```r
###### COURSE: Basics of R programming language for statistical analysis #######

#Instructor: Marina FERENT-PIPAS
#email: marinaferent@gmail.com
#Multicultural Business Institute | mcb-institute.org

#_______ CHAPTER 2: CONTROL STRUCTURES AND FUNCTIONS | Statistical measures _____#
########## MEETING 05: RECAP REPETITIVE STRUCTURES AND CONDITIONAL STATEMENTS |
########## Challenge: Median values| 07.09.2021 #############

#_____________________________________________________________________________#
#################### GOOD PROGRAMMING PRACTICES OF THE MEETING:
    ##1. DESIGN: Before coding a section - write down each step in order
    ##2. INDENTATION: Showcase the begining of a section, the body and the end of it
    through the right indentation.
            ##Anything that subordinates to a line is TABed once from that line.

#_____________________________________________________________________________#


#EXERCISE 1: Import the <<Campus crimes.csv>> into R. Save it as campusData.

setwd("E:/Multicultural Business Institute/R/Meeting 5") #sets the working directory.
all the files exported/imported following this line, are exported to/imported from this
location
campusData=read.csv("Campus crimes.csv")                ##imports the campusData from
a .csv file called Campus crimes

    #alternatively:

campusData=read.csv("E:/Multicultural Business Institute/R/Meeting 5/Campus
crimes.csv")

View(campusData)    #visualizes the campusData as a table
##We visualized together the campusData and observed that the names of the variables
were not correctly imported, as such we changed them:
colnames(campusData)

#to change the name of the first column:
colnames(campusData)[1]="enrollment" #colnames(campusData) is a vector of length 4 (4
elements). we change here the first element of the vector
    #alternatively:
colnames(campusData)=c("enrollment", "private", "police", "crime")

#EXERCISE 2: Compute the median value for enrollments.
median(campusData$enrollment)
    #alternatively:
median(campusData[,1])      #number of enrollments is in column 1


    ##alternatively - challenge Opt1:
#Step 1: Order ascendingly the first column:
campusData[,1]=sort(campusData[,1])     #ordonam crescator variabila de pe coloana 1
din campusData (i.e. enrollments)
n=length(campusData[,1])                #sample size (lungimea vectorului) = cate
colegii sunt in esantion

#Step 2: We check if N div with 2 or not
#Step 3: ##if N div by 2 -> median= mean(X_n/2, X_n/2+1)
         ##if N is NOT by 2 -> median=X_{n/2+1}=X[as.integer(n/2+1)]=X[trunc(n/2+1)]

if (n%%2!=0)        #daca restul impartirii lui n la 2 nu este 0 (if n is not div by 2)
{
    medianValue=campusData[trunc(n/2+1),1] #medianValue =elementul (din sirul ordonat
    crescator) al parte intreaga n/2+1 lea - rotunjire in sus
} else {            #daca restul impartirii lui n la 2 este 0 (if n is div by 2)
    medianValue=(campusData[n/2,1]+campusData[n/2+1,1])/2  #medianValue = medie din:
    elementul (din sirul ordonat crescator) al n/2 lea si urmatorul element
}
```

```r
medianValue

    ##alternatively - challenge Opt2:

if (n%%2!=0)        #daca restul impartirii lui n la 2 nu este 0 (if n is not div by 2)
{
    print(campusData[trunc(n/2+1),1]) #medianValue =elementul (din sirul ordonat crescator) al parte intreaga n/2+1 lea - rotunjire in sus
} else {            #daca restul impartirii lui n la 2 este 0 (if n is div by 2)
    print((campusData[n/2,1]+campusData[n/2+1,1])/2)  #medianValue = medie din: elementul (din sirul ordonat crescator) al n/2 lea si urmatorul element
}

    ##alternatively - challenge Opt3:

if (n%%2!=0)        #daca restul impartirii lui n la 2 nu este 0 (if n is not div by 2)
{
    medianValue=campusData[as.integer(n/2+1),1] #medianValue =elementul (din sirul ordonat crescator) al parte intreaga n/2+1 lea - rotunjire in sus
} else {            #daca restul impartirii lui n la 2 este 0 (if n is div by 2)
    medianValue=mean(campusData[n/2,1],campusData[n/2+1,1])  #medianValue = medie din: elementul (din sirul ordonat crescator) al n/2 lea si urmatorul element
}
medianValue


#EXERCISE 3: Compute the median value for all variables in campusData.
library("matrixStats")  #loading the package containing colMedians function
colMedians(campusData)  #calculeaza valorile mediane pentru toate variabilele din campusData

    ##alternatively - challenge Opt 1 - printing the median values:

for (i in 1:ncol(campusData))           #i takes values from 1 to number of columns of <<campusData>>
{                                       #beginning of for loop
  campusData[,i]=sort(campusData[,i])   #ordonam crescator variabila de pe coloana i din campusData
  n=length(campusData[,i])              #sample size (lungimea vectorului) = cate colegii sunt in esantion
  if (n%%2!=0)                          #daca restul impartirii lui n la 2 nu este 0 (if n is not div by 2)
  {
    print(campusData[trunc(n/2+1),i])   #returneaza elementul (din sirul ordonat crescator) al parte intreaga n/2+1 lea - rotunjire in sus
  } else {                              #daca restul impartirii lui n la 2 este 0 (if n is div by 2)
    print((campusData[n/2,i]+campusData[n/2+1,i])/2)    #returneaza medie din: elementul (din sirul ordonat crescator) al n/2 lea si urmatorul element
  }
}                                       #closing for loop

    ##alternatively - challenge Opt 2 - storing the median values in a matrix _ opt 1:

medianValues=matrix(NA, nrow=ncol(campusData), ncol=1) #a matrix full of "NA", with 4 rows and 1 column
rownames(medianValues)=colnames(campusData)             #variable names are stored as rownames
colnames(medianValues)="Median"                         #the name of the column is "Median"
View(medianValues)                                      #visualizing the matrix

for (i in 1:ncol(campusData))                           #i takes values from 1 to number of columns of <<campusData>>
{
    campusData[,i]=sort(campusData[,i])                 #ordonam crescator variabila de pe coloana i din campusData
    n=length(campusData[,i])                            #sample size (lungimea vectorului) = cate colegii sunt in esantion
```

```r
    if (n%%2!=0)                                    #daca restul impartirii lui n la 2 nu este 0 (if n is not div by 2)
    {
        medianValue=campusData[trunc(n/2+1),i]      #we store in medianValue= elementul (din sirul ordonat crescator) al parte intreaga n/2+1 lea - rotunjire in sus
    } else {                                        #daca restul impartirii lui n la 2 este 0 (if n is div by 2)
        medianValue=(campusData[n/2,i]+campusData[n/2+1,i])/2   #we store in medianValue=medie din: elementul (din sirul ordonat crescator) al n/2 lea si urmatorul element
    }
    medianValues[i]=medianValue                     #we store in row i, column 1 of matrix medianValues the current value of medianValue
}

medianValues
View(medianValues)


#EXERCISE_POINT_1: Save the matrix medianValues as "campusCrimes_medianValues.csv" on your computer.

#EXERCISE_POINT_2: Debug the following piece of code:
medianValues=matrix(NA, nrow=ncol(campusData), ncol=2)
rownames(medianValues)=colnames(campusData)
colnames(medianValues)="Median"
View(medianValues)

#EXERCISE_POINT_3: Add a second column to medianValues in which you store the interpretation of each median value.
    ##e.g. medianValues[1,2]= 50% of the colleges had at most 11 990 students enrolled
    ##HINT: check the paste0 function -> see for e.g. rBasics_Meeting2.r -> lines 100-101
#Name the new column: Interpretation


    ##alternatively - challenge Opt 3 - storing the median values in a matrix _ opt 2:

#Step 1: Define a medianValues matrix in which to store the name of the variables in column one and the respective median value in column 2
#We create an empty matrix of 4 rows and 2 columns:
    ##each cell in the matrix has value NA
    ##the matrix has the same number of rows as many variables as campusData has
    ##we have 2 columns: in column 1 we will store the name of the variables and in column 2 we will store the median of each variable

medianValues=matrix(NA, nrow=ncol(campusData), ncol=2) #a matrix full of "NA", with 4 rows and 2 columns
View(medianValues)

#Step 2: We write in each cell of our <<medianValues>> matrix step by step:
##first: i=1, we write in <<medianValues>> row 1 all the information regarding <<campusData>> column 1 (variable name, variable median)
##then: i=2, we write in <<medianValues>> row 2 all the information regarding <<campusData>> column 2 (variable name, variable median)
##then: i=3, we write in <<medianValues>> row 3 all the information regarding <<campusData>> column 3 (variable name, variable median)
##then: i=4, we write in <<medianValues>> row 4 all the information regarding <<campusData>> column 4 (variable name, variable median)
##then: we finished, since i takes values from 1 to 4

for (i in 1:ncol(campusData)) #i takes values from 1 to number of columns of <<campusData>>
{                             #beggining of for loop
    medianValues[i,1]=colnames(campusData)[i]     #in matrix <<medianValues>>, row i, column 1 we store the name of the variable in <<campusData>> column i
    sortedCampusData=sort(campusData[,i])         #ordonam crescator variabila de pe coloana i din campusData
    if(nrow(campusData)%%2==0)                    #daca restul impartirii lui n la 2
```

```
      este 0
160   {                       #beggining of if loop
161       medianValues[i,2]=mean(sortedCampusData[nrow(campusData)/2],
          sortedCampusData[nrow(campusData)/2 + 1]) #medianValues = medie din: elementul
          (din sirul ordonat crescator) al n/2 lea si urmatorul element
162   } else {                #ending of if loop, beggining of else loop: #daca restul
      impartirii lui n la 2 nu este 0
163       medianValues[i,2]=sortedCampusData[as.integer(nrow(campusData)/2 +1)]
          #medianValues =elementul (din sirul ordonat crescator) al parte intreaga n/2+1
          lea - rotunjire in sus
164   } #ending of else loop
165 }#ending of for loop
166
167 medianValues
168 View(medianValues)
```