**Assignment 3**

**A.KAILASH NATH REDDY(191911147)**

**ITA0448-STATISTICS AND R PROGRAMMING**

1. How to use the cbind() and rbind() in data frame for the fields city and zipcodedatas using vector and data frame.

Create a vectors: cbind() function:

Output: city zipcode

1. delhi 123456
2. bangalore 789654
3. chennai 698748 [4] mumbai 456986 rbind() function:

Output: city zipcode

1. delhi 123456
2. bangalore 789654
3. chennai 698748
4. mumbai 456986
5. punjab 456978 [6] kerala 569875 CODE:

> # create a vector for city

> city\_vec = c('delhi', 'bangalore', 'chennai', 'mumbai')

> # create a vector for zip code

> zip\_vec = c(123456, 789654, 698748, 456986)

> # use cbind() to create a data frame

> cbind\_df = data.frame(city = city\_vec, zipcode = zip\_vec)

> # view the output > print(cbind\_df) city zipcode 1 delhi 123456

1. bangalore 789654
2. chennai 698748
3. mumbai 456986

> # create a new data frame to add rows

> rbind\_df = data.frame(city = c('punjab', 'kerala'),

+ zipcode = c(456978, 569875))

> # use rbind() to add rows to the existing data frame

> final\_df = rbind(cbind\_df, rbind\_df)

> # view the output > print(final\_df) city zipcode 1 delhi 123456

1. bangalore 789654
2. chennai 698748
3. mumbai 456986
4. punjab 456978
5. kerala 569875

2. Create First Dataset with variables

* surname
* nationality

Create Second Dataset with variables

* surname
* movies

The common key variable is surname. How to merge both data and check if the dimensionality is 7x3.

Output:

surname nationality title

1. Hitchcock UK Psycho
2. Hitchcock UK North by Northwest
3. Polanski Poland Chinatown
4. Scorsese US Taxi Driver

1. Spielberg US Super 8
2. Spielberg US Catch Me If You Can 7 Tarantino US Reservoir Dogs CODE:

import pandas as pd

# create the first dataset

df1 = pd.DataFrame({

'surname': ['Hitchcock', 'Polanski', 'Scorsese'],

'nationality': ['UK', 'Poland', 'US']

})

# create the second dataset df2 = pd.DataFrame({

'surname': ['Hitchcock', 'Spielberg', 'Tarantino'],

'movies': ['Psycho, North by Northwest', 'Super 8, Catch Me If You Can',

'Reservoir Dogs']

})

# merge the two datasets merged\_df = pd.merge(df1, df2, on='surname')

# split the 'movies' column into separate rows merged\_df = merged\_df.assign(movies=merged\_df['movies'].str.split(',

')).explode('movies')

# add a 'title' column based on the 'movies' column merged\_df = merged\_df.assign(title=merged\_df['movies'])

# remove the 'movies' column merged\_df = merged\_df.drop('movies', axis=1)

# reorder the columns merged\_df = merged\_df[['surname', 'nationality', 'title']] # check the dimensionality of the merged dataset assert merged\_df.shape == (7, 3) # view the final output print(merged\_df)

3. Write a R program to create an empty data frame.

Output:

[1] &quot;Structure of the empty dataframe:&quot; &#39;data.frame&#39;: 0 obs. of 5 variables:

$ Ints : int

$ Doubles : num

$ Characters: chr

$ Logicals :logi

$ Factors : Factor w/ 0 levels:

NULL CODE:

> # create an empty data frame

> empty\_df <- data.frame(Ints = integer(),

+ Doubles = numeric(),

+ Characters = character(),

+ Logicals = logical(),

+ Factors = factor(levels = character()))

> # print the structure of the empty data frame > cat("Structure of the empty dataframe:\n") Structure of the empty dataframe:

> str(empty\_df)

'data.frame': 0 obs. of 5 variables:

$ Ints : int

$ Doubles : num

$ Characters: chr

$ Logicals : logi

$ Factors : Factor w/ 0 levels:

4. Write a R program to create a data frame from four given vectors name = c(&#39;Anastasia&#39;, &#39;Dima&#39;, &#39;Katherine&#39;,

&#39;James&#39;, &#39;Emily&#39;, &#39;Michael&#39;,

&#39;Matthew&#39;,

&#39;Laura&#39;, &#39;Kevin&#39;, &#39;Jonas&#39;) 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(&#39;yes&#39;, &#39;no&#39;, &#39;yes&#39;, &#39;no&#39;,

&#39;no&#39;, &#39;yes&#39;, &#39;yes&#39;, &#39;no&#39;, &#39;no&#39;, &#39;yes&#39;) Output:

[1] &quot;Original data frame:&quot;

[1] &quot;Anastasia&quot; &quot;Dima&quot; &quot;Katherine&quot;

&quot;James&quot; &quot;Emily&quot; &quot;Michael&quot;

[7] &quot;Matthew&quot; &quot;Laura&quot; &quot;Kevin&quot;

&quot;Jonas&quot;

[1] 12.5 9.0 16.5 12.0 9.0 20.0 14.5 13.5 8.0 19.0

[1] 1 3 2 3 2 3 1 1 2 1

[1] &quot;yes&quot; &quot;no&quot; &quot;yes&quot; &quot;no&quot;

&quot;no&quot; &quot;yes&quot; &quot;yes&quot; &quot;no&quot;

&quot;no&quot; &quot;yes&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes

# CODE:

> # create the vectors

> name <- c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Ke vin', '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')

> # create a data frame from the vectors

> df <- data.frame(name, score, attempts, qualify)

> # print the original data frame > cat("Original data frame:\n") Original data frame:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

5. Write a R program to extract specific column from a data frame using column name.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no

1. Katherine 16.5 2 yes
2. James 12.0 3 no
3. Emily 9.0 2 no
4. Michael 20.0 3 yes
5. Matthew 14.5 1 yes
6. Laura 13.5 1 no
7. Kevin 8.0 2 no
8. Jonas 19.0 1 yes

[1] &quot;Extract Specific columns:&quot; exam\_data.name exam\_data.score

1. Anastasia 12.5
2. Dima 9.0
3. Katherine 16.5
4. James 12.0
5. Emily 9.0
6. Michael 20.0
7. Matthew 14.5
8. Laura 13.5
9. Kevin 8.0 10 Jonas 19.0 CODE:

> # Create data frame

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

> exam\_data = data.frame(name, score, attempts, qualify)

> # Print original data frame > cat("Original dataframe:\n") Original dataframe: > print(exam\_data)

name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Extract specific columns

> cat("\nExtract Specific columns:\n")

Extract Specific columns:

> extracted\_data = data.frame(name = exam\_data$name, score = exam\_data$score)

> print(extracted\_data) name score

1. Anastasia 12.5
2. Dima 9.0
3. Katherine 16.5
4. James 12.0
5. Emily 9.0
6. Michael 20.0
7. Matthew 14.5
8. Laura 13.5
9. Kevin 8.0
10. Jonas 19.0

6. Write a R program to extract first two rows from a given data frame.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;Extract first two rows:&quot; name score attempts qualify

1 Anastasia 12.5 1 yes 2 Dima 9.0 3 no

CODE:

# Create the data frame

> name <- c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Ke vin', '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')

> exam\_data <- data.frame(name, score, attempts, qualify)

> # Display the original data frame > cat("Original dataframe:\n") Original dataframe: > print(exam\_data) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Extract the first two rows > cat("Extract first two rows:\n") Extract first two rows: > exam\_data[1:2, ] name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no

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

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no

1. Katherine 16.5 2 yes
2. James 12.0 3 no
3. Emily 9.0 2 no
4. Michael 20.0 3 yes
5. Matthew 14.5 1 yes
6. Laura 13.5 1 no
7. Kevin 8.0 2 no
8. Jonas 19.0 1 yes

[1] &quot;Extract 3rd and 5th rows with 1st and 3rd columns :&quot; name attempts

3 Katherine 2

5 Emily 2

CODE:

> # Create the original data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "

Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Extract 3rd and 5th rows with 1st and 3rd columns

> df\_extracted <- df[c(3, 5), c(1, 3)]

> # Print the extracted data frame

> cat("Extract 3rd and 5th rows with 1st and 3rd columns:\n") Extract 3rd and 5th rows with 1st and 3rd columns:

> print(df\_extracted) name attempts 3 Katherine 2

5 Emily 2

8. Write a R program to add a new column in a given data frame

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;New data frame after adding the &#39;country&#39; column:&quot; name score attempts qualify country

1. Anastasia 12.5 1 yes USA
2. Dima 9.0 3 no USA
3. Katherine 16.5 2 yes USA
4. James 12.0 3 no USA
5. Emily 9.0 2 no USA
6. Michael 20.0 3 yes USA
7. Matthew 14.5 1 yes USA
8. Laura 13.5 1 no USA
9. Kevin 8.0 2 no USA
10. Jonas 19.0 1 yes USA

CODE:

> # Create data frame

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

> exam\_data = data.frame(name, score, attempts, qualify)

> # Print original data frame > cat("Original dataframe:\n") Original dataframe: > print(exam\_data) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Extract specific columns

> cat("\nExtract Specific columns:\n")

Extract Specific columns:

> extracted\_data = data.frame(name = exam\_data$name, score = exam\_data$score)

> print(extracted\_data) name score 1 Anastasia 12.5

1. Dima 9.0
2. Katherine 16.5
3. James 12.0
4. Emily 9.0
5. Michael 20.0
6. Matthew 14.5
7. Laura 13.5
8. Kevin 8.0
9. Jonas 19.0

>

>

> # Create the data frame

> name <- c('Anastasia', 'Dima', 'Katherine', 'James', 'Emily', 'Michael', 'Matthew', 'Laura', 'Ke vin', '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')

> exam\_data <- data.frame(name, score, attempts, qualify)

> # Display the original data frame > cat("Original dataframe:\n") Original dataframe: > print(exam\_data) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Extract the first two rows > cat("Extract first two rows:\n") Extract first two rows: > exam\_data[1:2, ] name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no

>

>

> # Create the original data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "

Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Extract 3rd and 5th rows with 1st and 3rd columns

> df\_extracted <- df[c(3, 5), c(1, 3)]

> # Print the extracted data frame

> cat("Extract 3rd and 5th rows with 1st and 3rd columns:\n") Extract 3rd and 5th rows with 1st and 3rd columns:

> print(df\_extracted) name attempts 3 Katherine 2

5 Emily 2

>

>

> # create the original data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily",

+ "Michael", "Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

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

+ qualify = c("yes", "no", "yes", "no", "no", "yes", "yes", "no", "no", "yes"))

> # add a new column "country" with value "USA"

> df$country <- "USA"

> # print the original and new data frames > cat("Original dataframe:\n") Original dataframe: > print(df) name score attempts qualify country

1. Anastasia 12.5 1 yes USA
2. Dima 9.0 3 no USA
3. Katherine 16.5 2 yes USA
4. James 12.0 3 no USA
5. Emily 9.0 2 no USA
6. Michael 20.0 3 yes USA
7. Matthew 14.5 1 yes USA
8. Laura 13.5 1 no USA
9. Kevin 8.0 2 no USA
10. Jonas 19.0 1 yes USA

> cat("\nNew data frame after adding the 'country' column:\n")

New data frame after adding the 'country' column:

> print(df) name score attempts qualify country

1. Anastasia 12.5 1 yes USA
2. Dima 9.0 3 no USA
3. Katherine 16.5 2 yes USA
4. James 12.0 3 no USA
5. Emily 9.0 2 no USA
6. Michael 20.0 3 yes USA
7. Matthew 14.5 1 yes USA
8. Laura 13.5 1 no USA
9. Kevin 8.0 2 no USA
10. Jonas 19.0 1 yes USA

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

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes

1. James 12.0 3 no
2. Emily 9.0 2 no
3. Michael 20.0 3 yes
4. Matthew 14.5 1 yes
5. Laura 13.5 1 no
6. Kevin 8.0 2 no
7. Jonas 19.0 1 yes

[1] &quot;After adding new row(s) to an existing data frame:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes
11. Robert 10.5 1 yes 12 Sophia 9.0 3 no

CODE:

> # create original data frame

> df <- data.frame(

+ name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "Matthew", "Lau ra", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # create new rows to add

> new\_rows <- data.frame(

+ name = c("Robert", "Sophia"),

+ score = c(10.5, 9.0),

+ attempts = c(1, 3),

+ qualify = c("yes", "no")

+ )

> # add new rows to existing data frame

> df <- rbind(df, new\_rows)

> # print updated data frame

> cat("After adding new row(s) to an existing data frame:\n") After adding new row(s) to an existing data frame:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes
11. Robert 10.5 1 yes
12. Sophia 9.0 3 no

10. Write a R program to drop column(s) by name from a given data frame.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes score attempts

1 12.5 1 2 9.0 3

1. 16.5 2
2. 12.0 3 5 9.0 2
3. 20.0 3
4. 14.5 1
5. 13.5 1
6. 8.0 2
7. 19.0 1 CODE:

> # create the original data frame

> df <- data.frame(

+ name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "Matthew", "Lau ra", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame

> cat("Original dataframe:\n")

Original dataframe: > print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # drop column(s) by name

> df <- df[, !names(df) %in% c("qualify")]

> # print the resulting data frame

> cat("\nAfter dropping column(s) by name:\n")

After dropping column(s) by name:

> print(df) name score attempts

1. Anastasia 12.5 1
2. Dima 9.0 3
3. Katherine 16.5 2
4. James 12.0 3
5. Emily 9.0 2
6. Michael 20.0 3
7. Matthew 14.5 1
8. Laura 13.5 1
9. Kevin 8.0 2
10. Jonas 19.0 1

11. Write a R program to drop row(s) by number from a given data frame.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes name score attempts qualify

1 Anastasia 12.5 1 yes

3 Katherine 16.5 2 yes

5 Emily 9.0 2 no

1. Matthew 14.5 1 yes
2. Laura 13.5 1 no
3. Kevin 8.0 2 no 10 Jonas 19.0 1 yes

CODE:

> # Create a data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "

Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Drop row(s) by number

> df <- df[-c(2, 4), ]

> # Print the modified data frame

> cat("Modified dataframe after dropping row(s):\n") Modified dataframe after dropping row(s):

> print(df) name score attempts qualify 1 Anastasia 12.5 1 yes

3 Katherine 16.5 2 yes

1. Emily 9.0 2 no
2. Michael 20.0 3 yes
3. Matthew 14.5 1 yes
4. Laura 13.5 1 no
5. Kevin 8.0 2 no
6. Jonas 19.0 1 yes

12. Write a R program to sort a given data frame by multiple column(s).

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;dataframe after sorting &#39;name&#39; and &#39;score&#39; columns:&quot;

name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no

5 Emily 9.0 2 no

4 James 12.0 3 no

10 Jonas 19.0 1 yes

3 Katherine 16.5 2 yes

9 Kevin 8.0 2 no

8 Laura 13.5 1 no

7 Matthew 14.5 1 yes

6 Michael 20.0 3 yes

CODE:

> df <- data.frame(

+ name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "Matthew", "Lau ra", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # sort the data frame by 'name' and 'score' columns

> df\_sorted <- df[order(df$name, df$score), ]

> # print the sorted data frame

> cat("dataframe after sorting 'name' and 'score' columns:\n") dataframe after sorting 'name' and 'score' columns:

> print(df\_sorted) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no

5 Emily 9.0 2 no

4 James 12.0 3 no

10 Jonas 19.0 1 yes

3 Katherine 16.5 2 yes

9 Kevin 8.0 2 no

8 Laura 13.5 1 no

7 Matthew 14.5 1 yes

6 Michael 20.0 3 yes

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

Output:

[1] &quot;Left outer Join:&quot; numid

# 1 10

# 2 11

# 3 12

# 4 14

[1] &quot;Right outer Join:&quot; numid

1. 11
2. 12
3. 13
4. 15

[1] &quot;Outer Join:&quot; numid

1. 10
2. 11
3. 12
4. 13
5. 14
6. 15

[1] &quot;Cross Join:&quot; numid.xnumid.y 1 12 13

1. 14 13
2. 10 13
3. 11 13

1. 12 15
2. 14 15
3. 10 15
4. 11 15
5. 12 11
6. 14 11
7. 10 11
8. 11 11
9. 12 12
10. 14 12
11. 10 12
12. 11 12 CODE:

> # create first data frame

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

+ value = c(100, 200, 300, 400))

> # create second data frame

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

+ price = c(10, 20, 30, 40))

> # perform left outer join

> left\_join <- merge(df1, df2, by = "numid", all.x = TRUE) > cat("Left outer Join:\n") Left outer Join: > print(left\_join) numid value price 1 10 100 NA

1. 11 200 10
2. 12 300 20
3. 14 400 NA

> # perform right outer join

> right\_join <- merge(df1, df2, by = "numid", all.y = TRUE) > cat("Right outer Join:\n") Right outer Join: > print(right\_join) numid value price

1. 11 200 10
2. 12 300 20
3. 13 NA 30
4. 15 NA 40

> # perform outer join

> outer\_join <- merge(df1, df2, by = "numid", all = TRUE) > cat("Outer Join:\n") Outer Join: > print(outer\_join) numid value price 1 10 100 NA

1. 11 200 10
2. 12 300 20
3. 13 NA 30
4. 14 400 NA
5. 15 NA 40

> # perform cross join

> cross\_join <- merge(df1, df2, by = NULL)

> cat("Cross Join:\n") Cross Join: > print(cross\_join)

numid.x value numid.y price

1. 10 100 11 10
2. 11 200 11 10
3. 12 300 11 10
4. 14 400 11 10
5. 10 100 12 20
6. 11 200 12 20
7. 12 300 12 20
8. 14 400 12 20
9. 10 100 13 30
10. 11 200 13 30
11. 12 300 13 30
12. 14 400 13 30
13. 10 100 15 40
14. 11 200 15 40
15. 12 300 15 40
16. 14 400 15 40

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

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;After removing NA with 3, the said dataframe becomes:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 3 no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes CODE:

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "

Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

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

+ qualify = c("yes", "no", "yes", "no", "no", "yes", "yes", "no", "no", "yes"))

> # Print the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Rename the 'name' column to 'student\_name'

> colnames(df)[1] <- "student\_name"

> # Print the updated data frame

> cat("\nChange column-name 'name' to 'student\_name' of the said dataframe:\n")

Change column-name 'name' to 'student\_name' of the said dataframe:

> print(df)

student\_name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

15. Write a R program to change a column name of a given data frame.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes

1. James 12.0 NA no
2. Emily 9.0 2 no
3. Michael 20.0 NA yes
4. Matthew 14.5 1 yes
5. Laura 13.5 NA no
6. Kevin 8.0 2 no
7. Jonas 19.0 1 yes

[1] &quot;Change column-name &#39;name&#39; to &#39;student\_name&#39; of the said dataframe:&quot; student\_name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes CODE:

# create the original data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "

Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

+ qualify = c("yes", "no", "yes", "no", "no", "yes", "yes", "no", "no", "yes"))

> # display the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes 7 Matthew 14.5 1 yes
7. Laura 13.5 NA no
8. Kevin 8.0 2 no
9. Jonas 19.0 1 yes

> # change the column names

> names(df)[1:3] <- c("student\_name", "avg\_score", "attempts")

> # display the updated data frame

> cat("Change more than one column name of the said dataframe:\n") Change more than one column name of the said dataframe:

> print(df)

student\_name avg\_score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes 7 Matthew 14.5 1 yes
7. Laura 13.5 NA no
8. Kevin 8.0 2 no
9. Jonas 19.0 1 yes

16. Write a R program to change more than one column name of a given data frame. Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;Change more than one column name of the said dataframe:&quot; student\_nameavg\_score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes CODE:

> # create the original data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "

Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

+ qualify = c("yes", "no", "yes", "no", "no", "yes", "yes", "no", "no", "yes"))

> # display the original data frame > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes 7 Matthew 14.5 1 yes
7. Laura 13.5 NA no
8. Kevin 8.0 2 no
9. Jonas 19.0 1 yes

> # change the column names

> names(df)[1:3] <- c("student\_name", "avg\_score", "attempts")

> # display the updated data frame

> cat("Change more than one column name of the said dataframe:\n") Change more than one column name of the said dataframe:

> print(df)

student\_name avg\_score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes 7 Matthew 14.5 1 yes
7. Laura 13.5 NA no
8. Kevin 8.0 2 no
9. Jonas 19.0 1 yes

17. Write a R program to select some random rows from a given data frame.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;Select three random rows of the said dataframe:&quot; name score attempts qualify

10 Jonas 19.0 1 yes

7 Matthew 14.5 1 yes 4 James 12.0 3 no

CODE:

> # Create the data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily",

+ "Michael", "Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame > cat("Original dataframe:\n") Original dataframe: > print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Set seed to make the results reproducible

> set.seed(123)

> # Randomly select three rows from the data frame

> selected\_rows <- sample(nrow(df), 3)

> # Print the randomly selected rows

> cat("\nSelect three random rows of the said dataframe:\n")

Select three random rows of the said dataframe:

> print(df[selected\_rows, ]) name score attempts qualify

3 Katherine 16.5 2 yes

10 Jonas 19.0 1 yes

2 Dima 9.0 3 no

18. Write a R program to reorder an given data frame by column name.

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;Reorder by column name:&quot; name attempts score qualify

1. Anastasia 1 12.5 yes
2. Dima 3 9.0 no
3. Katherine 2 16.5 yes
4. James 3 12.0 no
5. Emily 2 9.0 no
6. Michael 3 20.0 yes
7. Matthew 1 14.5 yes
8. Laura 1 13.5 no

1. Kevin 2 8.0 no 10 Jonas 1 19.0 yes CODE:

> # Create the data frame

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily",

+ "Michael", "Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ 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 the original data frame > cat("Original dataframe:\n") Original dataframe: > print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

> # Reorder the data frame by column name

> reordered\_df <- df[, c("name", "attempts", "score", "qualify")]

> # Print the reordered data frame

> cat("\nReorder by column name:\n")

Reorder by column name: > print(reordered\_df)

name attempts score qualify

1. Anastasia 1 12.5 yes
2. Dima 3 9.0 no
3. Katherine 2 16.5 yes
4. James 3 12.0 no
5. Emily 2 9.0 no
6. Michael 3 20.0 yes
7. Matthew 1 14.5 yes
8. Laura 1 13.5 no
9. Kevin 2 8.0 no
10. Jonas 1 19.0 yes

19. Write a R program to compare two data frames to find the elements in first data frame that are not present in second data frame.

Output:

[1] &quot;Original Dataframes&quot;

[1] &quot;a&quot; &quot;b&quot; &quot;c&quot; &quot;d&quot;

&quot;e&quot;

[1] &quot;d&quot; &quot;e&quot; &quot;f&quot; &quot;g&quot; [1] &quot;Data in first dataframe that are not present in second dataframe:&quot;

[1] &quot;a&quot; &quot;b&quot; &quot;c&quot; CODE:

> # Create the two data frames

> df1 <- data.frame(a = c("a", "b", "c", "d", "e"))

> df2 <- data.frame(a = c("d", "e", "f", "g"))

> # Print the original data frames

> cat("Original Dataframes\n")

Original Dataframes

> print(df1$a)

[1] "a" "b" "c" "d" "e"

> print(df2$a)

[1] "d" "e" "f" "g"

> # Find the elements in the first dataframe that are not present in the second dataframe

> diff\_df <- setdiff(df1$a, df2$a)

> # Print the difference between the data frames

> cat("Data in first dataframe that are not present in second dataframe:\n") Data in first dataframe that are not present in second dataframe:

> print(diff\_df)

[1] "a" "b" "c"

20. Write a R program to find elements which are present in two given data frames. Output:

[1] &quot;Original Dataframes&quot;

[1] &quot;a&quot; &quot;b&quot; &quot;c&quot; &quot;d&quot;

&quot;e&quot;

[1] &quot;d&quot; &quot;e&quot; &quot;f&quot; &quot;g&quot;

[1] &quot;Elements which are present in both dataframe:&quot; [1] &quot;d&quot; &quot;e&quot; CODE:

> # Create the two data frames

> df1 <- data.frame(a = c("a", "b", "c", "d", "e"))

> df2 <- data.frame(a = c("d", "e", "f", "g"))

> # Print the original data frames

> cat("Original Dataframes\n")

Original Dataframes

> print(df1$a)

[1] "a" "b" "c" "d" "e"

> print(df2$a)

[1] "d" "e" "f" "g"

> # Find the elements which are present in both data frames

> common\_df <- intersect(df1$a, df2$a)

> # Print the common elements

> cat("Elements which are present in both data frames:\n") Elements which are present in both data frames:

> print(common\_df)

[1] "d" "e"

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

Output:

[1] &quot;Original Dataframes&quot;

[1] &quot;a&quot; &quot;b&quot; &quot;c&quot; &quot;d&quot;

&quot;e&quot;

[1] &quot;d&quot; &quot;e&quot; &quot;f&quot; &quot;g&quot; [1] &quot;Find elements come only once that are common to both given dataframes:&quot;

[1] &quot;a&quot; &quot;b&quot; &quot;c&quot; &quot;d&quot; &quot;e&quot; &quot;f&quot; &quot;g&quot; CODE:

> # Create the two data frames

> df1 <- data.frame(a = c("a", "b", "c", "d", "e"))

> df2 <- data.frame(a = c("d", "e", "f", "g"))

> # Print the original data frames

> cat("Original Dataframes\n")

Original Dataframes

> print(df1$a)

[1] "a" "b" "c" "d" "e"

> print(df2$a)

[1] "d" "e" "f" "g"

> # Find the elements that are common to both data frames and occur only once > common\_once\_df <- df1$a[df1$a %in% df2$a & !duplicated(df1$a[df1$a %in% df2$a])] Warning message:

In df1$a %in% df2$a & !duplicated(df1$a[df1$a %in% df2$a]) : longer object length is not a multiple of shorter object length

> # Print the common elements that occur only once

> cat("Find elements come only once that are common to both given dataframes:\n") Find elements come only once that are common to both given dataframes:

> print(common\_once\_df)

[1] "d" "e"

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

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 3 no
3. Katherine 16.5 2 yes
4. James 12.0 3 no
5. Emily 9.0 2 no
6. Michael 20.0 3 yes
7. Matthew 14.5 1 yes
8. Laura 13.5 1 no
9. Kevin 8.0 2 no 10 Jonas 19.0 1 yes size isdir mode mtimectime

data.rda 344 FALSE 644 2018-10-25 12:06:09 2018-10-25 12:06:09 atimeuidgidunamegrname

data.rda 2018-10-25 12:06:09 1000 1000 trinket trinket CODE:

> df <- data.frame(name = c("Anastasia", "Dima", "Katherine", "James", "Emily",

+ "Michael", "Matthew", "Laura", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

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

+ qualify = c("yes", "no", "yes", "no", "no", "yes", "yes", "no", "no", "yes"))

> # save the data frame in a file

> save(df, file = "data.rda")

> # display information about the file

> file.info("data.rda") size isdir mode data.rda 297 FALSE 666

mtime

data.rda 2023-03-22 10:49:43

ctime

data.rda 2023-03-22 10:49:43

atime exe

data.rda 2023-03-22 10:49:43 no

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

Output:

[1] &quot;Original dataframe:&quot; name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes
7. Matthew 14.5 1 yes
8. Laura 13.5 NA no
9. Kevin 8.0 2 no
10. Jonas 19.0 1 yes

[1] &quot;The number of NA values in attempts column:&quot;

[1] 4

CODE:

> # create the data frame

> df <- data.frame(

+ name = c("Anastasia", "Dima", "Katherine", "James", "Emily", "Michael", "Matthew", "Lau ra", "Kevin", "Jonas"),

+ score = c(12.5, 9.0, 16.5, 12.0, 9.0, 20.0, 14.5, 13.5, 8.0, 19.0),

+ attempts = c(1, NA, 2, NA, 2, NA, 1, NA, 2, 1),

+ qualify = c("yes", "no", "yes", "no", "no", "yes", "yes", "no", "no", "yes") + )

> # count the number of NA values in the 'attempts' column

> n\_na <- sum(is.na(df$attempts))

> # print the original data frame and the result > cat("Original dataframe:\n") Original dataframe:

> print(df) name score attempts qualify

1. Anastasia 12.5 1 yes
2. Dima 9.0 NA no
3. Katherine 16.5 2 yes
4. James 12.0 NA no
5. Emily 9.0 2 no
6. Michael 20.0 NA yes 7 Matthew 14.5 1 yes
7. Laura 13.5 NA no
8. Kevin 8.0 2 no
9. Jonas 19.0 1 yes

> cat("The number of NA values in attempts column:\n") The number of NA values in attempts column:

> print(n\_na)

[1] 4

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

Output:

[1] &quot;Original data frame:&quot; a b 1 10 10

1. 20 30
2. 10 10

# 4 10 20

5 40 0

# 6 50 50

1. 20 30
2. 30 30

[1] &quot;Duplicate elements of the said data frame:&quot;

[1] FALSE FALSE TRUE FALSE FALSEFALSE TRUE FALSE [1] &quot;Unique rows of the said data frame:&quot; a b 1 10 10

2 20 30

1. 10 20
2. 40 0
3. 50 50

[8 30 30 CODE: > # display the original data frame > cat("Original data frame:\n") Original data frame: > print(df) a 5](#_Toc100794)

[1 10 20](#_Toc100795)

[2 20 20](#_Toc100796)

[3 10 20](#_Toc100797)

[4 10 20](#_Toc100798)

[5 40 0 35](#_Toc100799)

[6 50 50 7 20 35](#_Toc100800)

> # create two vectors

> vec1 <- c(10, 20, 10, 10, 40, 50, 20, 30)

> vec2 <- c(10, 30, 10, 20, 0, 50, 30, 30)

> # create a data frame from the vectors

> df <- data.frame(a = vec1, b = vec2)

8 30 30

> # find duplicate elements in the data frame

> dup <- duplicated(df)

> # display the duplicated elements

> cat("\nDuplicate elements of the said data frame:\n")

Duplicate elements of the said data frame:

> print(dup)

[1] FALSE FALSE TRUE FALSE FALSE FALSE

[7] TRUE FALSE

> # find unique rows in the data frame

> unique\_df <- unique(df)

> # display the unique rows

> cat("\nUnique rows of the said data frame:\n")

Unique rows of the said data frame:

> print(unique\_df)

a b

1. 10 10
2. 20 30
3. 10 20
4. 40 0
5. 50 50

8 30 30

25. Write a R program to call the (built-in) dataset airquality. Check whether it

is a

data frame or not? Order the entire data frame by the first and second column.

Output:

[1] &quot;Original data: Daily air quality measurements in New York, May to

September

1973.&quot;

[1] &quot;data.frame&quot;

Ozone Solar.R Wind Temp Month Day

1. 41 190 7.4 67 5 1
2. 36 118 8.0 72 5 2
3. 12 149 12.6 74 5 3
4. 18 313 11.5 62 5 4
5. NA NA 14.3 56 5 5
6. 28 NA 14.9 66 5 6
7. 23 299 8.6 65 5 7
8. 19 99 13.8 59 5 8
9. 8 19 20.1 61 5 9
10. NA 194 8.6 69 5 10

[1] &quot;Order the entire data frame by the first and second column:&quot;

Ozone Solar.R Wind Temp Month Day

21 1 8 9.7 59 5 21

23 4 25 9.7 61 5 23 18 6 78 18.4 57 5 18

...........

119 NA 153 5.7 88 8 27 150 NA 145 13.2 77 9 27

CODE:

> # Call the built-in dataset airquality

> data(airquality)

> # Check whether it is a data frame or not

> cat("Original data: Daily air quality measurements in New York, May to September 1973.\n

")

Original data: Daily air quality measurements in New York, May to September 1973.

> cat(class(airquality), "\n")

data.frame

> # Order the entire data frame by the first and second column > cat("Order the entire data frame by the first and second column:\n") Order the entire data frame by the first and second column:

> airquality\_sorted <- airquality[order(airquality$Ozone, airquality$Solar.R),]

> print(airquality\_sorted)

Ozone Solar.R Wind Temp Month Day

21 1 8 9.7 59 5 21

23 4 25 9.7 61 5 23

18 6 78 18.4 57 5 18

76 7 48 14.3 80 7 15

147 7 49 10.3 69 9 24

11 7 NA 6.9 74 5 11

9 8 19 20.1 61 5 9

94 9 24 13.8 81 8 2

137 9 24 10.9 71 9 14

114 9 36 14.3 72 8 22

73 10 264 14.3 73 7 12

20 11 44 9.7 62 5 20

13 11 290 9.2 66 5 13

22 11 320 16.6 73 5 22

50 12 120 11.5 73 6 19

3 12 149 12.6 74 5 3

141 13 27 10.3 76 9 18

138 13 112 11.5 71 9 15

51 13 137 10.3 76 6 20

144 13 238 12.6 64 9 21

148 14 20 16.6 63 9 25

151 14 191 14.3 75 9 28

14 14 274 10.9 68 5 14

16 14 334 11.5 64 5 16

82 16 7 6.9 74 7 21

95 16 77 7.4 82 8 3

143 16 201 8.0 82 9 20

12 16 256 9.7 69 5 12

15 18 65 13.2 58 5 15

152 18 131 8.0 76 9 29

140 18 224 13.8 67 9 17

4 18 313 11.5 62 5 4

8 19 99 13.8 59 5 8

49 20 37 9.2 65 6 18

87 20 81 8.6 82 7 26

153 20 223 11.5 68 9 30

130 20 252 10.9 80 9 7

47 21 191 14.9 77 6 16

132 21 230 10.9 75 9 9

113 21 259 15.5 77 8 21

135 21 259 15.5 76 9 12

108 22 71 10.3 77 8 16

28 23 13 12.0 67 5 28

145 23 14 9.2 71 9 22

110 23 115 7.4 76 8 18

44 23 148 8.0 82 6 13

131 23 220 10.3 78 9 8

7 23 299 8.6 65 5 7

142 24 238 10.3 68 9 19

133 24 259 9.7 73 9 10

74 27 175 14.9 81 7 13

136 28 238 6.3 77 9 13

105 28 273 11.5 82 8 13

6 28 NA 14.9 66 5 6

38 29 127 9.7 82 6 7

149 30 193 6.9 70 9 26

19 30 322 11.5 68 5 19

111 31 244 10.9 78 8 19

24 32 92 12.0 61 5 24

129 32 92 15.5 84 9 6

64 32 236 9.2 81 7 3

17 34 307 12.0 66 5 17

78 35 274 10.3 82 7 17 97 35 NA 7.4 85 8 5

2 36 118 8.0 72 5 2

146 36 139 10.3 81 9 23

31 37 279 7.4 76 5 31

48 37 284 20.7 72 6 17

93 39 83 6.9 81 8 1

41 39 323 11.5 87 6 10

67 40 314 10.9 83 7 6

1 41 190 7.4 67 5 1

112 44 190 10.3 78 8 20

104 44 192 11.5 86 8 12

134 44 236 14.9 81 9 11

116 45 212 9.7 79 8 24

29 45 252 14.9 81 5 29

139 46 237 6.9 78 9 16

128 47 95 7.4 87 9 5

77 48 260 6.9 81 7 16

63 49 248 9.2 85 7 2

90 50 275 7.4 86 7 29

88 52 82 12.0 86 7 27

109 59 51 6.3 79 8 17

92 59 254 9.2 81 7 31

79 61 285 6.3 84 7 18

81 63 220 11.5 85 7 20

66 64 175 4.6 83 7 5

91 64 253 7.4 83 7 30

106 65 157 9.7 80 8 14

98 66 NA 4.6 87 8 6

40 71 291 13.8 90 6 9

126 73 183 2.8 93 9 3

118 73 215 8.0 86 8 26

120 76 203 9.7 97 8 28

68 77 276 5.1 88 7 7

125 78 197 5.1 92 9 2

96 78 NA 6.9 86 8 4

80 79 187 5.1 87 7 19

85 80 294 8.6 86 7 24

89 82 213 7.4 88 7 28

122 84 237 6.3 96 8 30

71 85 175 7.4 89 7 10

123 85 188 6.3 94 8 31

100 89 229 10.3 90 8 8

127 91 189 4.6 93 9 4

124 96 167 6.9 91 9 1

1. 97 267 6.3 92 7 8
2. 97 272 5.7 92 7 9

86 108 223 8.0 85 7 25 101 110 207 8.0 90 8 9

30 115 223 5.7 79 5 30

121 118 225 2.3 94 8 29

99 122 255 4.0 89 8 7

62 135 269 4.1 84 7 1

117 168 238 3.4 81 8 25

60 NA 31 14.9 77 6 29

58 NA 47 10.3 73 6 27

53 NA 59 1.7 76 6 22

107 NA 64 11.5 79 8 15

25 NA 66 16.6 57 5 25

54 NA 91 4.6 76 6 23

59 NA 98 11.5 80 6 28

65 NA 101 10.9 84 7 4

57 NA 127 8.0 78 6 26

56 NA 135 8.0 75 6 25

103 NA 137 11.5 86 8 11

61 NA 138 8.0 83 6 30

72 NA 139 8.6 82 7 11

150 NA 145 13.2 77 9 27

52 NA 150 6.3 77 6 21

119 NA 153 5.7 88 8 27

35 NA 186 9.2 84 6 4

10 NA 194 8.6 69 5 10

36 NA 220 8.6 85 6 5

102 NA 222 8.6 92 8 10

34 NA 242 16.1 67 6 3

43 NA 250 9.2 92 6 12

55 NA 250 6.3 76 6 24

115 NA 255 12.6 75 8 23

83 NA 258 9.7 81 7 22

42 NA 259 10.9 93 6 11

37 NA 264 14.3 79 6 6

26 NA 266 14.9 58 5 26

39 NA 273 6.9 87 6 8

1. NA 286 8.6 78 6 1
2. 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

26. Write a R program to call the (built-in) dataset airquality. Remove the variables

&#39;Solar.R&#39; and &#39;Wind&#39; and display the data frame.

Output:

[1] &quot;Original data: Daily air quality measurements in New York, May to

September

1973.&quot;

Ozone Solar.R Wind Temp Month Day

1. 41 190 7.4 67 5 1
2. 36 118 8.0 72 5 2
3. 12 149 12.6 74 5 3
4. 18 313 11.5 62 5 4
5. NA NA 14.3 56 5 5

.........

1. 18 131 8.0 76 9 29
2. 20 223 11.5 68 9 30

[1] &quot;data.frame after removing &#39;Solar.R&#39; and &#39;Wind&#39; variables:&quot;

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

.........

1. 18 76 9 29
2. 20 68 9 30

CODE:

> # Call the built-in dataset airquality

> data(airquality)

> # Display the original data frame

> cat("Original data: Daily air quality measurements in New York, May to September 1973.\n

")

Original data: Daily air quality measurements in New York, May to September 1973.

> print(airquality)

Ozone Solar.R Wind Temp Month Day

1. 41 190 7.4 67 5 1
2. 36 118 8.0 72 5 2
3. 12 149 12.6 74 5 3
4. 18 313 11.5 62 5 4
5. NA NA 14.3 56 5 5
6. 28 NA 14.9 66 5 6
7. 23 299 8.6 65 5 7
8. 19 99 13.8 59 5 8
9. 8 19 20.1 61 5 9
10. NA 194 8.6 69 5 10
11. 7 NA 6.9 74 5 11
12. 16 256 9.7 69 5 12
13. 11 290 9.2 66 5 13
14. 14 274 10.9 68 5 14
15. 18 65 13.2 58 5 15
16. 14 334 11.5 64 5 16
17. 34 307 12.0 66 5 17
18. 6 78 18.4 57 5 18
19. 30 322 11.5 68 5 19
20. 11 44 9.7 62 5 20
21. 1 8 9.7 59 5 21
22. 11 320 16.6 73 5 22
23. 4 25 9.7 61 5 23
24. 32 92 12.0 61 5 24
25. NA 66 16.6 57 5 25
26. NA 266 14.9 58 5 26
27. NA NA 8.0 57 5 27
28. 23 13 12.0 67 5 28
29. 45 252 14.9 81 5 29
30. 115 223 5.7 79 5 30
31. 37 279 7.4 76 5 31
32. NA 286 8.6 78 6 1
33. NA 287 9.7 74 6 2
34. NA 242 16.1 67 6 3
35. NA 186 9.2 84 6 4
36. NA 220 8.6 85 6 5
37. NA 264 14.3 79 6 6
38. 29 127 9.7 82 6 7
39. NA 273 6.9 87 6 8
40. 71 291 13.8 90 6 9
41. 39 323 11.5 87 6 10
42. NA 259 10.9 93 6 11
43. NA 250 9.2 92 6 12
44. 23 148 8.0 82 6 13
45. NA 332 13.8 80 6 14
46. NA 322 11.5 79 6 15
47. 21 191 14.9 77 6 16
48. 37 284 20.7 72 6 17
49. 20 37 9.2 65 6 18
50. 12 120 11.5 73 6 19 51 13 137 10.3 76 6 20
51. NA 150 6.3 77 6 21
52. NA 59 1.7 76 6 22
53. NA 91 4.6 76 6 23
54. NA 250 6.3 76 6 24
55. NA 135 8.0 75 6 25
56. NA 127 8.0 78 6 26
57. NA 47 10.3 73 6 27
58. NA 98 11.5 80 6 28
59. NA 31 14.9 77 6 29
60. NA 138 8.0 83 6 30
61. 135 269 4.1 84 7 1
62. 49 248 9.2 85 7 2
63. 32 236 9.2 81 7 3
64. NA 101 10.9 84 7 4
65. 64 175 4.6 83 7 5
66. 40 314 10.9 83 7 6
67. 77 276 5.1 88 7 7
68. 97 267 6.3 92 7 8
69. 97 272 5.7 92 7 9
70. 85 175 7.4 89 7 10
71. NA 139 8.6 82 7 11
72. 10 264 14.3 73 7 12
73. 27 175 14.9 81 7 13
74. NA 291 14.9 91 7 14
75. 7 48 14.3 80 7 15
76. 48 260 6.9 81 7 16
77. 35 274 10.3 82 7 17
78. 61 285 6.3 84 7 18
79. 79 187 5.1 87 7 19
80. 63 220 11.5 85 7 20
81. 16 7 6.9 74 7 21
82. NA 258 9.7 81 7 22
83. NA 295 11.5 82 7 23
84. 80 294 8.6 86 7 24
85. 108 223 8.0 85 7 25
86. 20 81 8.6 82 7 26
87. 52 82 12.0 86 7 27
88. 82 213 7.4 88 7 28
89. 50 275 7.4 86 7 29
90. 64 253 7.4 83 7 30
91. 59 254 9.2 81 7 31
92. 39 83 6.9 81 8 1
93. 9 24 13.8 81 8 2
94. 16 77 7.4 82 8 3
95. 78 NA 6.9 86 8 4
96. 35 NA 7.4 85 8 5 98 66 NA 4.6 87 8 6
97. 122 255 4.0 89 8 7
98. 89 229 10.3 90 8 8
99. 110 207 8.0 90 8 9
100. NA 222 8.6 92 8 10
101. NA 137 11.5 86 8 11
102. 44 192 11.5 86 8 12
103. 28 273 11.5 82 8 13
104. 65 157 9.7 80 8 14
105. NA 64 11.5 79 8 15
106. 22 71 10.3 77 8 16
107. 59 51 6.3 79 8 17
108. 23 115 7.4 76 8 18
109. 31 244 10.9 78 8 19
110. 44 190 10.3 78 8 20
111. 21 259 15.5 77 8 21
112. 9 36 14.3 72 8 22
113. NA 255 12.6 75 8 23
114. 45 212 9.7 79 8 24
115. 168 238 3.4 81 8 25
116. 73 215 8.0 86 8 26
117. NA 153 5.7 88 8 27
118. 76 203 9.7 97 8 28
119. 118 225 2.3 94 8 29
120. 84 237 6.3 96 8 30
121. 85 188 6.3 94 8 31
122. 96 167 6.9 91 9 1
123. 78 197 5.1 92 9 2
124. 73 183 2.8 93 9 3
125. 91 189 4.6 93 9 4
126. 47 95 7.4 87 9 5
127. 32 92 15.5 84 9 6
128. 20 252 10.9 80 9 7
129. 23 220 10.3 78 9 8
130. 21 230 10.9 75 9 9
131. 24 259 9.7 73 9 10
132. 44 236 14.9 81 9 11
133. 21 259 15.5 76 9 12
134. 28 238 6.3 77 9 13
135. 9 24 10.9 71 9 14
136. 13 112 11.5 71 9 15
137. 46 237 6.9 78 9 16
138. 18 224 13.8 67 9 17
139. 13 27 10.3 76 9 18
140. 24 238 10.3 68 9 19
141. 16 201 8.0 82 9 20
142. 13 238 12.6 64 9 21 145 23 14 9.2 71 9 22
143. 36 139 10.3 81 9 23
144. 7 49 10.3 69 9 24
145. 14 20 16.6 63 9 25
146. 30 193 6.9 70 9 26
147. NA 145 13.2 77 9 27
148. 14 191 14.3 75 9 28
149. 18 131 8.0 76 9 29
150. 20 223 11.5 68 9 30

> # Remove the variables 'Solar.R' and 'Wind'

> airquality\_new <- airquality[, c('Ozone', 'Temp', 'Month', 'Day')]

> # Display the data frame after removing 'Solar.R' and 'Wind' variables > cat("data.frame after removing 'Solar.R' and 'Wind' variables:\n") data.frame after removing 'Solar.R' and 'Wind' variables:

> print(airquality\_new)

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
11. 7 74 5 11
12. 16 69 5 12
13. 11 66 5 13
14. 14 68 5 14
15. 18 58 5 15
16. 14 64 5 16
17. 34 66 5 17
18. 6 57 5 18
19. 30 68 5 19
20. 11 62 5 20
21. 1 59 5 21
22. 11 73 5 22
23. 4 61 5 23
24. 32 61 5 24
25. NA 57 5 25
26. NA 58 5 26
27. NA 57 5 27
28. 23 67 5 28
29. 45 81 5 29
30. 115 79 5 30
31. 37 76 5 31
32. NA 78 6 1
33. NA 74 6 2
34. NA 67 6 3
35. NA 84 6 4
36. NA 85 6 5
37. NA 79 6 6
38. 29 82 6 7
39. NA 87 6 8
40. 71 90 6 9
41. 39 87 6 10
42. NA 93 6 11
43. NA 92 6 12
44. 23 82 6 13
45. NA 80 6 14
46. NA 79 6 15
47. 21 77 6 16
48. 37 72 6 17
49. 20 65 6 18
50. 12 73 6 19
51. 13 76 6 20
52. NA 77 6 21
53. NA 76 6 22
54. NA 76 6 23
55. NA 76 6 24
56. NA 75 6 25
57. NA 78 6 26
58. NA 73 6 27
59. NA 80 6 28
60. NA 77 6 29
61. NA 83 6 30
62. 135 84 7 1
63. 49 85 7 2
64. 32 81 7 3
65. NA 84 7 4
66. 64 83 7 5
67. 40 83 7 6
68. 77 88 7 7
69. 97 92 7 8
70. 97 92 7 9
71. 85 89 7 10
72. NA 82 7 11
73. 10 73 7 12
74. 27 81 7 13
75. NA 91 7 14
76. 7 80 7 15
77. 48 81 7 16
78. 35 82 7 17
79. 61 84 7 18
80. 79 87 7 19
81. 63 85 7 20
82. 16 74 7 21
83. NA 81 7 22
84. NA 82 7 23 85 80 86 7 24
85. 108 85 7 25
86. 20 82 7 26
87. 52 86 7 27
88. 82 88 7 28
89. 50 86 7 29
90. 64 83 7 30
91. 59 81 7 31
92. 39 81 8 1
93. 9 81 8 2
94. 16 82 8 3
95. 78 86 8 4
96. 35 85 8 5
97. 66 87 8 6
98. 122 89 8 7
99. 89 90 8 8
100. 110 90 8 9
101. NA 92 8 10
102. NA 86 8 11
103. 44 86 8 12
104. 28 82 8 13
105. 65 80 8 14
106. NA 79 8 15
107. 22 77 8 16
108. 59 79 8 17
109. 23 76 8 18
110. 31 78 8 19
111. 44 78 8 20
112. 21 77 8 21
113. 9 72 8 22
114. NA 75 8 23
115. 45 79 8 24
116. 168 81 8 25
117. 73 86 8 26
118. NA 88 8 27
119. 76 97 8 28
120. 118 94 8 29
121. 84 96 8 30
122. 85 94 8 31
123. 96 91 9 1
124. 78 92 9 2
125. 73 93 9 3
126. 91 93 9 4
127. 47 87 9 5
128. 32 84 9 6
129. 20 80 9 7
130. 23 78 9 8
131. 21 75 9 9
132. 24 73 9 10
133. 44 81 9 11
134. 21 76 9 12
135. 28 77 9 13
136. 9 71 9 14
137. 13 71 9 15
138. 46 78 9 16
139. 18 67 9 17
140. 13 76 9 18
141. 24 68 9 19
142. 16 82 9 20
143. 13 64 9 21
144. 23 71 9 22
145. 36 81 9 23
146. 7 69 9 24
147. 14 63 9 25
148. 30 70 9 26
149. NA 77 9 27
150. 14 75 9 28
151. 18 76 9 29
152. 20 68 9 30

27. Find the difference between Data Frames and other Data Structures with example.

Solution:

Data Structure:

There is also an array data structure that extends this idea to more than two dimensions. A collection of vectors that all have the same length. This is like a matrix, except that each column can contain a different data type.

Eg:Array, Linked Lists, Stack, Queues, Trees, Graphs, Sets, Hash Tables.

Data Frame:

A data frame can be used to represent an entire data set. A data frame is a table or a

two-dimensional array-like structure in which each column contains values of one variable and each row contains one set of values from each column. Eg: Matrices ANS:

Tables, Spreadsheets, Database tables.

Example:

Let's consider an example to understand the difference between Data Frames and other Data Structures. Suppose we have a dataset containing information about students in a class, including their names, ages, grades, and subjects. We want to analyze this data and find out which students are performing well in which subjects. Here are some ways we can represent this data:

Array: We can use a three-dimensional array to represent this data, where the first dimension represents the student, the second dimension represents the subject, and the third dimension represents the variable (name, age, grade). However, this can be difficult to work with, and we would need to use complex indexing to access specific values.

Linked List: We can use a linked list to represent each student, where each node in the list contains the student's information. However, this would not allow us to easily compare or analyze data across multiple students.

Data Frame: We can use a data frame to represent this data, where each column represents a variable (name, age, grade, subject), and each row represents a student. This would allow us to easily compare and analyze data across multiple students and subjects.

In summary, while other data structures like arrays and linked lists can be used to represent data, they may not be as efficient or convenient for analyzing complex data sets like those found in a data frame.

28. How to create the data frame and print it for the employee data set.

Emp\_id = 1:5

Emp\_name =

&quot;Ricky&quot;,&quot;Danish&quot;,&quot;Mini&quot;,&quot;Ryan&quot

;,&quot;Gary&quot;

Salary = 643.3,515.2,671.0,729.0,943.25

Start\_date = &quot;2022-01-01&quot;, &quot;2021-09-23&quot;, &quot;2020-

11-15&quot;, &quot;2021-05-11&quot;,&quot;2022-03- 27&quot; CODE:

> # create the data frame

> employee\_df <- data.frame(

+ Emp\_id = 1:5,

+ Emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

+ Salary = c(643.3,515.2,671.0,729.0,943.25),

+ Start\_date = c("2022-01-01", "2021-09-23", "2020-11-15", "2021-05-11","2022-03-27")

+ )

> # print the data frame

> employee\_df

Emp\_id Emp\_name Salary Start\_date

1. 1 Ricky 643.30 2022-01-01
2. 2 Danish 515.20 2021-09-23
3. 3 Mini 671.00 2020-11-15
4. 4 Ryan 729.00 2021-05-11
5. 5 Gary 943.25 2022-03-27

29. Write the code to get the Structure of the R Data Frame.

CODE:

> # create a sample data frame

> df <- data.frame(

+ x = c(1, 2, 3),

+ y = c("A", "B", "C"),

+ z = c(TRUE, FALSE, TRUE)

+ )

> # get the structure of the data frame

> str(df)

'data.frame': 3 obs. of 3 variables:

$ x: num 1 2 3

$ y: chr "A" "B" "C"

$ z: logi TRUE FALSE TRUE

30. How to extract data from data frame for the above employee dataset.

Expected Output: emp.data.emp\_name. emp.data.salary

1. Ricky 643.30
2. Danish 515.20
3. Mini 671.00
4. Ryan 729.00 5 Gary 943.25

CODE:

> # create the data frame

> employee\_df <- data.frame(

+ Emp\_id = 1:5,

+ Emp\_name = c("Ricky","Danish","Mini","Ryan","Gary"),

+ Salary = c(643.3,515.2,671.0,729.0,943.25),

+ Start\_date = c("2022-01-01", "2021-09-23", "2020-11-15", "2021-05-11","2022-03-27")

+ )

> # extract employee names and salaries

> emp\_names <- employee\_df$Emp\_name

> emp\_salaries <- employee\_df$Salary

> # create a data frame with the extracted data

> emp\_data <- data.frame(emp\_name = emp\_names, salary = emp\_salaries)

> # print the data frame

> emp\_data emp\_name salary

1. Ricky 643.30
2. Danish 515.20
3. Mini 671.00
4. Ryan 729.00
5. Gary 943.25

31. How to extract the first two rows and then all columns in employee data frame.

Expected Output: emp\_idemp\_name salary start\_date

1 Ricky 643.3 2012-01-01 2 Danish 515.2 2013-09-23

CODE:

> employee\_df[1:2, ]

Emp\_id Emp\_name Salary Start\_date

1. 1 Ricky 643.3 2022-01-01
2. 2 Danish 515.2 2021-09-23

32. Write a code to extract 3 rd and 5 th row with 2 nd and 4 th column of the employee data.

Expected Output: emp\_namestart\_date 3 Mini 2014-11-15 5 Gary 2015-03-27 CODE:

> employee\_df[c(3,5), c(2,4)]

Emp\_name Start\_date

3 Mini 2020-11-15

5 Gary 2022-03-27

Data Reshaping:

Data reshaping means changing how data is represented in rows and column. It includes splitting, merging or interchanging the rows and columns.

Reshaping functions:

* cbind()
* rbind()
* mergr()

33. How to expand the data frame by adding rows and columns in data frame

for employee data set. Add Column: dept&lt;- c(&quot;IT&quot;,&quot;Operations&quot;,&quot;IT&quot;,&quot;HR&quot;, &quot;Finance&quot;) Expected Output:

emp\_idemp\_name salary start\_date dept 1 Ricky 643.30 2012-01-01 IT

1. Danish 515.20 2013-09-23 Operations
2. Mini 671.00 2014-11-15
3. Ryan 729.00 2014-05-11 HR
4. Gary 943.25 2015-03-27 Finance

Add Row using the second dataframe given below: emp\_id = 6:8, emp\_name = &quot;Rasmi&quot;,&quot;Pranab&quot;,&quot;Tusar&quot;,

salary =578.0,722.5,632.8, start\_date = &quot;2022-05-21&quot;,&quot;2020-07-30&quot;,&quot;2019-

06-17&quot;,

dept = &quot;IT&quot;,&quot;Operations&quot;,&quot;Fianance&quot;, Expected Output:

emp\_idemp\_name salary start\_date dept 1 Ricky 643.30 2012-01-01 IT

1. Danish 515.20 2013-09-23 Operations
2. Mini 671.00 2014-11-15 IT
3. Ryan 729.00 2014-05-11 HR
4. Gary 943.25 2015-03-27 Finance
5. Rasmi 578.00 2013-05-21 IT
6. Pranab 722.50 2013-07-30 Operations 8 Tusar 632.80 2014-06-17 Fianance CODE:

34. Write a R program to compare two data frames to find the row(s) in first data frame that are not present in second data frame.

CODE:

# create the first data frame

> df1 <- data.frame(

+ ID = c(1, 2, 3, 4, 5),

+ Name = c("John", "Sara", "David", "Sarah", "Mike")

+ )

> # create the second data frame

> df2 <- data.frame(

+ ID = c(2, 4),

+ Name = c("Sara", "Sarah")

+ )

> # compare the two data frames and find rows in df1 that are not in df2 > df1\_not\_in\_df2 <- anti\_join(df1, df2, by = c("ID", "Name")) Error in anti\_join(df1, df2, by = c("ID", "Name")) : could not find function "anti\_join"

> # print the result

> df1\_not\_in\_df2

Error: object 'df1\_not\_in\_df2' not found

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

CODE:

> # create two example data frames

> df1 <- data.frame(A = c(1, 2, 3, 4, 5),

+ B = c("apple", "banana", "cherry", "banana", "apple"))

> df2 <- data.frame(A = c(2, 4, 6),

+ B = c("banana", "apple", "orange"))

> # find elements that occur only once and are common to both data frames

> common <- intersect(df1$B, df2$B)

> result <- unique(df1$B[duplicated(df1$B) & df1$B %in% common])

> # print the result

> print(result)

[1] "banana" "apple"

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

Practice Probs

File Read and Write Functions in R

Readline() con &lt;- file(&quot;Sample.txt&quot;, &quot;r&quot;) w&lt;-readLines(con) close(con) w[1] w[2] w[3] writeline()

sample&lt;-c(&quot;Class,Alcohol,Malic acid,Ash&quot;,&quot;1,14.23,1.71,2.43&quot;,&quot;1,13.2,1.78,2.14&quot;) writeLines(sample,&quot;sample.csv&quot;)

dput() and dget(): # Create a data frame

x &lt;- data.frame(Name = &quot;Mr. A&quot;, Gender = &quot;Male&quot;,

Age=35)

#Print &#39;dput&#39; output to your R console dput(x)

#Write the &#39;dput&#39; output to a file dput(x, file = &quot;w.R&quot;)

# Now read in &#39;dput&#39; output from the file y &lt;- dget(&quot;w.R&quot;)

y

dump() x&lt;-1:10

d &lt;- data.frame(Name = &quot;Mr. A&quot;, Gender = &quot;Male&quot;, Age=35) dump(c(&quot;x&quot;, &quot;d&quot;), file = &quot;dump\_data.R&quot;) rm(x, d) #After dumping just remove the variables from environment.

source(&quot;dump\_data.R&quot;) x d str(d)

read &amp; Write

&gt; data &lt;- read.csv(&quot;employee\_data.csv&quot;, header =

TRUE,sep=&quot;,&quot;)

&gt; is.data.frame(data)

[1] TRUE

&gt; ncol(data)

[1] 9

&gt; nrow(data)

[1] 1000

&gt; sal &lt;- max(data$salary)

&gt; sal

[1] 106905

&gt;retval &lt;-subset(data, gender==&quot;M&quot;)

&gt; write.csv(retval,&quot;output.csv&quot;)

&gt; dim(retval)

[1] 610 9

CODE:

# create two vectors

> vec1 <- c("A", "B", "C", "D", "E", "F")

> vec2 <- c(1, 2, 3, 4, 5, 6)

> # create a data frame from the vectors

> df <- data.frame(vec1, vec2)

> # display the duplicated elements

> duplicated\_elements <- df[duplicated(df),] > cat("Duplicated elements:\n") Duplicated elements:

> print(duplicated\_elements)

[1] vec1 vec2

<0 rows> (or 0-length row.names)

> # display the unique rows

> unique\_rows <- unique(df)

> cat("\nUnique rows:\n")

Unique rows: > print(unique\_rows)

vec1 vec2

1. A 1
2. B 2 3 C 3
3. D 4
4. E 5
5. F 6