



Faculty of CHAIR & SCOPE

BCSE351E L[25+26]

Submitted By:

Win Verma

Registration Number:

21BLC1219

Submitted To:

Dr. Vijayarajan

Date: 07/06/2023

Email: win.verma2021@vitstudent.ac.in

Lab Report – 3

Using R-Studio to Visualize Sample Codes

Aim:

- R programming and performing required operations.
 - Reading datasets using RStudio.
 - To view the details with RStudio.
 - To learn about Data Framing using Data Structures in R programming with R Studio [Latest Version with ALL Required Packages Installed].
-

1. Data_frame0.R:

```
celebrities <- data.frame(
  name = c("Andrew", "Mathew", "Dany", "Philip", "John", "Bing", "Monica"),
  age = c(28, 23, 49, 29, 38, 23, 29),
  income = c(25.2, 10.5, 11, 21.9, 44, 11.5, 45)
)

# Get elements from rows (2,5), columns (1,3)
elements <- celebrities[c(2, 5), c(1, 3)]

print(elements)

age <- c(17, 19, 21, 37, 18, 19, 47, 18, 19)
score <- c(12, 10, 11, 15, 16, 14, 25, 21, 29)

who()

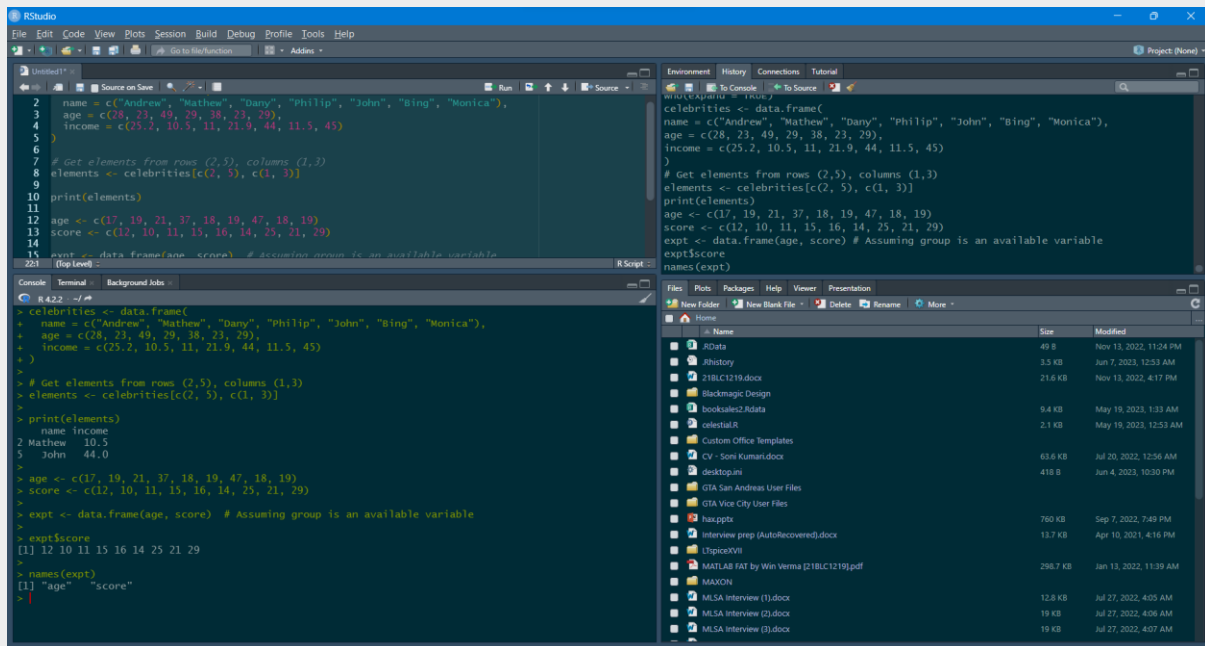
expt <- data.frame(age, gender, group, score) # Assuming gender and group are available
variables

expt$score

names(expt)

who(expand = TRUE)
```

Data Analytics – R Programming



2. Data_frame1.R:

Write an R program to create an empty data frame.

```
df <- data.frame(  
  Ints = integer(),  
  Doubles = double(),  
  Characters = character(),  
  Logicals = logical(),  
  Factors = factor(),  
  stringsAsFactors = FALSE  
)
```

```
print("Structure of the empty dataframe:")  
print(str(df))
```

Write an R program to create a data frame from four given vectors.

Write an R program to get the structure of a given data frame.

Write an R program to save the information of a data frame in a file and display the information of the file.

```
name <- c('Alex', 'Roy', 'Kathe', 'James', 'Emily', 'Mike', 'Matt', 'Little', 'Kevin', 'Jonas')  
score <- c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19)  
attempts <- c(1, 3, 2, 3, 2, 3, 1, 1, 2, 1)  
qualify <- c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
```

```
print("Original data frame:")  
print(name)  
print(score)  
print(attempts)  
print(qualify)
```

```
exam_data <- data.frame(name, score, attempts, qualify)

print("Original dataframe:")
print(exam_data)

print("Structure of the said data frame:")
print(str(exam_data))

save(exam_data, file = "data.rda")

load("data.rda")

file.info("data.rda")

# Write an R program to extract 3rd and 5th rows with 1st and 3rd columns from a given
data frame.

print("Extract 3rd and 5th rows with 1st and 3rd columns:")
result <- exam_data[c(3, 5), c(1, 3)]
print(result)

# Write an R program to replace NA values with 3 in a given data frame.

exam_data <- data.frame(
  name = c('Alex', 'Roy', 'Kathe', 'James', 'Emily', 'Mike', 'Matt', 'Little', 'Kevin', 'Jonas'),
  score = c(12.5, 9, 16.5, 12, 9, 20, 14.5, 13.5, 8, 19),
  attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),
  qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'yes', 'no', 'no', 'yes')
)

print("Original dataframe:")
print(exam_data)

exam_data[is.na(exam_data)] <- 3

print("After removing NA with 3, the said dataframe becomes:")
print(exam_data)

# Write an R program to create a data frame using two given vectors and display the
duplicated elements and unique rows of the said data frame.

a <- c(10, 20, 10, 10, 40, 50, 20, 30)
b <- c(10, 30, 10, 20, 0, 50, 30, 30)

print("Original data frame:")
ab <- data.frame(a, b)
print(ab)

print("Duplicate elements of the said data frame:")
print(duplicated(ab))
```

Data Analytics – R Programming

```
print("Unique rows of the said data frame:")
print(unique(ab))
```

Write an R program to count the number of NA values in a data frame column.

```
exam_data <- data.frame(
  name = c('Alex', 'Roy', 'Kathe', 'James', 'Emily', 'Mike', 'Jonas'),
  score = c(12.5, 9, 16.5, 12, 9, 20, 13.5),
  attempts = c(1, NA, 2, NA, 2, NA, NA),
  qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes', 'no')
)
```

```
print("Original dataframe:")
print(exam_data)
```

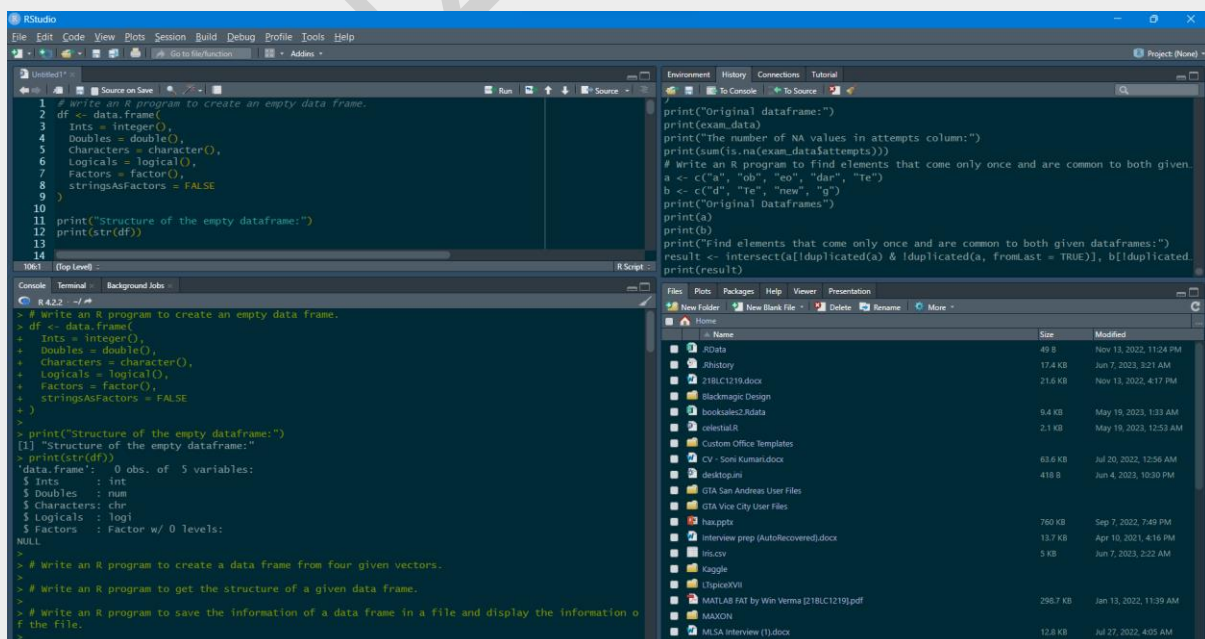
```
print("The number of NA values in attempts column:")
print(sum(is.na(exam_data$attempts)))
```

Write an R program to find elements that come only once and are common to both given data frames.

```
a <- c("a", "ob", "eo", "dar", "Te")
b <- c("d", "Te", "new", "g")
```

```
print("Original Dataframes")
print(a)
print(b)
```

```
print("Find elements that come only once and are common to both given dataframes:")
result <- intersect(a[!duplicated(a) & !duplicated(a, fromLast = TRUE)], b[!duplicated(b) & !duplicated(b, fromLast = TRUE)])
print(result)
```



The screenshot shows the RStudio environment with the following components:

- Source Editor:** Contains R code for creating a data frame, printing its structure, and finding unique elements in two vectors.
- Console:** Displays the output of the R code, including the structure of the empty data frame and the result of the intersection operation.
- Environment:** Shows the current environment with variables 'a' and 'b' listed.
- Files:** Shows the file explorer with various files and folders.

```
# Write an R program to create an empty data frame.
df <- data.frame()
Ints = integer(),
Doubles = double(),
Characters = character(),
Logicals = logical(),
Factors = factor(),
stringsAsFactors = FALSE
)
print("Structure of the empty dataframe:")
print(str(df))

# Write an R program to create a data frame from four given vectors.
# Write an R program to get the structure of a given data frame.
# Write an R program to save the information of a data frame in a file and display the information of the file.
```

```
print("Original dataframe:")
print(exam_data)

print("The number of NA values in attempts column:")
print(sum(is.na(exam_data$attempts)))

# Write an R program to find elements that come only once and are common to both given data frames.
a <- c("a", "ob", "eo", "dar", "Te")
b <- c("d", "Te", "new", "g")
print("Original Dataframes")
print(a)
print(b)

print("Find elements that come only once and are common to both given dataframes:")
result <- intersect(a[!duplicated(a) & !duplicated(a, fromLast = TRUE)], b[!duplicated(b) & !duplicated(b, fromLast = TRUE)])
print(result)
```

```
[1] "Structure of the empty dataframe:"
> print(str(df))
'data.frame': 0 obs. of 5 variables:
 $ Ints : int
 $ Doubles : num
 $ Characters: chr
 $ Logicals : logi
 $ Factors : Factor w/ 0 levels:
 NULL

> # Write an R program to create a data frame from four given vectors.
> # Write an R program to get the structure of a given data frame.
> # Write an R program to save the information of a data frame in a file and display the information of the file.
```

```
R 4.2.2 · ~/
> print("Original dataframe:")
[1] "Original dataframe:"
> print(exam_data)
  name score attempts qualify
1  Alex  12.5         1     yes
2   Roy   9.0        NA     no
3  Kathe 16.5         2     yes
4  James 12.0        NA     no
5  Emily   9.0         2     no
6   Mike 20.0        NA     yes
7   Matt 14.5         1     yes
8 Little 13.5        NA     no
9  Kevin   8.0         2     no
10 Jonas 19.0         1     yes
>
> exam_data[is.na(exam_data)] <- 3
>
> print("After removing NA with 3, the said dataframe becomes:")
[1] "After removing NA with 3, the said dataframe becomes:"
> print(exam_data)
  name score attempts qualify
1  Alex  12.5         1     yes
2   Roy   9.0         3     no
3  Kathe 16.5         2     yes
4  James 12.0         3     no
5  Emily   9.0         2     no
6   Mike 20.0         3     yes
7   Matt 14.5         1     yes
```

```
R 4.2.2 · ~/
5 Emily   9.0         2     no
6   Mike 20.0        NA     yes
7  Jonas 13.5        NA     no
>
> print("The number of NA values in attempts column:")
[1] "The number of NA values in attempts column:"
> print(sum(is.na(exam_data$attempts)))
[1] 4
>
> # Write an R program to find elements that come only once and are common to both given data frames.
>
> a <- c("a", "ob", "eo", "dar", "Te")
> b <- c("d", "Te", "new", "g")
>
> print("Original Dataframes")
[1] "Original Dataframes"
> print(a)
[1] "a" "ob" "eo" "dar" "Te"
> print(b)
[1] "d" "Te" "new" "g"
>
> print("Find elements that come only once and are common to both given dataframes:")
[1] "Find elements that come only once and are common to both given dataframes:"
> result <- intersect(a[!duplicated(a) & !duplicated(a, fromLast = TRUE)], b[!duplicated(b) & !duplicated(b, fromLast = TRUE)])
> print(result)
[1] "Te"
>
```

3. Data_frame2.R:

Write an R program to call the (built-in) dataset airquality. Check whether it is a data frame or not? Order the entire data frame

```
data <- airquality
```

```
print("Original data: Daily air quality measurements in New York, May to September 1973.")
print(class(data))
print(head(data, 10))
```

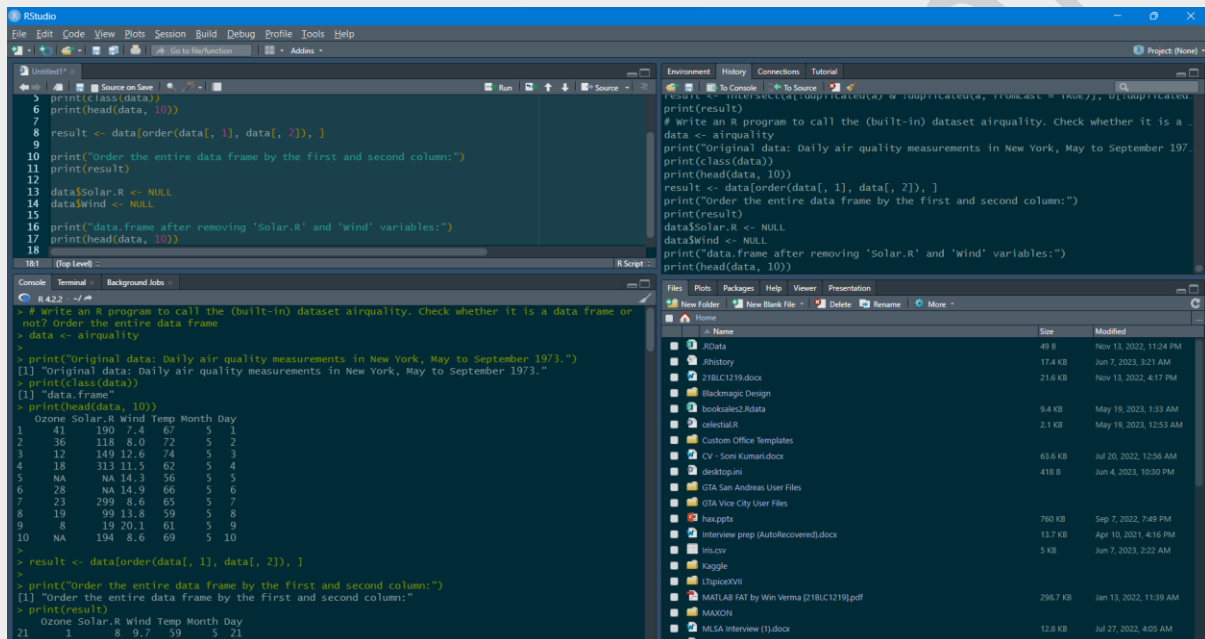
Data Analytics – R Programming

```
result <- data[order(data[, 1], data[, 2]), ]
```

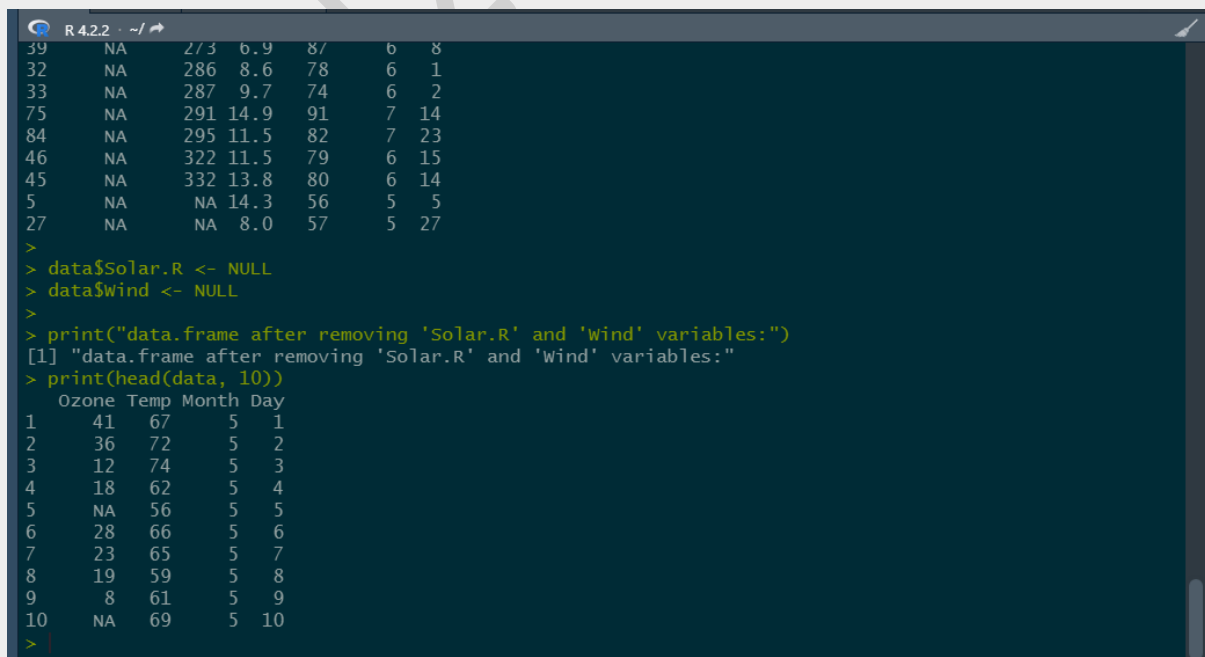
```
print("Order the entire data frame by the first and second column:")  
print(result)
```

```
data$Solar.R <- NULL  
data$Wind <- NULL
```

```
print("data.frame after removing 'Solar.R' and 'Wind' variables:")  
print(head(data, 10))
```



```
# Write an R program to call the (built-in) dataset airquality. Check whether it is a data frame or not? Order the entire data frame  
> data <- airquality  
>  
> print("Original data: Daily air quality measurements in New York, May to September 1973.")  
[1] "Original data: Daily air quality measurements in New York, May to September 1973."  
> print(class(data))  
[1] "data.frame"  
> print(head(data, 10))  
  Ozone Solar.R Wind Temp Month Day  
1    41     NA   7.4  67    5    1  
2    36     NA   8.0  72    5    2  
3    12     NA  12.6  74    5    3  
4    18     NA  11.5  62    5    4  
5    NA     NA  14.3  56    5    5  
6    28     NA  14.9  66    5    6  
7    23     NA   8.6  65    5    7  
8    19     NA  13.8  59    5    8  
9     8     NA  19.0  61    5    9  
10   NA     NA   8.6  69    5   10  
>  
> result <- data[order(data[, 1], data[, 2]), ]  
>  
> print("Order the entire data frame by the first and second column:")  
[1] "Order the entire data frame by the first and second column:"  
> print(result)  
  Ozone Solar.R Wind Temp Month Day  
21     1     8   9.7  59    5   21
```



```
R 4.2.2 ~ /  
39 NA 273 6.9 87 6 8  
32 NA 286 8.6 78 6 1  
33 NA 287 9.7 74 6 2  
75 NA 291 14.9 91 7 14  
84 NA 295 11.5 82 7 23  
46 NA 322 11.5 79 6 15  
45 NA 332 13.8 80 6 14  
5 NA NA 14.3 56 5 5  
27 NA NA 8.0 57 5 27  
>  
> data$Solar.R <- NULL  
> data$Wind <- NULL  
>  
> print("data.frame after removing 'Solar.R' and 'Wind' variables:")  
[1] "data.frame after removing 'Solar.R' and 'Wind' variables:"  
> print(head(data, 10))  
  Ozone Temp Month Day  
1    41    67    5    1  
2    36    72    5    2  
3    12    74    5    3  
4    18    62    5    4  
5    NA    56    5    5  
6    28    66    5    6  
7    23    65    5    7  
8    19    59    5    8  
9     8    61    5    9  
10   NA    69    5   10  
> |
```

4. Data_frame3.R:

Write an R program to add new row(s) to an existing data frame.

```
exam_data <- data.frame(  
  name = c('Alex', 'Roy', 'Kathe', 'James', 'Emily', 'Jonas'),  
  score = c(12.5, 9, 16.5, 12, 9, 20),  
  attempts = c(1, 3, 2, 3, 2, 3),  
  qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes')  
)
```

```
print("Original dataframe:")  
print(exam_data)
```

```
new_exam_data <- data.frame(  
  name = c('Robert', 'Sophia'),  
  score = c(10.5, 9),  
  attempts = c(1, 3),  
  qualify = c('yes', 'no')  
)
```

```
exam_data <- rbind(exam_data, new_exam_data)
```

```
print("After adding new row(s) to an existing data frame:")  
print(exam_data)
```

Write an R program to add a new column in a given data frame.
print("New data frame after adding the 'country' column:")

```
country <- c("USA", "USA", "USA", "USA", "USA", "USA", "USA", "USA")  
n <- nrow(exam_data)  
exam_data$country <- rep(country, length.out = n)
```

```
print(exam_data)
```

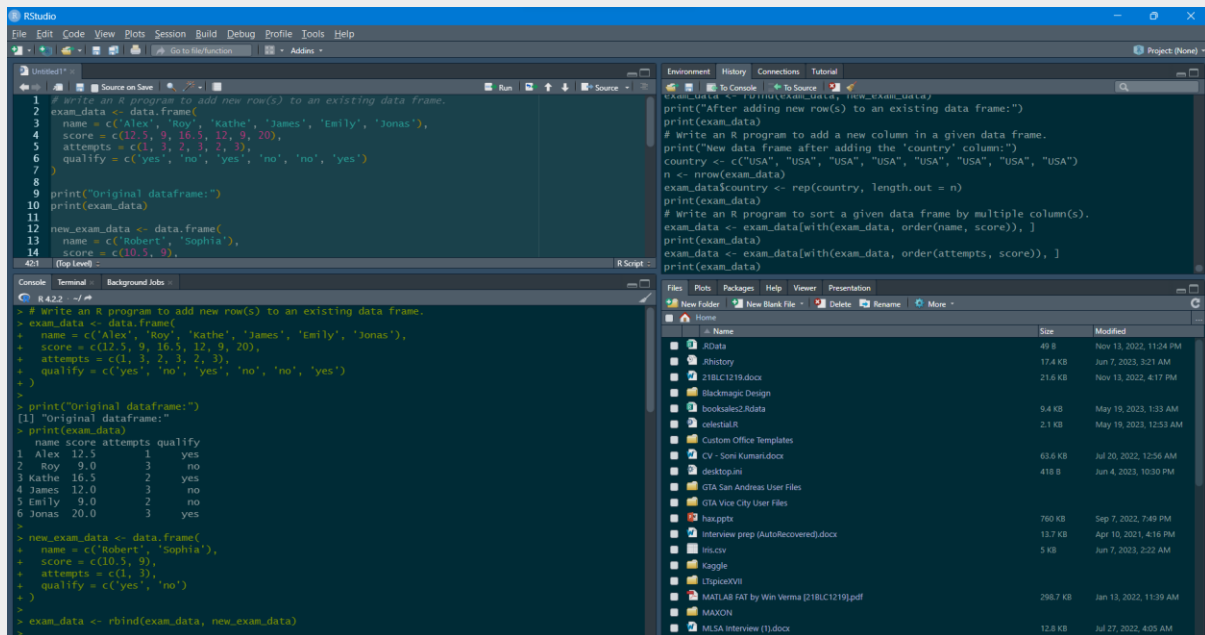
Write an R program to sort a given data frame by multiple column(s).
exam_data <- exam_data[with(exam_data, order(name, score)),]

```
print(exam_data)
```

```
exam_data <- exam_data[with(exam_data, order(attempts, score)), ]
```

```
print(exam_data)
```


Data Analytics – R Programming



```
# Write an R program to add new row(s) to an existing data frame.
exam_data <- data.frame(
  name = c('Alex', 'Roy', 'Kathe', 'James', 'Emily', 'Jonas'),
  score = c(12.5, 9, 16.5, 12, 9, 20),
  attempts = c(1, 2, 3, 2, 3),
  qualify = c('yes', 'no', 'yes', 'no', 'no', 'yes')
)

print("Original dataframe:")
print(exam_data)

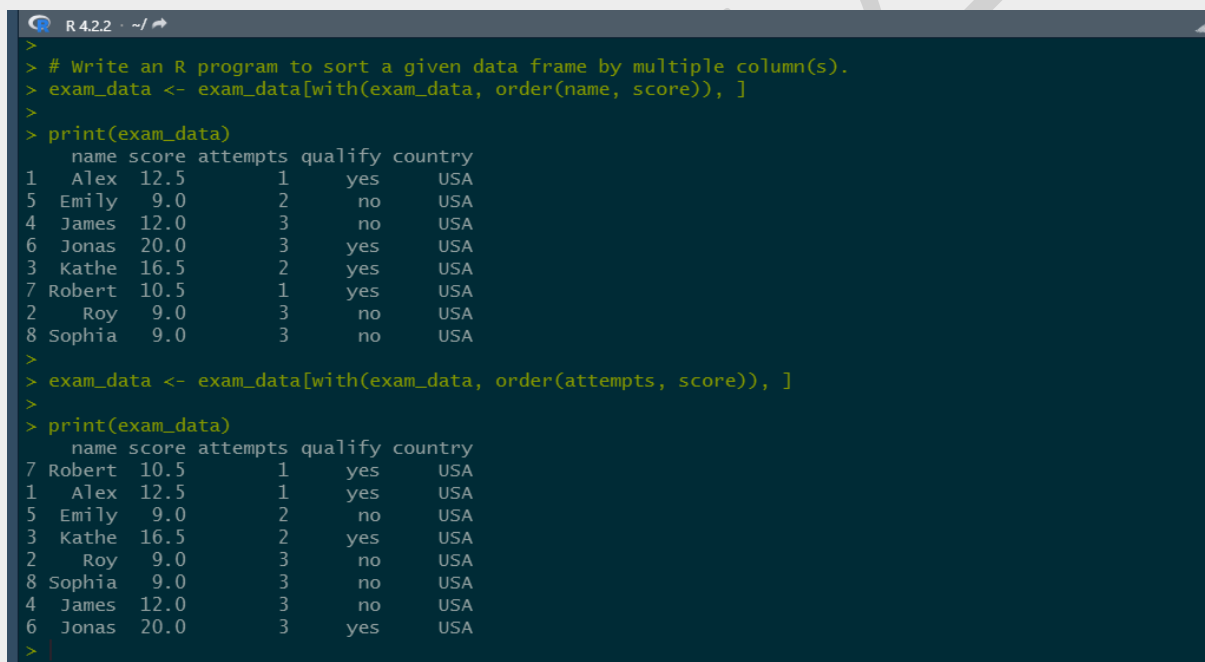
new_exam_data <- data.frame(
  name = c('Robert', 'Sophia'),
  score = c(10.5, 9)
)

exam_data <- rbind(exam_data, new_exam_data)

print(exam_data)

# Write an R program to add a new column in a given data frame.
print("New data frame after adding the 'country' column:")
country <- c("USA", "USA", "USA", "USA", "USA", "USA")
n <- nrow(exam_data)
exam_data$country <- rep(country, length.out = n)
print(exam_data)

# Write an R program to sort a given data frame by multiple column(s).
exam_data <- exam_data[with(exam_data, order(name, score)), ]
print(exam_data)
exam_data <- exam_data[with(exam_data, order(attempts, score)), ]
print(exam_data)
```



```
> # Write an R program to sort a given data frame by multiple column(s).
> exam_data <- exam_data[with(exam_data, order(name, score)), ]
> 
> print(exam_data)
  name score attempts qualify country
1 Alex 12.5      1    yes    USA
5 Emily  9.0      2    no    USA
4 James 12.0      3    no    USA
6 Jonas 20.0      3    yes    USA
3 Kathe 16.5      2    yes    USA
7 Robert 10.5     1    yes    USA
2 Roy    9.0      3    no    USA
8 Sophia 9.0      3    no    USA
> 
> exam_data <- exam_data[with(exam_data, order(attempts, score)), ]
> 
> print(exam_data)
  name score attempts qualify country
7 Robert 10.5      1    yes    USA
1 Alex 12.5      1    yes    USA
5 Emily  9.0      2    no    USA
3 Kathe 16.5      2    yes    USA
2 Roy    9.0      3    no    USA
8 Sophia 9.0      3    no    USA
4 James 12.0      3    no    USA
6 Jonas 20.0      3    yes    USA
> 
```

5. Data_frame4.R:

Write an R program to create inner, outer, left, right join (merge) from given two data frames.

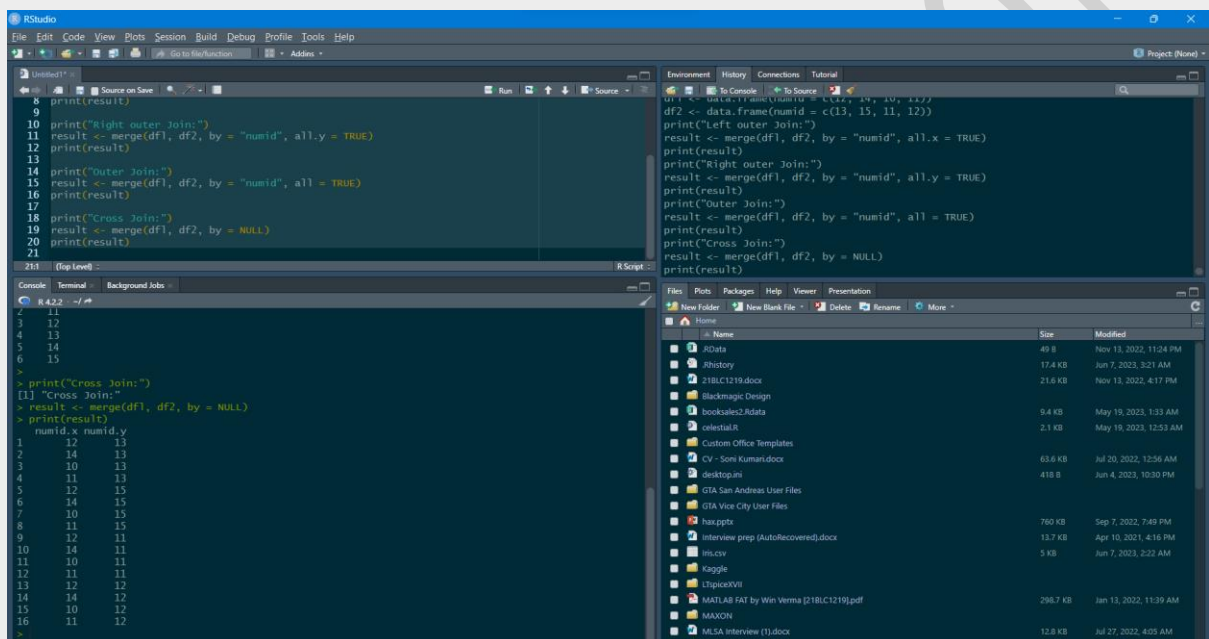
```
dfl <- data.frame(numid = c(12, 14, 10, 11))
df2 <- data.frame(numid = c(13, 15, 11, 12))
```

```
print("Left outer Join:")
result <- merge(dfl, df2, by = "numid", all.x = TRUE)
print(result)
```

```
print("Right outer Join:")
result <- merge(df1, df2, by = "numid", all.y = TRUE)
print(result)
```

```
print("Outer Join:")
result <- merge(df1, df2, by = "numid", all = TRUE)
print(result)
```

```
print("Cross Join:")
result <- merge(df1, df2, by = NULL)
print(result)
```



6. Data_frame5.R:

```
# creating a data frame
data_frame <- data.frame(col1 = 1:10,
                        col2 = 11:20,
                        col3 = c(rep(TRUE, 4), rep(FALSE, 6)))
```

```
print("Original DataFrame")
print(data_frame)
```

```
# defining the function
user_defined_func <- function(x) {
  # subtracting the value of 1 from each
  x - 1
}
```

```
data_frame_temp <- apply(data_frame[, c("col1", "col2")], 2, user_defined_func)
```

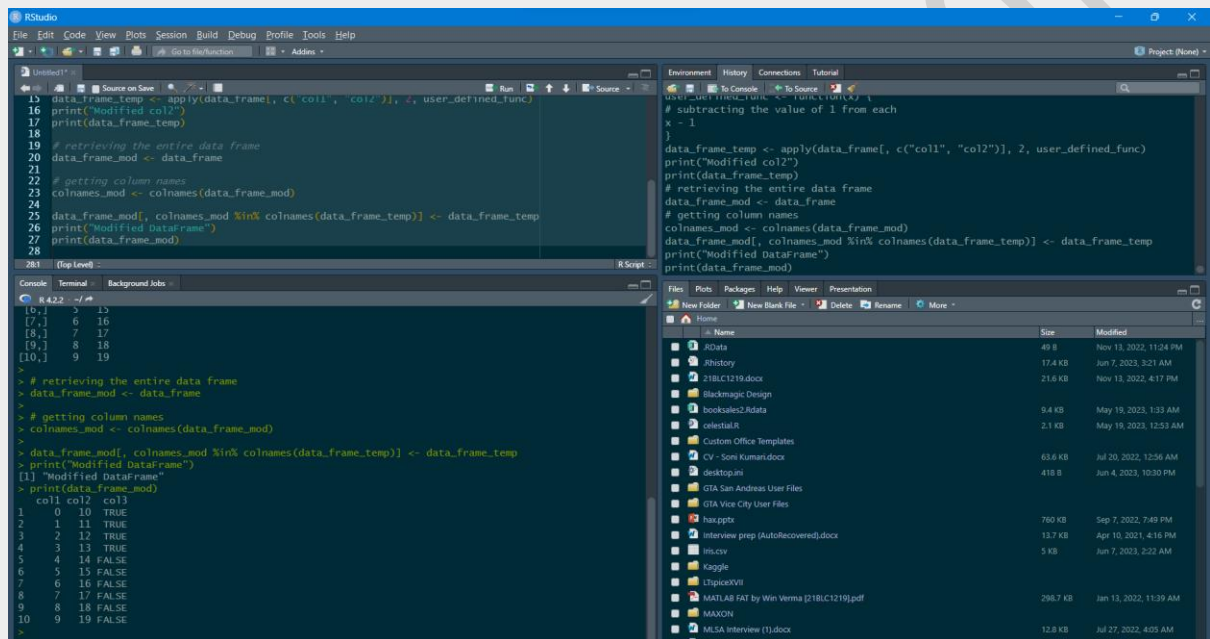
Data Analytics – R Programming

```
print("Modified col2")
print(data_frame_temp)
```

```
# retrieving the entire data frame
data_frame_mod <- data_frame
```

```
# getting column names
colnames_mod <- colnames(data_frame_mod)
```

```
data_frame_mod[, colnames_mod %in% colnames(data_frame_temp)] <- data_frame_temp
print("Modified DataFrame")
print(data_frame_mod)
```



The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains the R code for applying a function to columns, retrieving the entire data frame, getting column names, and modifying the data frame based on column names.
- Environment:** Shows the objects created in the environment, including `data_frame`, `data_frame_mod`, `colnames_mod`, and `data_frame_temp`.
- Console:** Displays the output of the code, including the modified data frame and the column names.
- Files:** Shows the file explorer with various files and folders.

```
13 data_frame_temp <- apply(data_frame[, c("col1", "col2")], 2, user_defined_func)
14 print("Modified col2")
15 print(data_frame_temp)
16
17 # retrieving the entire data frame
18 data_frame_mod <- data_frame
19
20 # getting column names
21 colnames_mod <- colnames(data_frame_mod)
22
23 data_frame_mod[, colnames_mod %in% colnames(data_frame_temp)] <- data_frame_temp
24 print("Modified DataFrame")
25 print(data_frame_mod)
```

Console Output:

```
[1] "Modified DataFrame"
  col1 col2 col3
1    0   10  TRUE
2    1   11  TRUE
3    2   12  TRUE
4    3   13  TRUE
5    4   14 FALSE
6    5   15 FALSE
7    6   16 FALSE
8    7   17 FALSE
9    8   18 FALSE
10   9   19 FALSE
```

Thanks for scrolling all the way down till here! 😊