452 13532 0.08772661 5 0  
## Nov 1972 168 2397 1089 462 12220 0.08742885 17 0  
## Dec 1972 198 2654 1208 497 12025 0.08703543 14 0  
## Jan 1973 144 2097 903 354 11692 0.08644992 13 0  
## Feb 1973 146 1963 916 347 11081 0.08587264 5 0  
## Mar 1973 109 1677 787 276 13745 0.08539822 8 0  
## Apr 1973 131 1941 1114 472 14382 0.08382198 5 0  
## May 1973 151 2003 1014 487 14391 0.08459078 12 0  
## Jun 1973 140 1813 1022 505 15597 0.08413690 11 0  
## Jul 1973 153 2012 1114 619 16834 0.08377841 13 0  
## Aug 1973 140 1912 1132 640 17282 0.08351074 15 0  
## Sep 1973 161 2084 1111 559 15779 0.08280639 11 0  
## Oct 1973 168 2080 1008 453 13946 0.08117889 11 0  
## Nov 1973 152 2118 916 418 12701 0.08285361 10 0  
## Dec 1973 136 2150 992 419 10431 0.09419012 13 0  
## Jan 1974 113 1608 731 262 11616 0.09239984 8 0  
## Feb 1974 100 1503 665 299 10808 0.10816148 6 0  
## Mar 1974 103 1548 724 303 12421 0.10721169 8 0  
## Apr 1974 103 1382 744 401 13605 0.11404297 14 0  
## May 1974 121 1731 910 413 14455 0.11245412 12 0  
## Jun 1974 134 1798 883 426 15019 0.11131625 14 0  
## Jul 1974 133 1779 900 516 15662 0.11030125 13 0  
## Aug 1974 129 1887 1057 600 16745 0.10819718 9 0  
## Sep 1974 144 2004 1076 459 14717 0.10702744 4 0  
## Oct 1974 154 2077 919 443 13756 0.10494698 13 0  
## Nov 1974 156 2092 920 412 12531 0.11935775 6 0  
## Dec 1974 163 2051 953 400 12568 0.11762190 15 0  
## Jan 1975 122 1577 664 278 11249 0.13302742 12 0  
## Feb 1975 92 1356 607 302 11096 0.13084524 16 0  
## Mar 1975 117 1652 777 381 12637 0.12831848 7 0  
## Apr 1975 95 1382 633 279 13018 0.12354745 12 0  
## May 1975 96 1519 791 442 15005 0.11858681 10 0  
## Jun 1975 108 1421 790 409 15235 0.11633748 9 0  
## Jul 1975 108 1442 803 416 15552 0.11516148 9 0  
## Aug 1975 106 1543 884 511 16905 0.11450120 6 0  
## Sep 1975 140 1656 769 393 14776 0.11352298 7 0  
## Oct 1975 114 1561 732 345 14104 0.11193018 13 0  
## Nov 1975 158 1905 859 391 12854 0.11061053 14 0  
## Dec 1975 161 2199 994 470 12956 0.11527439 13 0  
## Jan 1976 102 1473 704 266 12177 0.11379349 14 0  
## Feb 1976 127 1655 684 312 11918 0.11234958 11 0  
## Mar 1976 125 1407 671 300 13517 0.11175347 11 0  
## Apr 1976 101 1395 643 373 14417 0.10964252 10 0  
## May 1976 97 1530 771 412 15911 0.10844090 4 0  
## Jun 1976 112 1309 644 322 15589 0.10788494 8 0  
## Jul 1976 112 1526 828 458 16543 0.10908477 9 0  
## Aug 1976 113 1327 748 427 17925 0.10757145 10 0  
## Sep 1976 108 1627 767 346 15406 0.10616402 10 0  
## Oct 1976 128 1748 825 421 14601 0.10630000 5 0  
## Nov 1976 154 1958 810 344 13107 0.10482531 13 0  
## Dec 1976 162 2274 986 370 12268 0.10345175 12 0  
## Jan 1977 112 1648 714 291 11972 0.10144992 10 0  
## Feb 1977 79 1401 567 224 12028 0.10040232 9 0  
## Mar 1977 82 1411 616 266 14033 0.09886203 7 0  
## Apr 1977 127 1403 678 338 14244 0.10249615 5 0  
## May 1977 108 1394 742 298 15287 0.10302743 10 0  
## Jun 1977 110 1520 840 386 16954 0.10217891 5 0  
## Jul 1977 123 1528 888 479 17361 0.09983664 6 0  
## Aug 1977 103 1643 852 473 17694 0.09263669 8 0  
## Sep 1977 97 1515 774 332 16222 0.09181496 6 0  
## Oct 1977 140 1685 831 391 14969 0.09072430 12 0  
## Nov 1977 165 2000 889 370 13624 0.09002121 15 0  
## Dec 1977 183 2215 1046 431 13842 0.08933071 7 0  
## Jan 1978 148 1956 889 366 12387 0.08844273 14 0  
## Feb 1978 111 1462 626 250 11608 0.08835257 4 0  
## Mar 1978 116 1563 808 355 15021 0.08675736 10 0  
## Apr 1978 115 1459 746 304 14834 0.08499524 8 0  
## May 1978 100 1446 754 379 16565 0.08456794 7 0  
## Jun 1978 106 1622 865 440 16882 0.08443190 11 0  
## Jul 1978 134 1657 980 500 18012 0.08435088 3 0  
## Aug 1978 125 1638 959 511 18855 0.08360098 5 0  
## Sep 1978 117 1643 856 384 17243 0.08341726 11 0  
## Oct 1978 122 1683 798 366 16045 0.08274514 10 0  
## Nov 1978 153 2050 942 432 14745 0.08523527 10 0  
## Dec 1978 178 2262 1010 390 13726 0.08477030 7 0  
## Jan 1979 114 1813 796 306 11196 0.08445892 10 0  
## Feb 1979 94 1445 643 232 12105 0.08535212 11 0  
## Mar 1979 128 1762 794 342 14723 0.08755921 9 0  
## Apr 1979 119 1461 750 329 15582 0.09038292 7 0  
## May 1979 111 1556 809 394 16863 0.09078329 8 0  
## Jun 1979 110 1431 716 355 16758 0.10874278 13 0  
## Jul 1979 114 1427 851 385 17434 0.11414223 8 0  
## Aug 1979 118 1554 931 463 18359 0.11299293 5 0  
## Sep 1979 115 1645 834 453 17189 0.11132071 8 0  
## Oct 1979 132 1653 762 373 16909 0.10912623 7 0  
## Nov 1979 153 2016 880 401 15380 0.10769846 12 0  
## Dec 1979 171 2207 1077 466 15161 0.10760157 10 0  
## Jan 1980 115 1665 748 306 14027 0.10377502 7 0  
## Feb 1980 95 1361 593 263 14478 0.10711417 4 0  
## Mar 1980 92 1506 720 323 16155 0.10737477 10 0  
## Apr 1980 100 1360 646 310 16585 0.11169537 4 0  
## May 1980 95 1453 765 424 18117 0.11063818 8 0  
## Jun 1980 114 1522 820 403 17552 0.11185521 8 0  
## Jul 1980 102 1460 807 406 18299 0.10974234 7 0  
## Aug 1980 104 1552 885 466 19361 0.10819393 10 0  
## Sep 1980 132 1548 803 381 17924 0.10625536 8 0  
## Oct 1980 136 1827 860 369 17872 0.10419303 14 0  
## Nov 1980 117 1737 825 378 16058 0.10193397 8 0  
## Dec 1980 137 1941 911 392 15746 0.10279382 9 0  
## Jan 1981 111 1474 704 284 15226 0.10476034 8 0  
## Feb 1981 106 1458 691 316 14932 0.10400254 6 0  
## Mar 1981 98 1542 688 321 16846 0.11665552 7 0  
## Apr 1981 84 1404 714 358 16854 0.11516148 6 0  
## May 1981 94 1522 814 378 18146 0.11298954 5 0  
## Jun 1981 105 1385 736 382 17559 0.11386064 4 0  
## Jul 1981 123 1641 876 433 18655 0.11911808 5 0  
## Aug 1981 109 1510 829 506 19453 0.12448999 10 0  
## Sep 1981 130 1681 818 428 17923 0.12322295 7 0  
## Oct 1981 153 1938 942 479 17915 0.12067793 10 0  
## Nov 1981 134 1868 782 370 16496 0.12104898 12 0  
## Dec 1981 99 1726 823 349 13544 0.11696857 7 0  
## Jan 1982 115 1456 595 238 13601 0.11275026 4 0  
## Feb 1982 104 1445 673 285 15667 0.10807931 5 0  
## Mar 1982 131 1456 660 324 17358 0.10883852 6 0  
## Apr 1982 108 1365 676 346 18112 0.11129177 4 0  
## May 1982 103 1487 755 410 18581 0.11130401 4 0  
## Jun 1982 115 1558 815 411 18759 0.11545436 8 0  
## Jul 1982 122 1488 867 496 20668 0.11476830 8 0  
## Aug 1982 122 1684 933 534 21040 0.11720743 3 0  
## Sep 1982 125 1594 798 396 18993 0.11907640 7 0  
## Oct 1982 137 1850 950 470 18668 0.11796586 12 0  
## Nov 1982 138 1998 825 385 16768 0.11744913 2 0  
## Dec 1982 152 2079 911 411 16551 0.11698846 7 0  
## Jan 1983 120 1494 619 281 16231 0.11261054 8 0  
## Feb 1983 95 1057 426 300 15511 0.11365702 3 1  
## Mar 1983 100 1218 475 318 18308 0.11314445 2 1  
## Apr 1983 89 1168 556 391 17793 0.11849553 6 1  
## May 1983 82 1236 559 398 19205 0.11796940 3 1  
## Jun 1983 89 1076 483 337 19162 0.11768661 7 1  
## Jul 1983 60 1174 587 477 20997 0.12005924 6 1  
## Aug 1983 84 1139 615 422 20705 0.11943775 8 1  
## Sep 1983 113 1427 618 495 18759 0.11888127 8 1  
## Oct 1983 126 1487 662 471 19240 0.11846236 4 1  
## Nov 1983 122 1483 519 368 17504 0.11801660 3 1  
## Dec 1983 118 1513 585 345 16591 0.11770662 5 1  
## Jan 1984 92 1357 483 296 16224 0.11777609 5 1  
## Feb 1984 86 1165 434 319 16670 0.11479699 3 1  
## Mar 1984 81 1282 513 349 18539 0.11573525 4 1  
## Apr 1984 84 1110 548 375 19759 0.11535626 3 1  
## May 1984 87 1297 586 441 19584 0.11481536 6 1  
## Jun 1984 90 1185 522 465 19976 0.11477748 6 1  
## Jul 1984 79 1222 601 472 21486 0.11493598 7 1  
## Aug 1984 96 1284 644 521 21626 0.11479699 5 1  
## Sep 1984 122 1444 643 429 20195 0.11409316 7 1  
## Oct 1984 120 1575 641 408 19928 0.11646552 7 1  
## Nov 1984 137 1737 711 490 18564 0.11602611 4 1  
## Dec 1984 154 1763 721 491 18149 0.11606673 7 1  
##   
## Jan 1969  
## Feb 1969  
## Mar 1969  
## Apr 1969  
## May 1969  
## Jun 1969  
## Jul 1969  
## Aug 1969  
## Sep 1969  
## Oct 1969  
## Nov 1969  
## Dec 1969  
## Jan 1970  
## Feb 1970  
## Mar 1970  
## Apr 1970  
## May 1970  
## Jun 1970  
## Jul 1970  
## Aug 1970  
## Sep 1970  
## Oct 1970  
## Nov 1970  
## Dec 1970  
## Jan 1971  
## Feb 1971  
## Mar 1971  
## Apr 1971  
## May 1971  
## Jun 1971  
## Jul 1971  
## Aug 1971  
## Sep 1971  
## Oct 1971  
## Nov 1971  
## Dec 1971  
## Jan 1972  
## Feb 1972  
## Mar 1972  
## Apr 1972  
## May 1972  
## Jun 1972  
## Jul 1972  
## Aug 1972  
## Sep 1972  
## Oct 1972  
## Nov 1972  
## Dec 1972  
## Jan 1973  
## Feb 1973  
## Mar 1973  
## Apr 1973  
## May 1973  
## Jun 1973  
## Jul 1973  
## Aug 1973  
## Sep 1973  
## Oct 1973  
## Nov 1973  
## Dec 1973  
## Jan 1974  
## Feb 1974  
## Mar 1974  
## Apr 1974  
## May 1974  
## Jun 1974  
## Jul 1974  
## Aug 1974  
## Sep 1974  
## Oct 1974  
## Nov 1974  
## Dec 1974  
## Jan 1975  
## Feb 1975  
## Mar 1975  
## Apr 1975  
## May 1975  
## Jun 1975  
## Jul 1975  
## Aug 1975  
## Sep 1975  
## Oct 1975  
## Nov 1975  
## Dec 1975  
## Jan 1976  
## Feb 1976  
## Mar 1976  
## Apr 1976  
## May 1976  
## Jun 1976  
## Jul 1976  
## Aug 1976  
## Sep 1976  
## Oct 1976  
## Nov 1976  
## Dec 1976  
## Jan 1977  
## Feb 1977  
## Mar 1977  
## Apr 1977  
## May 1977  
## Jun 1977  
## Jul 1977  
## Aug 1977  
## Sep 1977  
## Oct 1977  
## Nov 1977  
## Dec 1977  
## Jan 1978  
## Feb 1978  
## Mar 1978  
## Apr 1978  
## May 1978  
## Jun 1978  
## Jul 1978  
## Aug 1978  
## Sep 1978  
## Oct 1978  
## Nov 1978  
## Dec 1978  
## Jan 1979  
## Feb 1979  
## Mar 1979  
## Apr 1979  
## May 1979  
## Jun 1979  
## Jul 1979  
## Aug 1979  
## Sep 1979  
## Oct 1979  
## Nov 1979  
## Dec 1979  
## Jan 1980  
## Feb 1980  
## Mar 1980  
## Apr 1980  
## May 1980  
## Jun 1980  
## Jul 1980  
## Aug 1980  
## Sep 1980  
## Oct 1980  
## Nov 1980  
## Dec 1980  
## Jan 1981  
## Feb 1981  
## Mar 1981  
## Apr 1981  
## May 1981  
## Jun 1981  
## Jul 1981  
## Aug 1981  
## Sep 1981  
## Oct 1981  
## Nov 1981  
## Dec 1981  
## Jan 1982  
## Feb 1982  
## Mar 1982  
## Apr 1982  
## May 1982  
## Jun 1982  
## Jul 1982  
## Aug 1982  
## Sep 1982  
## Oct 1982  
## Nov 1982  
## Dec 1982  
## Jan 1983  
## Feb 1983  
## Mar 1983  
## Apr 1983  
## May 1983  
## Jun 1983  
## Jul 1983  
## Aug 1983  
## Sep 1983  
## Oct 1983  
## Nov 1983  
## Dec 1983  
## Jan 1984  
## Feb 1984  
## Mar 1984  
## Apr 1984  
## May 1984  
## Jun 1984  
## Jul 1984  
## Aug 1984  
## Sep 1984  
## Oct 1984  
## Nov 1984  
## Dec 1984  
## [1] TRUE TRUE  
## [1] FALSE TRUE  
## [1] "logical"  
## [1] "numeric"  
## [1] "NULL"  
## [1] TRUE

## Tree  
## 1 1  
## 2 2  
## 3 3  
## 4 4  
## 5 5

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

summary(cars)

## speed dist   
## Min. : 4.0 Min. : 2.00   
## 1st Qu.:12.0 1st Qu.: 26.00   
## Median :15.0 Median : 36.00   
## Mean :15.4 Mean : 42.98   
## 3rd Qu.:19.0 3rd Qu.: 56.00   
## Max. :25.0 Max. :120.00

## Including Plots

You can also embed plots, for example:



Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.