

**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

|  |  |  |
| --- | --- | --- |
| **Course Code: ITA04** | **Course Name: Statistics with R Programming** | |
| **Branch: CSE** |  | **Academic Year: 2021-2020 (Odd)** |
| **Date of Exam: 13/7/2021** | **Max. Marks: 20M** | **Time: 1 hour** |

**ANSWER ALL THE QUESTIONS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **CLASS TEST 1** | **MARKS** | **COs** | **BT LEVEL** |
| 1 | What are the different forms of data types and how to test the data type in R? Give one example for each. | 10 | CO1 | K3 |
| 2 | 1. Write a R program to create a sequence of numbers from 20 to 50 and find the mean of numbers from 20 to 60 and sum of numbers from 51 to 91. 2. Write a R program to create an array with three columns, three rows, and two "tables", taking two vectors as input to the array. Print the array. | 10 | CO1 | K3 |

**CLASS TEST 1 -ANSWER**

1.

Numeric:

# Numeric data type

x <- 10.5

print(typeof(x)) # Output: "double"

Character:

# Character data type

name <- "John"

print(typeof(name)) # Output: "character"

print(is.character(name)) # Output: TRUE

Logical:

# Logical data type

is\_valid <- TRUE

print(typeof(is\_valid)) # Output: "logical"

print(is.logical(is\_valid)) # Output: TRUE

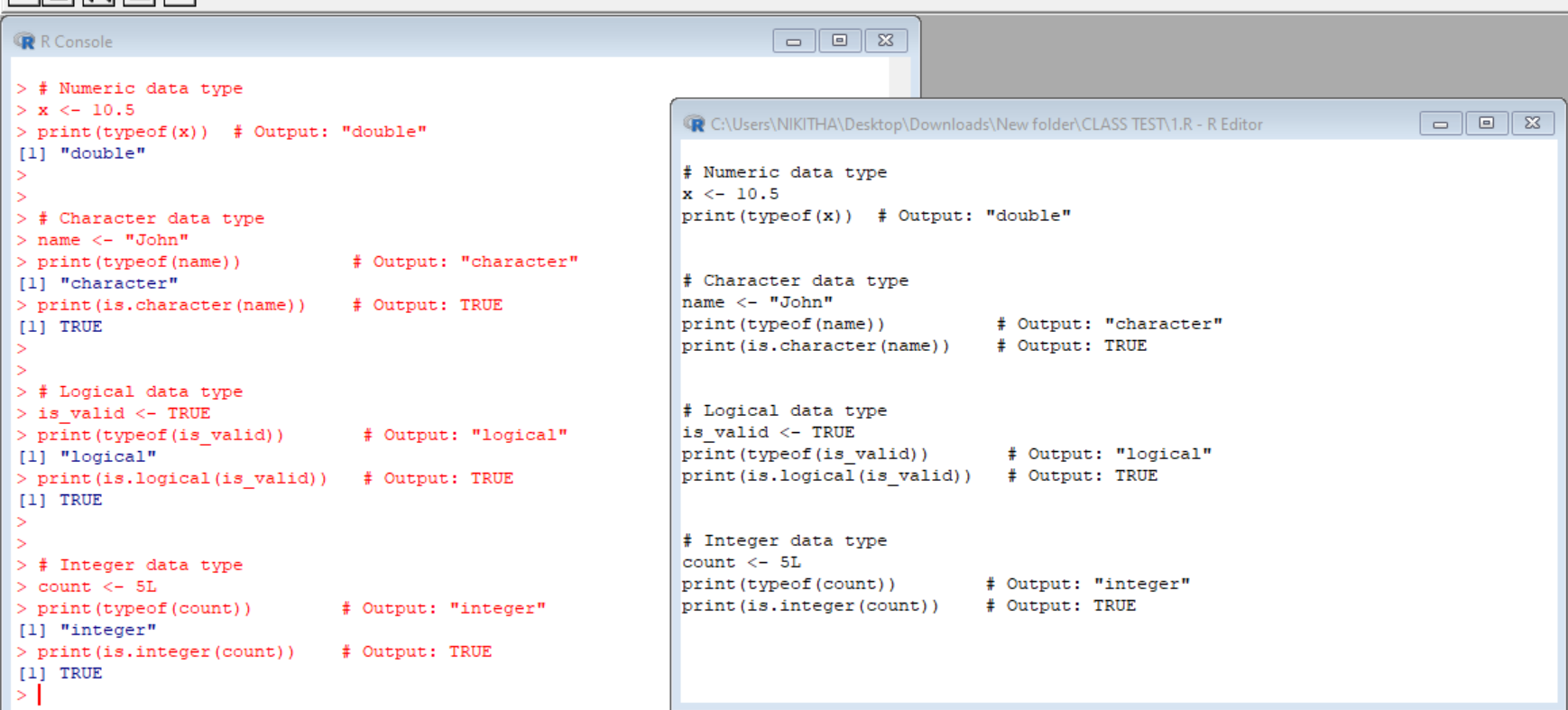
Integer:

# Integer data type

count <- 5L

print(typeof(count)) # Output: "integer"

print(is.integer(count)) # Output: TRUE



2.(i) # Create a sequence of numbers from 20 to 50

sequence <- seq(20, 50)

# Calculate the mean of numbers from 20 to 60

mean\_20\_to\_60 <- mean(seq(20, 60))

# Calculate the sum of numbers from 51 to 91

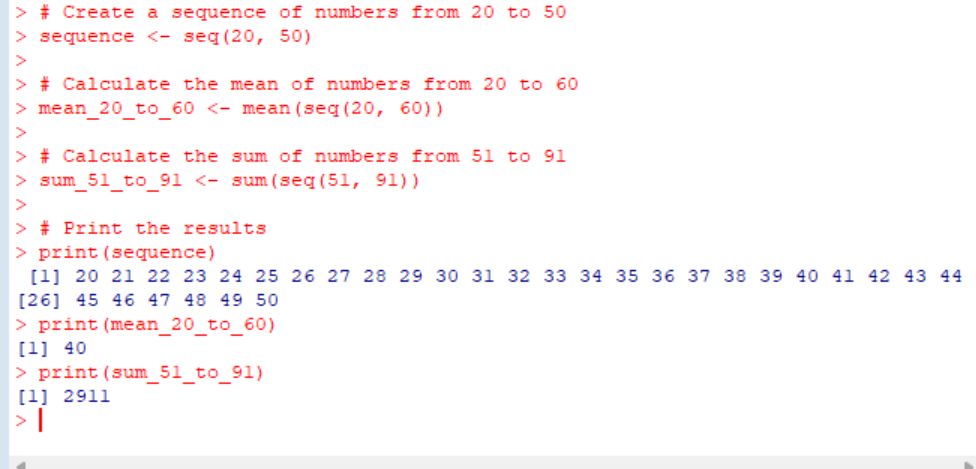
sum\_51\_to\_91 <- sum(seq(51, 91))

# Print the results

print(sequence)

print(mean\_20\_to\_60)

print(sum\_51\_to\_91)



(ii) # Create two vectors

vector1 <- c(1, 2, 3)

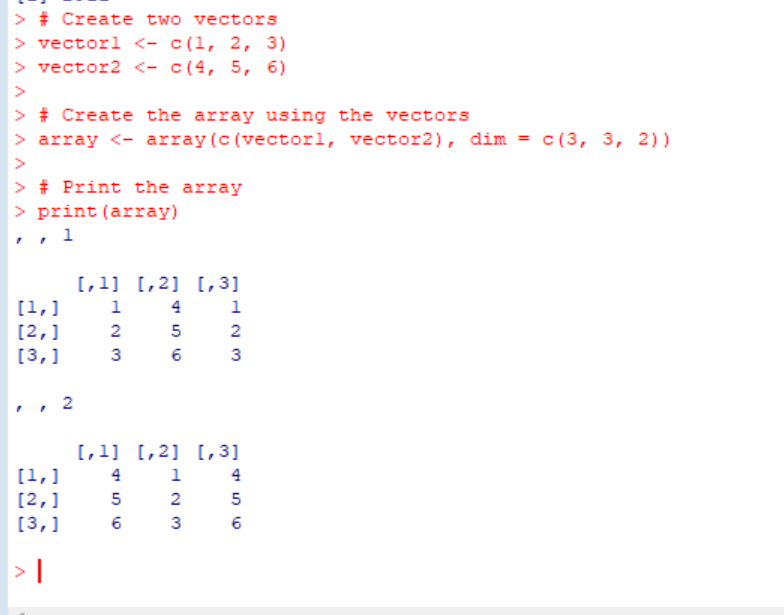
vector2 <- c(4, 5, 6)

# Create the array using the vectors

array <- array(c(vector1, vector2), dim = c(3, 3, 2))

# Print the array

print(array)





**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

|  |  |  |
| --- | --- | --- |
| **Course Code: ITA04** | **Course Name: Statistics with R Programming** | |
| **Branch: CSE** |  | **Academic Year: 2021-2020 (Odd)** |
| **Date of Exam: 30/7/21** | **Max. Marks: 20M** | **Time: 1 hour** |

**ANSWER ALL THE QUESTIONS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **CLASS TEST 2** | **MARKS** | **COs** | **BT LEVEL** |
| 1 | Explain the various types of control statements in R programming. Give examples. | 10 | CO2 | K3 |
| 2 | Write R code function to generate first n terms of a Fibonacci series | 10 | CO2 | K3 |

**CLASS TEST 2-ANSWERS**

**1.**

**If-else Statement:**

x <- 10

if (x > 5) {

print("x is greater than 5")

} else {

print("x is less than or equal to 5")

}

**For Loop:**

sum <- 0

for (i in 1:5) {

sum <- sum + i

}

print(sum) # Output: 15

**While Loop:**

i <- 1

while (i <= 5) {

print(i)

i <- i + 1

}

**Repeat loop:**

while (TRUE) {

num <- runif(1) # Generate a random number

if (num > 0.8) {

break # Exit the loop if the condition is met

}

print(num)

}

**Switch Statement:**

day <- "Tuesday"

switch(day,

"Monday" = print("Today is Monday."),

"Tuesday" = print("Today is Tuesday."),

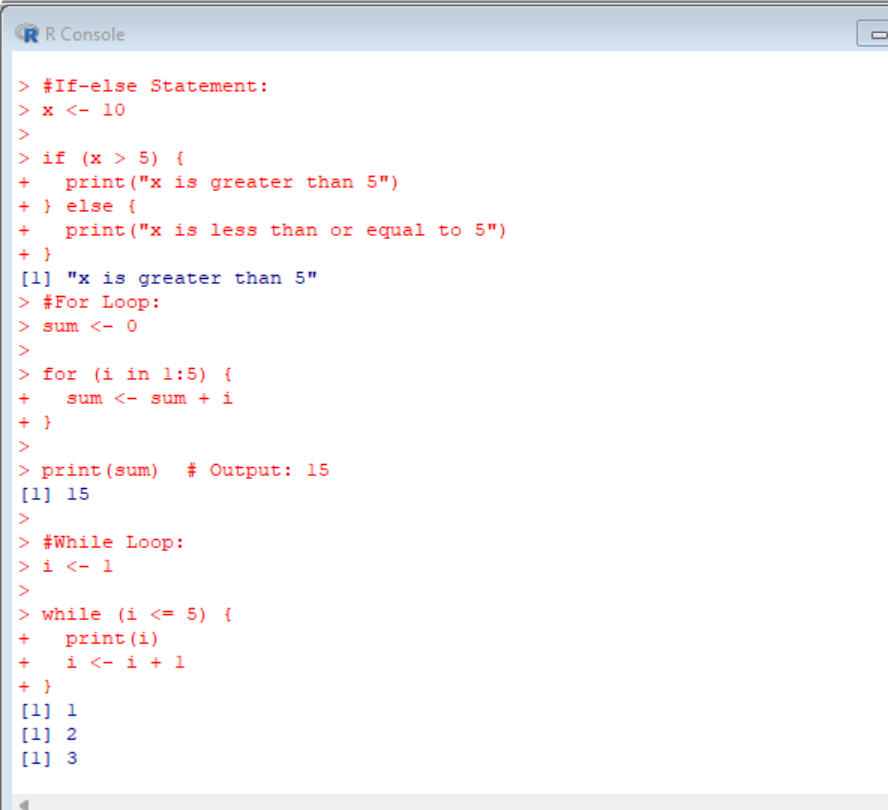
"Wednesday" = print("Today is Wednesday."),

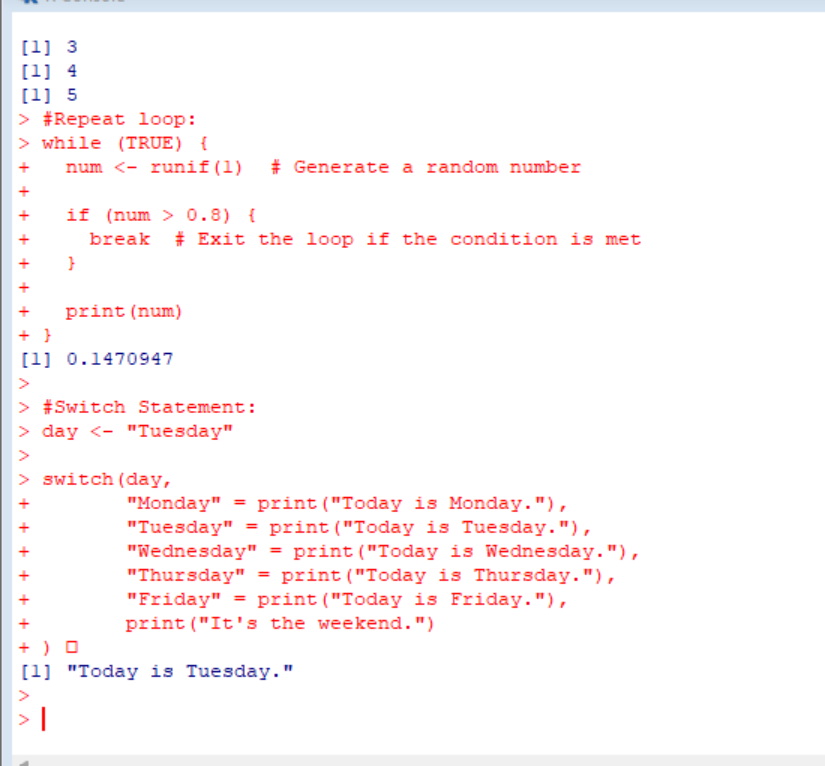
"Thursday" = print("Today is Thursday."),

"Friday" = print("Today is Friday."),

print("It's the weekend.")

)

****

****

**2.** generateFibonacci <- function(n) {

fibonacci <- c(0, 1) # Initialize the Fibonacci sequence with the first two terms

if (n <= 2) {

return(fibonacci[1:n]) # Return the requested number of terms if n is less than or equal to 2

}

for (i in 3:n) {

nextTerm <- fibonacci[i-1] + fibonacci[i-2] # Calculate the next Fibonacci term

fibonacci <- c(fibonacci, nextTerm) # Append the next term to the sequence

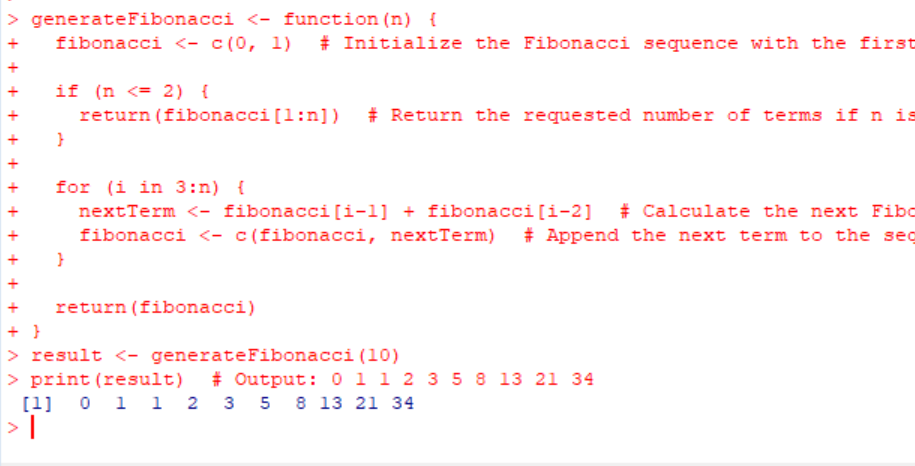
}

return(fibonacci)

}

result <- generateFibonacci(10)

print(result) # Output: 0 1 1 2 3 5 8 13 21 34





**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

|  |  |  |
| --- | --- | --- |
| **Course Code: ITA04** | **Course Name: Statistics with R Programming** | |
| **Branch: CSE** |  | **Academic Year: 2021-2020 (Odd)** |
| **Date of Exam: 12/8/2021** | **Max. Marks: 20M** | **Time: 1 hour** |

**ANSWER ALL THE QUESTIONS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **CLASS TEST 3** | **MARKS** | **COs** | **BT LEVEL** |
| 1 | Write R code to perform the operations with following data exam\_data = data.frame( name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura'), score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5), attempts = c(1, 3, 2, 3, 2, 3, 1, 1), qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no'))  (a) Create a Data frame  (b) Extract score and attempts column by their names  (c) Extract name and qualify column by their postions  (d) To extract 3rd and 5th rows with 1st and 3rd columns  (e) To extract ‘James’ details | 10 | CO3 | K3 |
| 2 | Create Data Frame for following data n <- c(1, 1, 2, 2), time <- c(1, 2, 1, 2) , x <- c(6, 3, 2, 5), y <- c(1, 4, 6, 9)  (i)Write a R code to melt the data and display as a long-format data?  (ii) Write a R code, use cast function appropriately to compute the average of x and y with respect to “time”. | 10 | CO3 | K3 |

**CLASS TEST 3 -ANSWERS**

**1.** # Create the exam\_data data frame

exam\_data <- data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no')

)

# (a) Create a Data frame

print(exam\_data)

# (b) Extract score and attempts column by their names

score\_attempts <- exam\_data[c("score", "attempts")]

print(score\_attempts)

# (c) Extract name and qualify column by their positions

name\_qualify <- exam\_data[, c(1, 4)]

print(name\_qualify)

# (d) To extract 3rd and 5th rows with 1st and 3rd columns

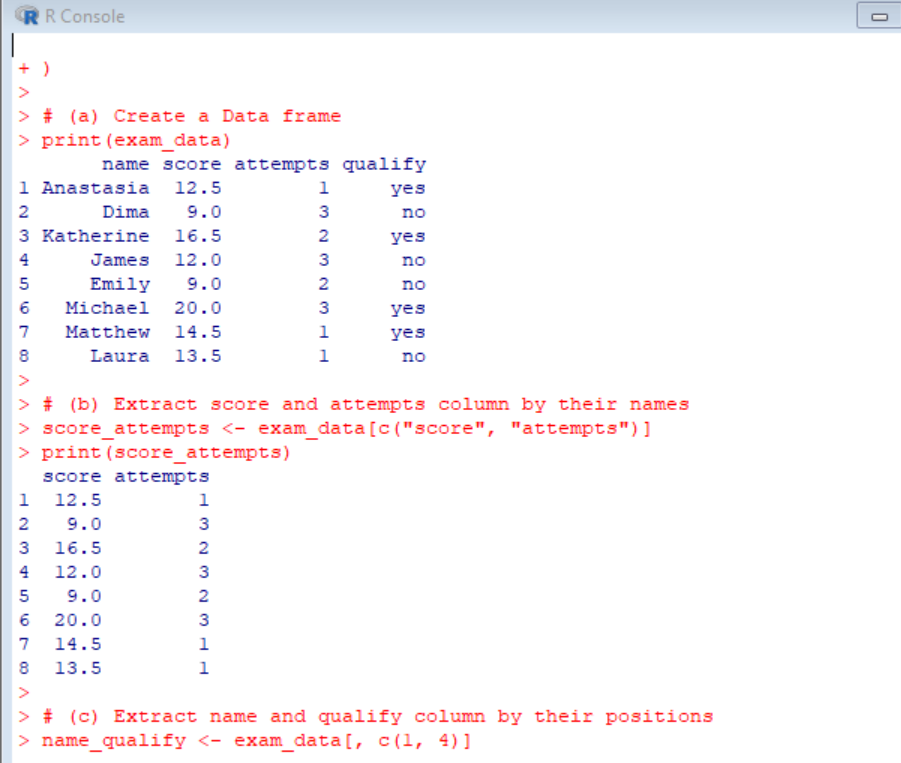
extracted\_data <- exam\_data[c(3, 5), c(1, 3)]

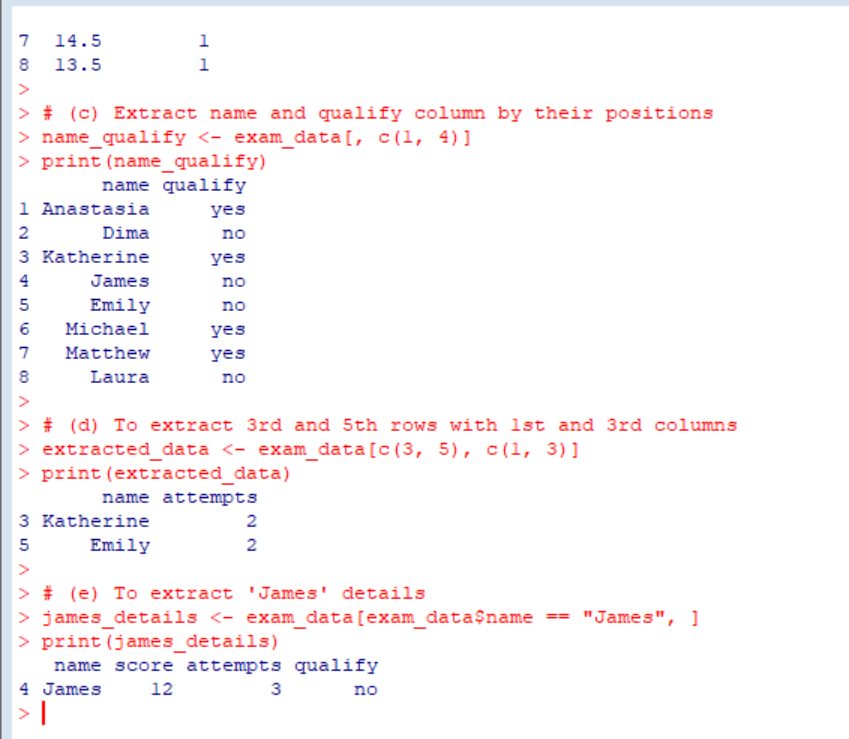
print(extracted\_data)

# (e) To extract 'James' details

james\_details <- exam\_data[exam\_data$name == "James", ]

print(james\_details)





**2.** # Create the data frame

n <- c(1, 1, 2, 2)

time <- c(1, 2, 1, 2)

x <- c(6, 3, 2, 5)

y <- c(1, 4, 6, 9)

data <- data.frame(n, time, x, y)

# (i) Melt the data and display as a long-format data

library("reshape2")

melted\_data <- melt(data, id.vars = c("n", "time"))

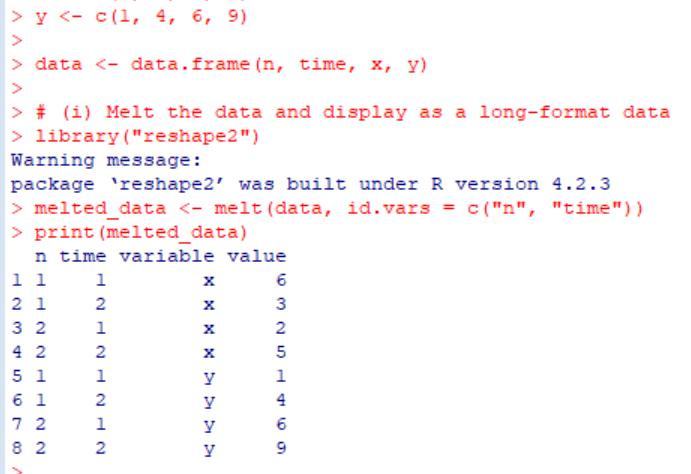
print(melted\_data)

# (ii) Compute the average of x and y with respect to "time"

library("reshape")

averages <- cast(data, time ~ ., mean)

print(averages)





**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

|  |  |  |
| --- | --- | --- |
| **Course Code: ITA04** | **Course Name: Statistics with R Programming** | |
| **Branch: CSE** |  | **Academic Year: 2021-2020 (Odd)** |
| **Date of Exam: 30/8/21** | **Max. Marks: 20M** | **Time: 1 hour** |

**ANSWER ALL THE QUESTIONS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **CLASS TEST 4** | **MARKS** | **COs** | **BT LEVEL** |
| 1 | Calculate the coefficient of correlation to the following data  x <-10, 12 ,18, 24, 23, 27  y <-13 ,18, 12 ,25 ,30 ,10 | 10 | CO4 | K3 |
| 2 | Suppose a hospital tested the age and body fat data for 18 randomly selected adults with the following result    Calculate the standard deviation of age and %fat. | 10 | CO4 | K3 |

**CLASS TEST 4-ANSWERS**

**1.** # Given data

x <- c(10, 12, 18, 24, 23, 27)

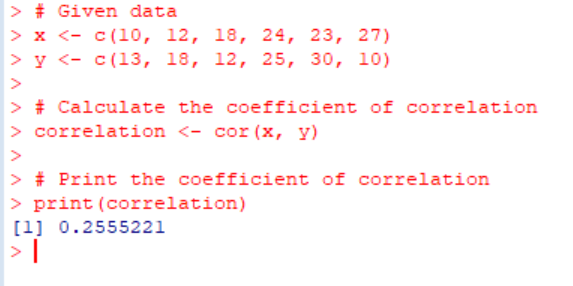
y <- c(13, 18, 12, 25, 30, 10)

# Calculate the coefficient of correlation

correlation <- cor(x, y)

# Print the coefficient of correlation

print(correlation)

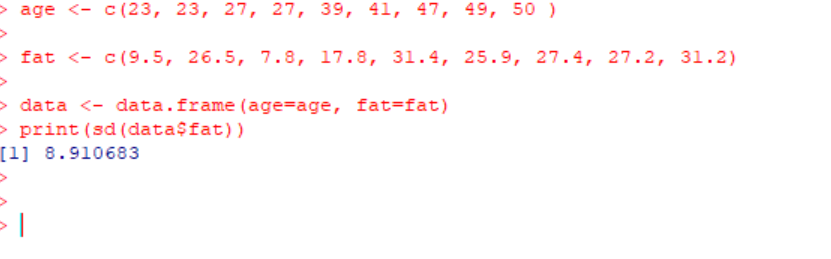
****

**2.** age <- c(23, 23, 27, 27, 39, 41, 47, 49, 50 )

fat <- c(9.5, 26.5, 7.8, 17.8, 31.4, 25.9, 27.4, 27.2, 31.2)

data <- data.frame(age=age, fat=fat)

print(sd(data$fat))

****

**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

|  |  |  |
| --- | --- | --- |
| **Course Code: ITA04** | **Course Name: Statistics with R Programming** | |
| **Branch: CSE** |  | **Academic Year: 2021-2020 (Odd)** |
| **Date of Exam: 18/9/21** | **Max. Marks: 20M** | **Time: 1 hour** |

**ANSWER ALL THE QUESTIONS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **CLASS TEST 5** | **MARKS** | **COs** | **BT LEVEL** |
| 1 | Explain in detail about the high level plotting functions. | 10 | CO5 | K3 |
| 2 | Suppose that the data for analysis includes the attribute age. The age values for the data tuples are (in increasing order) 13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70.  Can you find (roughly) the first quartile (Q1) and the third quartile (Q3) of the data?.  Draw the Boxplot with that information. Give the R code for same. | 10 | CO5 | K3 |

**CLASS TEST 5-ANSWERS**

**1.** In R, high-level plotting functions provide a convenient and intuitive way to create various types of plots and visualizations. These functions are part of the base R graphics system and are easy to use for generating a wide range of plots. Here's an overview of some commonly used high-level plotting functions in R:

**plot():** The **plot()** function is a versatile function that can be used to create various types of plots, including scatter plots, line plots, bar plots, and more. It takes multiple arguments to specify the data to be plotted, such as x and y variables, data points' appearance, labels, and titles. For example, **plot(x, y)** creates a scatter plot of variable **x** against **y**.

1. **hist():** The **hist()** function is used to create histograms, which display the distribution of a numeric variable. It takes a single argument representing the data to be plotted, along with additional arguments to customize the appearance of the histogram, such as the number of bins, colors, labels, and titles.
2. **boxplot():** The **boxplot()** function is used to create box plots, also known as whisker plots. Box plots summarize the distribution of a numeric variable by displaying quartiles, median, and outliers. The function takes one or more numeric vectors as input and provides options to customize the appearance of the box plot, including labels, titles, and colors.
3. **barplot():** The **barplot()** function is used to create vertical or horizontal bar charts. It is commonly used to visualize categorical data or to compare different groups or categories. The function takes a vector or matrix of numeric values, and additional arguments allow customization of the bar chart's appearance, such as colors, labels, titles, and legends.
4. **pie():** The **pie()** function is used to create pie charts, which represent proportions of a whole. It takes a vector of numeric values as input, where each value represents a portion of the pie. Additional arguments can be used to customize the appearance of the pie chart, including colors, labels, title, and explode slices.

**plotly::plot\_ly():** The **plot\_ly()** function from the Plotly package provides interactive plots and visualizations. It allows you to create interactive scatter plots, line plots, bar plots, and more. With Plotly, you can add interactivity, tooltips, zooming, panning, and other interactive features to your plots.

**2.** # Given data

age <- c(13, 15, 16, 16, 19, 20, 20, 21, 22, 22, 25, 25, 25, 25, 30, 33, 33, 35, 35, 35, 35, 36, 40, 45, 46, 52, 70)

# Calculate quartiles

Q1 <- quantile(age, 0.25)

Q3 <- quantile(age, 0.75)

# Create boxplot

boxplot(age, main = "Boxplot of Age", ylab = "Age", ylim = c(min(age) - 5, max(age) + 5), col = "lightblue")

# Add quartile lines

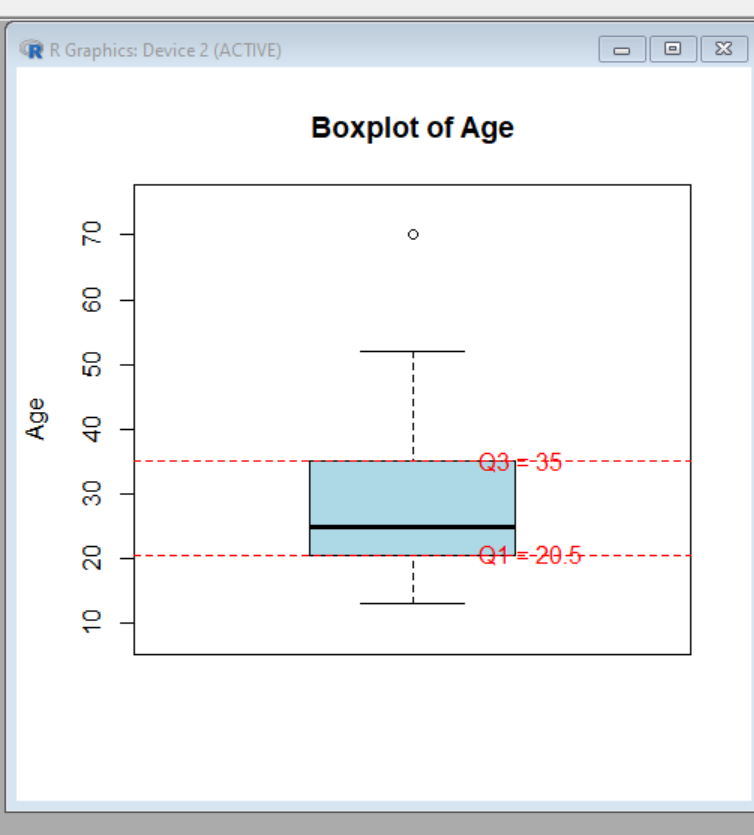
abline(h = Q1, lty = 2, col = "red")

abline(h = Q3, lty = 2, col = "red")

# Add labels

text(1.1, Q1, paste("Q1 =", Q1), pos = 4, col = "red")

text(1.1, Q3, paste("Q3 =", Q3), pos = 4, col = "red")





**SAVEETHA SCHOOL OF ENGINEERING**

**SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES**

|  |  |  |
| --- | --- | --- |
| **Course Code: ITA04** | **Course Name: Statistics with R Programming** | |
| **Branch: CSE** |  | **Academic Year: 2021-2020 (Odd)** |
| **Date of Exam: 20-10-2021** | **Max. Marks: 100 M** | **Time: 3 hours** |

**ANSWER ALL THE QUESTIONS**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S.No** | **MODEL EXAMINATION** | **MARKS** | **COs** | **BT LEVEL** |
| 1 | What are the different forms of data types and how to test the data type in R? Give one example for each. | 10 | CO1 | K3 |
| 2 | Write R code function to generate first n terms of a Fibonacci series | 10 | CO1 | K3 |
| 3 | The price of one kg of rice is Rs. 40.75 and one kg of sugar is Rs. 30. Write R program to get the total amount of 2kg rice and 5kg sugar purchase. | 10 | CO2 | K3 |
| 4 | Create an Array with name “MySales” with 30 observations using following methods:  a) By using the array with dimensions 3, 5 and2.  b) By using Vector method | 10 | CO2 | K3 |
| 5 | Write R Program to find maximum and minimum value of a given vector using control statement. nums = c(10, 20, 30, 40, 50, 60) | 10 | CO3 | K3 |
| 6 | Given 3 linear equations  5x1-x2+3x3=7  3x1+2x2+4x3=10  7x1-4x2+8x3=-14  Solve the linear equations and find x1, x2, x3 values using appropriate R function | 10 | CO3 | K3 |
| 7 | (a)Write a R program to sort a given data frame by name and score columns  (b)Write R program to find a average score based on “qualify”  exam\_data = data.frame(  name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura'),  score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5),  attempts = c(1, 3, 2, 3, 2, 3, 1, 1),  qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no')) | 10 | CO4 | K3 |
| 8 | Write a R program to create inner, outer, left, right join(merge) from given two data frames  df1 = data.frame(numid = c(12, 14, 10, 11))  df2 = data.frame(numid = c(13, 15, 11, 12)) | 10 | CO4 | K3 |
| 9 | Compute the statistical summary for the following data and write R code for same Find the Skewness of the data  X<-62, 58, 68, 45, 81, 60, 68 ,48, 58, 70 | 10 | CO5 | K3 |
| 10 | Find the quartiles of this data set: 6, 47, 49, 15, 43, 41, 7, 39, 43, 41, 36. and draw the boxplot. Give the R code for same. | 10 | CO5 | K3 |

**MODEL EXAMINATION - ANSWER KEY**

1 A ) In R, there are several different data types that can be used to store and manipulate data. Here are some of the common data types in R:

1)Numeric: This data type is used to represent numbers, both integers and decimals. You can test the data type using the is.numeric() function.

Example:

x <- 10

is.numeric(x) # Returns TRUE

2)Integer: This data type is specifically used to represent integers. You can test the data type using the is.integer() function.

Example:x <- 5L

is.integer(x) # Returns TRUE

3)Character: This data type is used to represent strings of characters. You can test the data type using the is.character() function.

Example:x <- "Hello, world!"

is.character(x) # Returns TRUE

4)Logical: This data type is used to represent Boolean values (TRUE or FALSE). You can test the data type using the is.logical() function.

Example:

Copy code

x <- TRUE

is.logical(x) # Returns TRUE

5)Factor: This data type is used to represent categorical data with fixed levels or categories. You can test the data type using the is.factor() function.

Example:

Copy code

x <- factor(c("red", "blue", "green"))

is.factor(x) # Returns TRUE

Date: This data type is used to represent dates. You can test the data type using the is.Date() function.

Example:

Copy code

x <- as.Date("2023-05-23")

is.Date(x) # Returns TRUE

6)Complex: This data type is used to represent complex numbers with real and imaginary parts. You can test the data type using the is.complex() function.

Example:

Copy code

x <- 2 + 3i

is.complex(x) # Returns TRUE

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2 A )generateFibonacci <- function(n) {

fibonacci <- numeric(n) # Create an empty vector to store the Fibonacci series

if (n >= 1) {

fibonacci[1] <- 0 # Set the first term of Fibonacci series to 0

}

if (n >= 2) {

fibonacci[2] <- 1 # Set the second term of Fibonacci series to 1

}

if (n > 2) {

for (i in 3:n) {

fibonacci[i] <- fibonacci[i-1] + fibonacci[i-2] # Generate subsequent terms of Fibonacci series

}

}

return(fibonacci)

}

# Example usage:

n <- 10

fibSeries <- generateFibonacci(n)

print(fibSeries)

output : [1] 0 1 1 2 3 5 8 13 21 34

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

3 A ) rice\_price <- 40.75 # Price of one kg of rice

sugar\_price <- 30.00 # Price of one kg of sugar

rice\_weight <- 2 # Weight of rice in kg

sugar\_weight <- 5 # Weight of sugar in kg

rice\_amount <- rice\_price \* rice\_weight # Total amount for rice

sugar\_amount <- sugar\_price \* sugar\_weight # Total amount for sugar

total\_amount <- rice\_amount + sugar\_amount # Total amount for rice and sugar

# Print the total amount

cat("Total amount:", total\_amount)

output : Total amount: 247.5

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4 A ) a) By using the array with dimensions 3, 5, and 2:

# Using the array method

MySales <- array(1:30, dim = c(3, 5, 2))

print(MySales)

output : , , 1

[,1] [,2] [,3] [,4] [,5]

[1,] 1 4 7 10 13

[2,] 2 5 8 11 14

[3,] 3 6 9 12 15

, , 2

[,1] [,2] [,3] [,4] [,5]

[1,] 16 19 22 25 28

[2,] 17 20 23 26 29

[3,] 18 21 24 27 30

b) By using the vector method:

# Using the vector method

MySales <- 1:30

print(MySales)

output : [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5 A ) nums <- c(10, 20, 30, 40, 50, 60) # Given vector

# Initializing variables to store the maximum and minimum values

max\_value <- nums[1]

min\_value <- nums[1]

# Finding the maximum and minimum values using control statements

for (num in nums) {

if (num > max\_value) {

max\_value <- num

}

if (num < min\_value) {

min\_value <- num

}

}

# Printing the maximum and minimum values

cat("Maximum value:", max\_value, "\n")

cat("Minimum value:", min\_value)

output : Maximum value: 60

Minimum value: 10

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

6 A ) # Define the coefficient matrix A

A <- matrix(c(5, -1, 3,

3, 2, 4,

7, -4, 8), nrow = 3, byrow = TRUE)

# Define the constant vector b

b <- c(7, 10, -14)

# Solve the system of equations

solution <- solve(A, b)

# Extract the values of x1, x2, x3 from the solution vector

x1 <- solution[1]

x2 <- solution[2]

x3 <- solution[3]

# Print the values of x1, x2, x3

cat("x1 =", x1, "\n")

cat("x2 =", x2, "\n")

cat("x3 =", x3)

output : x1 = 1

x2 = -1

x3 = -1

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

7 A ) a) Here's an R program to sort a given data frame by the "name" and "score" columns:

exam\_data <- data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no')

)

# Sort the data frame by name and score columns

sorted\_data <- exam\_data[order(exam\_data$name, exam\_data$score), ]

# Print the sorted data frame

print(sorted\_data)

output :

name score attempts qualify

1 Anastasia 12.5 1 yes

4 James 12.0 3 no

3 Katherine 16.5 2 yes

5 Emily 9.0 2 no

2 Dima 9.0 3 no

8 Laura 13.5 1 no

7 Matthew 14.5 1 yes

6 Michael 20.0 3 yes

b) Here's an R program to find the average score based on the "qualify" column:

exam\_data <- data.frame(

name = c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura'),

score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5),

attempts = c(1, 3, 2, 3, 2, 3, 1, 1),

qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no')

)

# Calculate the average score based on qualify

average\_score <- mean(exam\_data$score[exam\_data$qualify == "yes"])

# Print the average score

cat("Average score based on qualify: ", average\_score)

output :

Average score based on qualify: 15.66667

8 A) df1 <- data.frame(numid = c(12, 14, 10, 11))

df2 <- data.frame(numid = c(13, 15, 11, 12))

# Inner join

inner\_join <- merge(df1, df2, by = "numid", all = FALSE)

print("Inner Join:")

print(inner\_join)

# Outer join

outer\_join <- merge(df1, df2, by = "numid", all = TRUE)

print("Outer Join:")

print(outer\_join)

# Left join

left\_join <- merge(df1, df2, by = "numid", all.x = TRUE)

print("Left Join:")

print(left\_join)

# Right join

right\_join <- merge(df1, df2, by = "numid", all.y = TRUE)

print("Right Join:")

print(right\_join)

output : Inner Join:

numid

1 11

2 12

Outer Join:

numid

1 10

2 11

3 12

4 13

5 14

6 15

Left Join:

numid

1 10

2 11

3 12

4 14

Right Join:

numid

1 11

2 12

3 13

4 15

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

9 A ) # Install and load the e1071 package (if not already installed)

if (!require(e1071)) {

install.packages("e1071")

}

library(e1071)

# Define the data

X <- c(62, 58, 68, 45, 81, 60, 68, 48, 58, 70)

# Compute the statistical summary

summary(X)

# Calculate the skewness

skewness(X)

output : Min. 1st Qu. Median Mean 3rd Qu. Max.

45.00 58.00 62.00 61.20 68.00 81.00

[1] -0.2581064

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

10 A ) # Define the data set

data <- c(6, 47, 49, 15, 43, 41, 7, 39, 43, 41, 36)

# Find the quartiles

q <- quantile(data, probs = c(0.25, 0.5, 0.75))

# Print the quartiles

cat("First Quartile (Q1):", q[1], "\n")

cat("Median (Q2):", q[2], "\n")

cat("Third Quartile (Q3):", q[3], "\n")

# Draw the boxplot

boxplot(data, main = "Boxplot of Data", xlab = "Data Set", ylab = "Values")

output : First Quartile (Q1): 15

Median (Q2): 41

Third Quartile (Q3): 43

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_