**• Write a program for creation of vectors and perform operations on vectors in R.**

# Creating vectors

vector1 <- c(1, 2, 3, 4, 5)

vector2 <- c(5, 4, 3, 2, 1)

# Vector addition

sum\_vector <- vector1 + vector2

# Vector subtraction

diff\_vector <- vector1 - vector2

# Scalar multiplication

scalar <- 2

scaled\_vector <- vector1 \* scalar

# Vector dot product

dot\_product <- sum(vector1 \* vector2)

# Vector cross product (for 3D vectors)

# This requires the 'pracma' package, which you can install with: install.packages("pracma")

# library(pracma)

# cross\_product <- cross(vector1, vector2)

# Printing the results

cat("Vector 1: ", vector1, "\n")

cat("Vector 2: ", vector2, "\n")

cat("Vector 1 + Vector 2: ", sum\_vector, "\n")

cat("Vector 1 - Vector 2: ", diff\_vector, "\n")

cat("Scalar \* Vector 1: ", scaled\_vector, "\n")

cat("Vector 1 . Vector 2: ", dot\_product, "\n")

# Uncomment the relevant code if you want to perform the cross product operation.

**•Write a program to implement the Chi-square test analysis techniques using R programming.**

# Create a contingency table (2x2) with your data

data <- matrix(c(30, 15, 10, 25), nrow = 2)

colnames(data) <- c("Category\_A", "Category\_B")

rownames(data) <- c("Group\_1", "Group\_2")

# Perform the chi-square test

result <- chisq.test(data)

# Print the result of the chi-square test

cat("Chi-Square Test Result:\n")

print(result)

# Extract and print the p-value

p\_value <- result$p.value

cat("\nP-Value: ", p\_value, "\n")

# Check if the result is statistically significant (using a significance level of 0.05)

if (p\_value < 0.05) {

cat("There is a significant association between the categories.\n")

} else {

cat("There is no significant association between the categories.\n")

}

**Write a program for creation of atomic vectors in R and access the elements on the basis of indexing.**

# Creating atomic vectors

numeric\_vector <- c(1, 2, 3, 4, 5)

character\_vector <- c("apple", "banana", "cherry", "date", "fig")

logical\_vector <- c(TRUE, FALSE, TRUE, TRUE, FALSE)

# Accessing elements by index

# Indexing in R starts from 1

# Access elements from the numeric vector

element1 <- numeric\_vector[1]

element3 <- numeric\_vector[3]

# Access elements from the character vector

fruit1 <- character\_vector[1]

fruit4 <- character\_vector[4]

# Access elements from the logical vector

value2 <- logical\_vector[2]

value5 <- logical\_vector[5]

# Printing the elements

cat("Numeric Vector: ", numeric\_vector, "\n")

cat("Character Vector: ", character\_vector, "\n")

cat("Logical Vector: ", logical\_vector, "\n")

cat("Element 1 of Numeric Vector: ", element1, "\n")

cat("Element 3 of Numeric Vector: ", element3, "\n")

cat("Element 1 of Character Vector: ", fruit1, "\n")

cat("Element 4 of Character Vector: ", fruit4, "\n")

cat("Element 2 of Logical Vector: ", value2, "\n")

cat("Element 5 of Logical Vector: ", value5, "\n")

**Write a program to implement the T-test analysis techniques using R programming.**

# Sample data for two groups

group1 <- c(23, 27, 30, 25, 21)

group2 <- c(18, 22, 28, 20, 24)

# Perform an independent two-sample t-test

t\_test\_result <- t.test(group1, group2)

# Print the t-test result

cat("T-Test Result:\n")

print(t\_test\_result)

# Extract and print the p-value

p\_value <- t\_test\_result$p.value

cat("\nP-Value: ", p\_value, "\n")

# Check if the result is statistically significant (using a significance level of 0.05)

if (p\_value < 0.05) {

cat("There is a significant difference between the groups.\n")

} else {

cat("There is no significant difference between the groups.\n")

}

**Write a program to create a list and giving the name to list elements.**

# Create a list with named elements

my\_list <- list(

first\_name = "John",

last\_name = "Doe",

age = 30,

city = "New York"

)

# Print the list

print(my\_list)

# Access elements by name

first\_name <- my\_list$first\_name

last\_name <- my\_list$last\_name

age <- my\_list$age

city <- my\_list$city

# Print the elements

cat("First Name: ", first\_name, "\n")

cat("Last Name: ", last\_name, "\n")

cat("Age: ", age, "\n")

cat("City: ", city, "\n")

**Write a program for creating R objects- Data frame and perform operations on data frame.**

# Create a data frame

data <- data.frame(

Name = c("Alice", "Bob", "Charlie", "David", "Eve"),

Age = c(25, 30, 22, 28, 35),

Score = c(95, 89, 75, 88, 92)

)

# Print the data frame

print(data)

# Access specific columns

names <- data$Name

ages <- data$Age

scores <- data$Score

cat("Names: ", names, "\n")

cat("Ages: ", ages, "\n")

cat("Scores: ", scores, "\n")

# Calculate summary statistics

mean\_age <- mean(data$Age)

mean\_score <- mean(data$Score)

cat("Mean Age: ", mean\_age, "\n")

cat("Mean Score: ", mean\_score, "\n")

# Filter the data frame

young\_students <- data[data$Age < 30, ]

cat("Young Students:\n")

print(young\_students)

**•Write a program to demonstrate Importing and exporting of data in R programming. (Excel file, Text file and CSV file)**

#### **From Excel File (XLSX):**

# Load the "readxl" package for Excel file import (install it if not already installed)

# install.packages("readxl")

library(readxl)

# Import data from an Excel file

excel\_data <- read\_xlsx("your\_excel\_file.xlsx")

# Print the imported data

print(excel\_data)

#### **From Text File (TXT):**

# Import data from a text file

text\_data <- read.table("your\_text\_file.txt", header = TRUE, sep = "\t")

# Print the imported data

print(text\_data)

#### **From CSV File:**

# Import data from a CSV file

csv\_data <- read.csv("your\_csv\_file.csv")

# Print the imported data

print(csv\_data)

### Exporting Data

#### **To Excel File (XLSX):**

# Load the "writexl" package for Excel file export (install it if not already installed)

# install.packages("writexl")

library(writexl)

# Export data to an Excel file

write\_xlsx(your\_data\_frame, "exported\_data.xlsx")

**To Text File (TXT):**

# Export data to a text file (tab-separated values)

Write.table(your\_data\_frame, “exported\_data.txt”, sep = “\t”, row.names = FALSE)

**To CSV File**

# Export data to a CSV file

Write.csv(your\_data\_frame, “exported\_data.csv”, row.names = FALSE)

•**Write a program for creation of lists and perform operation on list in R programming.**

# Create a list

my\_list <- list(

name = "John",

age = 30,

scores = c(85, 92, 78, 88),

is\_student = TRUE

)

# Print the list

print(my\_list)

# Access elements by name

name <- my\_list$name

age <- my\_list$age

scores <- my\_list$scores

is\_student <- my\_list$is\_student

cat("Name: ", name, "\n")

cat("Age: ", age, "\n")

cat("Scores: ", scores, "\n")

cat("Is Student: ", is\_student, "\n")

# Add an element to the list

my\_list$city <- "New York"

# Print the updated list

print(my\_list)

# Remove an element from the list

my\_list$city <- NULL

# Print the list after removing "city"

print(my\_list)

# Check if an element exists in the list

if ("city" %in% names(my\_list)) {

cat("The 'city' element exists in the list.\n")

} else {

cat("The 'city' element does not exist in the list.\n")

}

# Check the length of the list

list\_length <- length(my\_list)

cat("Length of the list: ", list\_length, "\n")

• **Write a program for Exploring Data Manipulations (Summarizing, Sorting, Merging, Joining.)**

# Create two sample data frames

data\_frame1 <- data.frame(

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

Name = c("Alice", "Bob", "Charlie", "David", "Eve"),

Score = c(85, 92, 78, 88, 91)

)

data\_frame2 <- data.frame(

ID = c(4, 5, 6, 7, 8),

Age = c(25, 30, 22, 28, 35),

Grade = c("A", "B", "C", "B", "A")

)

# Summarize data using summary function

summary\_data1 <- summary(data\_frame1)

summary\_data2 <- summary(data\_frame2)

cat("Summary of Data Frame 1:\n")

print(summary\_data1)

cat("\nSummary of Data Frame 2:\n")

print(summary\_data2)

# Sort data frames by a column

sorted\_data1 <- data\_frame1[order(data\_frame1$Score), ]

sorted\_data2 <- data\_frame2[order(data\_frame2$Age), ]

cat("\nData Frame 1 sorted by Score:\n")

print(sorted\_data1)

cat("\nData Frame 2 sorted by Age:\n")

print(sorted\_data2)

# Merge data frames by a common column (ID)

merged\_data <- merge(data\_frame1, data\_frame2, by = "ID")

cat("\nMerged Data Frame:\n")

print(merged\_data)

# Join data frames by a common column (ID) using dplyr package

# Install and load dplyr package if not already installed

# install.packages("dplyr")

library(dplyr)

joined\_data <- left\_join(data\_frame1, data\_frame2, by = "ID")

cat("\nJoined Data Frame:\n")

print(joined\_data)

• **Write a program for creation of lists and perform operation on list in R programming.**

# Create a list

My\_list <- list(

Name = “John”,

Age = 30,

Scores = c(85, 92, 78, 88),

Is\_student = TRUE

)

# Print the list

Print(my\_list)

# Access elements by name

Name <- my\_list$name

Age <- my\_list$age

Scores <- my\_list$scores

Is\_student <- my\_list$is\_student

Cat(“Name: “, name, “\n”)

Cat(“Age: “, age, “\n”)

Cat(“Scores: “, scores, “\n”)

Cat(“Is Student: “, is\_student, “\n”)

# Add an element to the list

My\_list$city <- “New York”

# Print the updated list

Print(my\_list)

# Remove an element from the list

My\_list$city <- NULL

# Print the list after removing “city”

Print(my\_list)

# Check if an element exists in the list

If (“city” %in% names(my\_list)) {

Cat(“The ‘city’ element exists in the list.\n”)

} else {

Cat(“The ‘city’ element does not exist in the list.\n”)

}

# Check the length of the list

List\_length <- length(my\_list)

Cat(“Length of the list: “, list\_length, “\n”)

•**Write a program for creation of Array and perform operations on array in R programming.**

# Create a matrix

Matrix\_data <- matrix(1:12, nrow = 3, ncol = 4)

# Create an array from the matrix

My\_array <- array(matrix\_data, dim = c(3, 4, 2)) # 3x4x2 array

# Print the array

Print(my\_array)

# Access elements of the array

Element1 <- my\_array[1, 2, 1]

Element2 <- my\_array[2, 3, 2]

Cat(“Element (1, 2, 1): “, element1, “\n”)

Cat(“Element (2, 3, 2): “, element2, “\n”)

# Sum of the array

Array\_sum <- sum(my\_array)

Cat(“Sum of the array: “, array\_sum, “\n”)

# Mean of the array

Array\_mean <- mean(my\_array)

Cat(“Mean of the array: “, array\_mean, “\n”)

# Transpose the array

Transposed\_array <- t(my\_array)

Cat(“Transposed Array:\n”)

Print(transposed\_array)

**•Write a program for Validating the data in R programming**

# Sample data to validate

data\_to\_validate <- c(25, 30, 22, 28, 35)

# Define the valid range

min\_value <- 18

max\_value <- 40

# Validate the data

is\_valid <- all(data\_to\_validate >= min\_value & data\_to\_validate <= max\_value)

if (is\_valid) {

cat("Data is valid. All values are within the specified range.\n")

} else {

cat("Data is not valid. Some values are outside the specified range.\n")

}

•**Write a program for creating R objects- Vectors, Array, Matrices, Data frame and Lists.**

# Create a numeric vector

numeric\_vector <- c(1, 2, 3, 4, 5)

print("Numeric Vector:")

print(numeric\_vector)

# Create a character vector

character\_vector <- c("apple", "banana", "cherry", "date", "fig")

print("\nCharacter Vector:")

print(character\_vector)

# Create a logical vector

logical\_vector <- c(TRUE, FALSE, TRUE, TRUE, FALSE)

print("\nLogical Vector:")

print(logical\_vector)

# Create a matrix

matrix\_data <- matrix(1:12, nrow = 3, ncol = 4)

print("\nMatrix:")

print(matrix\_data)

# Create an array from the matrix

array\_data <- array(matrix\_data, dim = c(3, 4, 2))

print("\nArray:")

print(array\_data)

# Create a data frame

data\_frame <- data.frame(

Name = c("Alice", "Bob", "Charlie"),

Age = c(25, 30, 22),

Score = c(85, 92, 78)

)

print("\nData Frame:")

print(data\_frame)

# Create a list

my\_list <- list(

name = "John",

age = 30,

scores = c(85, 92, 78),

is\_student = TRUE

)

print("\nList:")

print(my\_list)

• **Write a program to implement different loops and vectorization of missing values in R programming.**

# Create a vector with missing values

data\_vector <- c(1, NA, 3, 4, NA, 6, NA, 8)

# Using a for loop to replace missing values with the mean of the non-missing values

for (i in 1:length(data\_vector)) {

if (is.na(data\_vector[i])) {

data\_vector[i] <- mean(data\_vector, na.rm = TRUE)

}

}

cat("After using a for loop:\n")

print(data\_vector)

# Reset the vector with missing values

data\_vector <- c(1, NA, 3, 4, NA, 6, NA, 8)

# Using a while loop to replace missing values with the median of the non-missing values

i <- 1

while (i <= length(data\_vector)) {

if (is.na(data\_vector[i])) {

data\_vector[i] <- median(data\_vector, na.rm = TRUE)

}

i <- i + 1

}

cat("\nAfter using a while loop:\n")

print(data\_vector)

# Reset the vector with missing values

data\_vector <- c(1, NA, 3, 4, NA, 6, NA, 8)

# Using vectorization to replace missing values with the mode of the non-missing values

mode\_value <- as.numeric(names(sort(table(data\_vector), decreasing = TRUE)[1]))

data\_vector[is.na(data\_vector)] <- mode\_value

cat("\nAfter using vectorization:\n")

print(data\_vector)

**•Write a program for creation of Matrix and perform operations in R programming.**

# Create a matrix

matrix\_data <- matrix(1:12, nrow = 3, ncol = 4)

# Print the matrix

cat("Original Matrix:\n")

print(matrix\_data)

# Access elements of the matrix

element12 <- matrix\_data[1, 2] # Row 1, Column 2

element32 <- matrix\_data[3, 2] # Row 3, Column 2

cat("\nElement (1, 2): ", element12, "\n")

cat("Element (3, 2): ", element32, "\n")

# Sum of the matrix

matrix\_sum <- sum(matrix\_data)

cat("\nSum of the Matrix: ", matrix\_sum, "\n")

# Mean of the matrix

matrix\_mean <- mean(matrix\_data)

cat("Mean of the Matrix: ", matrix\_mean, "\n")

# Transpose the matrix

transposed\_matrix <- t(matrix\_data)

cat("\nTransposed Matrix:\n")

print(transposed\_matrix)