DRONACHARYA
College of Engineering

R-PROGRAMMING LAB

LABORATORY MANUAL

B.Tech. Semester VIII

CSE/CSIT/IT

Subject Code: PCC-IT-402 G

Session: 2023-24, Even Semester

Name:	
Roll. No.:	
Group/Branch :	

DRONACHARYA COLLEGE OF ENGINEERING, GURUGRAM
DEPARTMENT OF CSE/CSIT
AFFILATED TO GURUGRAM UNIVERSITY, GURUGRAM

Table of Contents

1. Vision and Mission of the Institute
2. Vision and Mission of the Department
3. Programme Educational Objectives (PEOs)
4. Programme Outcomes (POs)
5. Programme Specific Outcomes (PSOs)
6. University Syllabus
7. Course Outcomes (COs)
8. CO- PO and CO-PSO mapping
9. Course Overview
10. List of Experiments
11. DOs and DON'Ts
12. General Safety Precautions
13. Guidelines for students for report preparation
14. Lab assessment criteria
15. Lab Experiments

Vision and Mission of the Institute

Vision:

"Empowering human values and advanced technical education to navigate and address global challenges with excellence."

Mission:

- M1: Seamlessly integrate human values with advanced technical education.
- M2: Supporting the cultivation of a new generation of innovators who are not only skilled but also ethically responsible.
- M3: Inspire global citizens who are equipped to create positive and sustainable impact, driving progress towards a more inclusive and harmonious worl

VISION OF DEPARTMENT

Vision:

"Steering the future of computer science through innovative advancements, fostering ethical values and principles through technical education."

MISSION OF THE DEPARTMENT

Mission:

M1: Directing future innovations in computer science through revolutionary progress.

M2: Instilling a foundation of ethical values and principles in every technologist.

M3: Offering a comprehensive technical education to equip individuals for a meaningful and influential future.

Programme Educational Objectives (PEOs)

PEO1: Apply the technical competence in Computer Science and Engineering for solving problems in the real world.

PEO2: Carry out research and develop solutions on problems of social applications.

PEO3: Work in a corporate environment, demonstrating team skills, work morals, flexibility and lifelong learning.

.

PROGRAM OUTCOMES (POs)

PO1: Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2: Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3: Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4: Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5: Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.

PO6: The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7: Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8: Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9: Individual and teamwork: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10: Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

P11: Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

P12: Life-long learning: Recognize the need for and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1: Exhibit design and programming skills to develop and mechanize business solutions using revolutionary technologies.

PSO2: Learn strong theoretical foundation leading to brilliance and enthusiasm towards research, to provide well-designed solutions to complicated problems.

PSO3: Work effectively with diverse Engineering fields as a team to design, build and develop system applications.

University Syllabus

Course code						
Category	Laboratory course					
Course title	R - Programming Lab					
Scheme and Credits	L	T	P	Credits	Semester =	
	0	0	2	1		
Classwork	50 Marks					
Exam	50 Marks					
Total	100 Marks					
Duration of Exam	02 Hours					

1. Download and install R-Programming environment and install basic packages using install.packages() command in R.
2. Learn all the basics of R-Programming (Data types, Variables, Operators etc.
3. Implement R-Loops with different examples
4. Learn the basics of functions in R and implement with examples.
5. Implement data frames in R. Write a program to join columns and rows in a data frame using cbind() and rbind() in R
6. Implement different String Manipulation functions in R.
7. Implement different data structures in R (Vectors, Lists, Data Frames).
8. Write a program to read a csv file and analyze the data in the file in R.
9. Create pie charts and bar charts using R.
10. Create a data set and do statistical analysis on the data using R

Course Outcomes (COs)

Upon successful completion of the course, the students will be able to:

CO1: Show the installation of R Programming Environment.
CO2: Utilize and R Data types for developing programs.
CO3: Make use of different R Data Structures.
CO4: Develop programming logic using R Packages.
CO5: Analyze the datasets using R programming capabilities
CO6: Apply R programming for reading, cleaning, visualizing and analyzing data

CO-PO Mapping

CO	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3	2	2	-	-	-	-	-	-	-	-	-
CO2	2	2	2	-	-	-	-	-	-	-	-	-

CO3	**1**	**1**	**3**	**-**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**
CO4	**1**	**1**	**2**	**-**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**
C05	**2**	**2**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**
C06	**3**	**2**	**1**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**

CO-PSO Mapping

CO	PSO1	PSO2	PSO3	PSO4
CO1	**3**	**2**	**2**	**-**
CO2	**2**	**3**	**2**	**-**
CO3	**2**	**3**	**2**	**-**
CO4	**3**	**2**	**2**	**-**
CO5	**3**	**2**	**1**	**-**
C06	**3**	**2**	**2**	**-**

*3-HIGH
*2-MEDIUM
*1-LOW

Course Overview

The R-Programming Lab is a practical hands-on course designed to provide students with practical experience in statistics and data analysis applications. It aims to complement the theoretical concepts taught in lectures by giving students the opportunity to apply their knowledge in a real-world data analysis environment.

List of Experiments mapped with COs

Program No	List of Programs	Page No.
1.	Download and install R-Programming environment and install basic packages using install.packages() command in R.	
2.	Learn all the basics of R-Programming (Data types, Variables, Operators etc.)	
3.	Implement R-Loops with different examples.	
4.	Learn the basics of functions in R and implement with examples.	
5.	Implement data frames in R. Write a program to join columns and rows in a data frame using cbind() and rbind() in R	
6.	Implement different String Manipulation functions in R.	
7.	Implement different data structures in R (Vectors, Lists, Data Frames).	
8.	Write a program to read a csv file and analyze the data in the file in R	
9.	Create pie charts and bar charts using R	
10.	Create a data set and do statistical analysis on the data using R	

DOs and DON'Ts

DOs

1. Login-on with your username and password.
2. Log off the Computer every time when you leave the Lab.
3. Arrange your chair properly when you are leaving the lab.
4. Put your bags in the designated area.
5. Ask permission to print.

DON'Ts

1. Do not share your username and password.
2. Do not remove or disconnect cables or hardware parts.
3. Do not personalize the computer setting.
4. Do not run programs that continue to execute after you log off.
5. Do not download or install any programs, games or music on computer in Lab.
6. Personal Internet use chat room for Instant Messaging (IM) and Sites is strictly prohibited.
7. No Internet gaming activities allowed.
8. Tea, Coffee, Water & Eatables are not allowed in the Computer Lab.

General Safety Precautions

Precautions (In case of Injury or Electric Shock)

1. To break the victim with live electric source, use an insulator such as fire wood or plastic to break the contact. Do not touch the victim with bare hands to avoid the risk of electrifying yourself.
2. Unplug the risk of faulty equipment. If main circuit breaker is accessible, turn the circuit off.
3. If the victim is unconscious, start resuscitation immediately, use your hands to press the chest in and out to continue breathing function. Use mouth-to-mouth resuscitation if necessary.
4. Immediately call medical emergency and security. Remember! Time is critical; be best.

Precautions (In case of Fire)

1. Turn the equipment off. If power switch is not immediately accessible, take plug off.
2. If fire continues, try to curb the fire, if possible, by using the fire extinguisher or by covering it with a heavy cloth if possible, isolate the burning equipment from the other surrounding equipment.
3. Sound the fire alarm by activating the nearest alarm switch located in the hallway.
4. Call security and emergency department immediately:

Emergency : 201 (Reception)
Security: 231 (Gate No.1)

Guidelines to students for report preparation

All students are required to maintain a record of the experiments conducted by them. Guidelines for its preparation are as follows: -

1) All files must contain a title page followed by an index page. ***The files will not be signed by the faculty without an entry in the index page.***

2) Student's Name, roll number and date of conduction of experiment must be written on all pages.

3) For each experiment, the record must contain the following

(i) Aim/Objective of the experiment

(ii) Pre-experiment work (as given by the faculty)

(iii) Lab assignment questions and their solutions

(iv) Test Cases (if applicable to the course)

(v) Results/ output

Note:

1. Students must bring their lab record along with them whenever they come for the lab.
2. Students must ensure that their lab record is regularly evaluated.

Lab Assessment Criteria

An estimated 10 lab classes are conducted in a semester for each lab course. These lab classes are assessed continuously. Each lab experiment is evaluated based on 5 assessment criteria as shown in following table. Assessed performance in each experiment is used to compute CO attainment as well as internal marks in the lab course.

Grading Criteria	**Exemplary (4)**	**Competent (3)**	**Needs Improvement (2)**	**Poor (1)**
AC1: Pre-Lab written work (this may be assessed through viva)	Complete procedure with underlined concept is properly written	Underlined concept is written but procedure is incomplete	Not able to write concept and procedure	Underlined concept is not clearly understood
AC2: Program Writing/ Modeling	Unable to understand the reason for errors/ bugs even after they are explicitly pointed out	Assigned problem is properly analyzed, correct solution designed, appropriate language constructs/ tools are applied	Assigned problem is properly analyzed & correct solution designed	Assigned problem is properly analyzed
AC3: Identification & Removal of errors/ bugs	Able to identify errors/ bugs and remove them	Able to identify errors/ bugs and remove them with little bit of guidance	Is dependent totally on someone for identification of errors/ bugs and their removal	Unable to understand the reason for errors/ bugs even after they are explicitly pointed out
AC4: Execution & Demonstration	All variants of input /output are tested, Solution is well demonstrated and implemented concept is clearly explained	All variants of input /output are not tested, However, solution is well demonstrated and implemented concept is clearly explained	Only few variants of input /output are tested, Solution is well demonstrated but implemented concept is not clearly explained	Solution is not well demonstrated and implemented concept is not clearly explained
AC5: Lab Record Assessment	All assigned problems are well recorded with objective, design constructs and solution along with Performance analysis using all variants of input and output	More than 70 % of the assigned problems are well recorded with objective, design contracts and solution along with Performance analysis is done with all variants of input and output	Less than 70 % of the assigned problems are well recorded with objective, design contracts and solution along with Performance analysis is done with all variants of input and output	

LAB EXPERIMENTS

LAB EXPERIMENT 1

PRE EXPERIMENT QUESTIONS

Q 1: What is the purpose of R Programming?

Q 2: What is R, and what is it commonly used for in the context of data analysis and statistics?

OBJECTIVE:

Download and install R-Programming environment and install basic packages using install.packages() command in R.

BRIEF DISCUSSION AND EXPLANATION

Step 1: Download and Install R

1. Go to the CRAN (Comprehensive R Archive Network) website.
2. Choose a CRAN mirror location near you.
3. Download the R installer appropriate for your operating system (Windows, macOS, or Linux).
4. Run the installer and follow the installation instructions.

Step 2: Open R Console (RGui or RStudio)

- If you're using the default RGui on Windows, open it by searching for "R" in the Start menu.
- Alternatively, if you're using RStudio, open RStudio.

Step 3: Install Basic Packages

In the R console or RStudio, you can use the **install.packages()** command to install additional R packages. Let's install a few basic packages as examples:

```
# Install the dplyr package for data manipulation

install.packages("dplyr")


# Install the ggplot2 package for data visualization

install.packages("ggplot2")


# Install the tidyr package for data tidying

install.packages("tidyr")
```

Run these commands one by one, and R will download and install the specified packages from the CRAN repository.

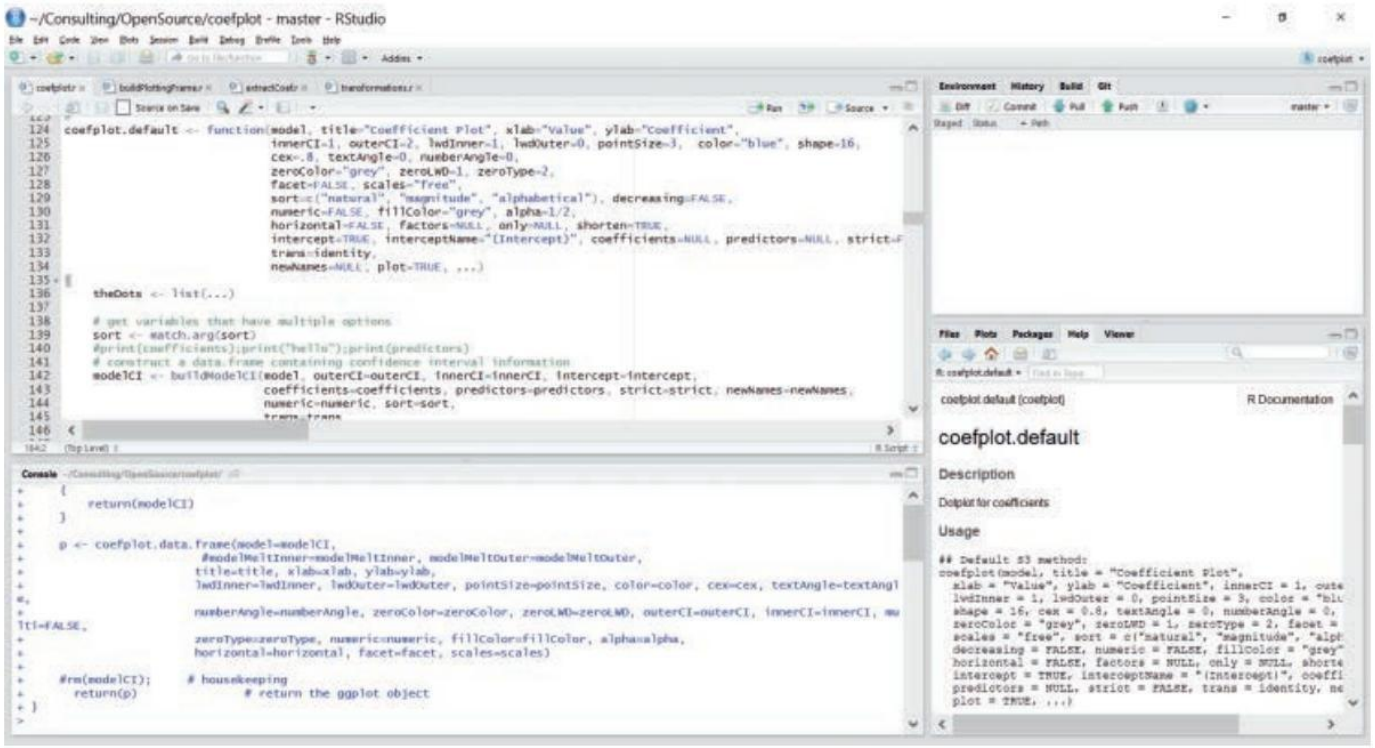
Step 4: Load Installed Packages (if needed)

After installation, you typically need to load the installed packages into your R session using the **library()** function:

```
# Load the dplyr package

library(dplyr)


# Load the ggplot2 package

library(ggplot2)


# Load the tidyr package

library(tidyr)
```

OUTPUT

Now, you have R installed, and you've installed and loaded some basic packages. You can start using these packages for data manipulation, visualization, and analysis.



POST EXPERIMENT QUESTIONS

Q1. Describe the steps to install a new package in R. Which function is commonly used for this purpose?

LAB EXPERIMENT 2

PRE EXPERIMENT QUESTIONS

Q1. What specific objectives do you aim to achieve by learning all the basics of R programming, including data types, variables, and operators?
Q2. What is your current level of familiarity with programming concepts and languages?
Q3. What challenges or difficulties do you anticipate in learning the basics of R programming?

OBJECTIVE

Learn all the basics of R-Programming (Data types, Variables, Operators etc.)

BRIEF DISCUSSION AND EXPLANATION

Here some of the basics of R programming, including data types, variables, operators, and other fundamental concepts.

1. Data Types:

Numeric:

```
num_var <- 42
```

Character:

```
char_var <- "Hello, R!"
```

Logical:

```
logical_var <- TRUE
```

Factor:

```
factor_var <- factor(c("High", "Medium", "Low"))
```

2. Variables:

```
age <- 25

name <- "Alice"

is_student <- TRUE
```

3. Operators:

Arithmetic Operators:

```
sum_result <- 5 + 3
```

```
diff_result <- 7 - 2

prod_result <- 4 * 6

div_result <- 8 / 2
```

Comparison Operators:

```
is_equal <- (5 == 5)

not_equal <- (3 != 7)

greater_than <- (10 > 5)

less_than_equal <- (8 <= 10)
```

Logical Operators:

```
logical_and <- TRUE & FALSE

logical_or <- TRUE | FALSE

logical_not <- !TRUE
```

4. Vectors:

```
numeric_vector <- c(1, 2, 3, 4, 5)

char_vector <- c("apple", "banana", "orange")

logical_vector <- c(TRUE, FALSE, TRUE)
```

5. Matrices:

```
matrix_data <- matrix(c(1, 2, 3, 4, 5, 6), nrow = 2, ncol = 3)
```

6. Lists:

```
my_list <- list(name = "John", age = 30, is_student = FALSE)
```

7. Data Frames:

```
data_frame <- data.frame(name = c("Alice", "Bob", "Charlie"), age = c(25, 30, 22))
```

OUTPUT

Data Type

[1] 42

[1] "Hello, R!"

[1] TRUE

[1] High Medium Low

Levels: High Medium Low

Variable

[1] 25

[1] "Alice"

[1] TRUE

Operators

[1] 8

[1] 5

[1] 24

[1] 4

Vectors

[1] 1 2 3 4 5

[1] "apple" "banana" "orange"

[1] TRUE FALSE TRUE

Matrices

```
     [,1] [,2] [,3]
[1,]    1    3    5
[2,]    2    4    6
```

Lists

$name

[1] "John"

$age

[1] 30

$is_student

[1] FALSE

Data frame

	name	age
1	Alice	25
2	Bob	30
3	Charlie	22

POST EXPERIMENT QUESTIONS

Q1. How well did you grasp the basics of R programming, including data types, variables, and operators?

Q2. Were you able to successfully apply the basics of R programming to accomplish specific tasks or solve problems?

Q3. What challenges did you encounter during the learning process, and how did you overcome them?

LAB EXPERIMENT 3

PRE EXPERIMENT QUESTIONS

Q 1: What specific objectives do you aim to achieve by implementing R-loops with different examples?

Q 2: What is your current understanding or familiarity with programming loops, especially in the context of R?

Q 3: What challenges or difficulties do you anticipate in learning and implementing loops in R?

OBJECTIVE

Implement R-Loops with different examples.

BRIEF DISCUSSION AND EXPLANATION

There are several types of loops in R, and I'll provide examples of a few of them: **for** loop, **while** loop, and **repeat** loop.

Example 1: `for` Loop

```
# Example of a for loop

for (i in 1:5) {

  print(paste("Iteration:", i))

}
```

This loop iterates over the values from 1 to 5, and in each iteration, it prints a message.

Example 2: `while` Loop

```
# Example of a while loop

j <- 1

while (j <= 5) {

  print(paste("Iteration:", j))

  j <- j + 1

}
```

This loop does the same thing as the `for` loop, but it uses a `while` loop instead.

Example 3: `repeat` Loop with `break`

```
# Example of a repeat loop with break

k <- 1

repeat {

 print(paste("Iteration:", k))

 k <- k + 1


 if (k > 5) {

  break  # exit the loop when k is greater than 5

 }

}
```

his example uses a **`repeat`** loop, which continues indefinitely until the **`break`** statement is encountered.

OUTPUT

Output for `for` Loop:

```
[1] "Iteration: 1"

[1] "Iteration: 2"

[1] "Iteration: 3"

[1] "Iteration: 4"

[1] "Iteration: 5"
```

Output for `while` Loop:

```
[1] "Iteration: 1"

[1] "Iteration: 2"

[1] "Iteration: 3"

[1] "Iteration: 4"
```

[1] "Iteration: 5"

Output for `repeat` Loop with `break`:

[1] "Iteration: 1"

[1] "Iteration: 2"

[1] "Iteration: 3"

[1] "Iteration: 4"

[1] "Iteration: 5"

POST EXPERIMENT QUESTIONS

Q 1: How well did you grasp the concepts of loops in R during the learning process?

Q 2: Were you able to successfully implement loops in R to accomplish specific tasks or solve problems?

Q 3: What challenges did you encounter during the implementation, and how did you overcome them?

LAB EXPERIMENT 4

PRE EXPERIMENT QUESTIONS

Q 1: What specific objectives do you aim to achieve by learning the basics of functions in R and implementing them with examples??

Q 2: What challenges or difficulties do you anticipate in learning and implementing functions in R?

OBJECTIVE

Learn the basics of functions in R and implement with examples.

BRIEF DISCUSSION AND EXPLANATION

Defining a Function:

You can define a function in R using the **function** keyword. The basic syntax is as follows:

```
function_name <- function(arg1, arg2, ...) {
  # Function body
  # Perform operations using arguments

  return(result)  # Optional: Return a value
}
```

Example 1: Simple Function without Arguments

```
# Function definition
hello_world <- function() {
print("Hello, World!")
}

# Function call
hello_world()
```

Example 2: Function with Arguments

```
# Function definition

add_numbers <- function(a, b) {

result <- a + b

  return(result)

}


# Function call

sum_result <- add_numbers(3, 7)

print(sum_result)
```

OUTPUT

```
[1] "Hello, World!"

[1] 10
```

POST EXPERIMENT QUESTIONS

Q 1: How well did you grasp the basics of functions in R during the learning process?

Q 2: Were you able to successfully implement functions in R to accomplish specific tasks or solve problems?

LAB EXPERIMENT 5

PRE EXPERIMENT QUESTIONS

Q1. What specific objectives or tasks do you aim to achieve by implementing data frames and performing operations like joining columns and rows using **cbind()** and **rbind()** in R?

Q2. Why have you chosen the **cbind()** and **rbind()** operations for your experiment? How do these operations align with your research objectives?

OBJECTIVE

Implement data frames in R. Write a program to join columns and rows in a data frame using cbind() and rbind() in R

BRIEF DISCUSSION AND EXPLANATION

In R, a data frame is a two-dimensional tabular data structure similar to a table in a database or a spreadsheet. It is a convenient way to store and manipulate data. Here's an example of creating a data frame and using **cbind()** and **rbind()** functions to join columns and rows, respectively:

```
# Create two data frames
df1 <- data.frame(ID = c(1, 2, 3),
                  Name = c("Alice", "Bob", "Charlie"),
                  Age = c(25, 30, 22))

df2 <- data.frame(ID = c(4, 5, 6),
                  Name = c("David", "Eve", "Frank"),
                  Age = c(28, 35, 29))

# Display the original data frames
cat("Original Data Frame 1:\n")
print(df1)

cat("\nOriginal Data Frame 2:\n")
print(df2)

# Using cbind() to join columns
cbind_result <- cbind(df1, Salary = c(50000, 60000, 45000))
```

```
cat("\nJoined Data Frame using cbind():\n")
print(cbind_result)

# Using rbind() to join rows
rbind_result <- rbind(df1, df2)

cat("\nJoined Data Frame using rbind():\n")
print(rbind_result)
```

This code creates two data frames (`df1` and `df2`) with three columns (ID, Name, Age) each. It then uses `cbind()` to join a new column, "Salary," to `df1`. Finally, it uses `rbind()` to concatenate `df1` and `df2` along the rows.

OUTPUT

```
Original Data Frame 1:
  ID    Name Age
1  1  Alice  25
2  2     Bob 30
3  3 Charlie  22

Original Data Frame 2:
  ID  Name Age
1  4 David  28
2  5  Eve  35
3  6 Frank  29

Joined Data Frame using cbind():
  ID    Name Age Salary
1  1   Alice  25  50000
2  2     Bob  30  60000
3  3 Charlie  22  45000

Joined Data Frame using rbind():
  ID    Name Age
1  1  Alice  25
```

2 2 Bob 30
3 3 Charlie 22
4 4 David 28
5 5 Eve 35
6 6 Frank 29

POST EXPERIMENT QUESTIONS

Q 1: How did the implementation of **cbind()** and **rbind()** operations perform in terms of efficiency and speed? Were there notable differences between the two operations??

Q 2: To what extent did **cbind()** and **rbind()** meet the intended functionality? Were there any limitations or unexpected benefits in terms of flexibility?

LAB EXPERIMENT 6

PRE EXPERIMENT QUESTIONS

Q 1: What specific objectives or tasks do you intend to achieve by implementing different string manipulation functions in R??

Q 2: Why have you chosen particular string manipulation functions for your experiment? How do these functions align with your research objectives??

OBJECTIVE

Implement different String Manipulation functions in R.

BRIEF DISCUSSION AND EXPLANATION

In R, there are several string manipulation functions available in the base and additional packages. Here are examples of some commonly used string manipulation functions:

1. Concatenation:

```
# Concatenate strings

string1 <- "Hello"

string2 <- "World"


result <- paste(string1, string2)

cat("Concatenated String:", result, "\n")
```

2. Substring Extraction:

```
# Extract substring

original_string <- "DataScience"

substring <- substr(original_string, start = 5, stop = 9)
```

```
cat("Substring:", substring, "\n")
```

3. String Length:

```
# Calculate string length

string <- "Programming"

length_result <- nchar(string)


cat("String Length:", length_result, "\n")
```

4. Uppercase and Lowercase:

```
# Convert to uppercase and lowercase

uppercase_string <- toupper(string)

lowercase_string <- tolower(string)


cat("Uppercase String:", uppercase_string, "\n")

cat("Lowercase String:", lowercase_string, "\n")
```

5. String Replacement:

```
# Replace a substring

original_string <- "I love programming in R"

modified_string <- gsub("R", "Python", original_string)


cat("Modified String:", modified_string, "\n")
```

6. Splitting Strings:

```
# Split a string

text <- "apple,orange,banana"

split_result <- strsplit(text, ",")
```

```
cat("Split Result:", unlist(split_result), "\n")
```

OUTPUT

Concatenated String: Hello World

Substring: Scien

String Length: 11

Uppercase String: PROGRAMMING

Lowercase String: programming

Modified String: I love programming in Python

Split Result: apple orange banana

POST EXPERIMENT QUESTIONS

Q 1: How did the implemented string manipulation functions perform in terms of efficiency and speed? Were there notable differences among the functions?

Q 2: How readable and usable was the code using these string manipulation functions? Did any of them enhance or hinder code clarity?

LAB EXPERIMENT 7

PRE EXPERIMENT QUESTIONS

Q1. What specific objectives or goals do you intend to achieve by implementing different data structures (Vectors, Lists, Data Frames) in R?

Q2. Why have you chosen vectors, lists, and data frames for your experiment? How do these data structures align with your research objectives??

Q3. What challenges or difficulties do you anticipate in implementing and working with these data structures?

OBJECTIVE

Implement different data structures in R (Vectors, Lists, Data Frames).

BRIEF DISCUSSION AND EXPLANATION

In R, you can work with different data structures such as vectors, lists, and data frames. Here's a brief overview and examples of each:

1. Vectors:

Vectors are one-dimensional arrays that can hold elements of the same data type.

 # Creating a numeric vector

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

 # Creating a character vector

character_vector <- c("apple", "orange", "banana")

 # Accessing elements of a vector

print(numeric_vector[3]) # Output: 3

2. Lists:

Lists are collections of elements that can be of different data types. Each element in a list can be a vector, list, or any other R object.

```
# Creating a list
my_list <- list(numeric_vector, character_vector, TRUE, "Hello")


# Accessing elements of a list
print(my_list[[1]]) # Output: Numeric vector
print(my_list[[2]])  # Output: Character vector
```

3. Data Frames:

Data frames are two-dimensional structures, similar to a table or spreadsheet, with rows and columns.

```
# Creating a data frame
my_data <- data.frame(
  Name = c("Alice", "Bob", "Charlie"),
  Age = c(25, 30, 22),
  Grade = c("A", "B", "C")
)


# Accessing elements of a data frame
print(my_data$Name)  # Output: Names column
print(my_data[2, 3])  # Output: Element in the second row and third column
```

Output:

```
[1] 3
```

[1] 1 2 3 4 5

[1] "apple" "orange" "banana"

[1] Alice Bob Charlie

Levels: Alice Bob Charlie

[1] B

Levels: A B C

POST EXPERIMENT QUESTIONS

Q1. How did the implemented data structures perform in terms of efficiency and speed? Were there notable differences among vectors, lists, and data frames?

Q2. How readable and maintainable was the code using these data structures? Did any of them enhance or hinder code clarity?

LAB EXPERIMENT 8

PRE EXPERIMENT QUESTIONS

Q1. What specific objectives do you aim to achieve by reading and analyzing the CSV file in R?

Q2. Where does the CSV file come from, and what kind of data does it contain??

Q3. What steps will you take for data cleaning and preprocessing before performing the analysis??

OBJECTIVE

Write a program to read a csv file and analyze the data in the file in R.

BRIEF DISCUSSION AND EXPLANATION

In R, you can use the **read.csv()** function to read a CSV file and then perform various analyses on the data. Here's a simple example:

Assuming you have a CSV file named "data.csv" with the following content:

Name, Age, Grade

Alice, 25, A

Bob, 30, B

Charlie, 22, C

Now, let's write a program to read and analyze this CSV file:

 # Read CSV file

data <- read.csv("data.csv", header = TRUE)

```
# Display the data
print("Data in the CSV file:")
print(data)

# Summary statistics
summary_stats <- summary(data$Age)
print(paste("Summary statistics for Age:", summary_stats))

# Mean of Age
mean_age <- mean(data$Age)
print(paste("Mean Age:", mean_age))

# Maximum Age
max_age <- max(data$Age)
print(paste("Maximum Age:", max_age))

# Minimum Age
min_age <- min(data$Age)
print(paste("Minimum Age:", min_age))
```

This program reads the CSV file, displays the data, calculates summary statistics for the "Age" column, calculates the mean, maximum, and minimum values of the "Age" column. Adjust the code according to your specific requirements and the structure of your CSV file.

OUTPUT

```
[1] "Data in the CSV file:"
```

```
    Name Age Grade
1   Alice  25     A
2     Bob  30     B
3 Charlie  22     C
```

```
[1] "Summary statistics for Age:"
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
   22.0    23.5    25.0    25.7    27.5    30.0
```

```
[1] "Mean Age: 25.6666666666667"
[1] "Maximum Age: 30"
[1] "Minimum Age: 22"
```

This output includes the displayed data, summary statistics for the "Age" column, mean age, maximum age, and minimum age based on the content of the provided CSV file.

POST EXPERIMENT QUESTIONS

Q1. What insights or patterns did you discover from the analysis of the CSV file?

Q2. How would you describe the quality of the data in the CSV file? Were there any issues encountered during data analysis that might indicate data quality concerns??

LAB EXPERIMENT 9

PRE EXPERIMENT QUESTIONS

Q1. What is the primary objective of using pie charts and bar charts in your research or experiment??

Q2. What are the key variables you plan to represent using pie charts and bar charts?

Q3. Why have you chosen pie charts and bar charts specifically for your visualizations? What insights do you expect to gain from each??

OBJECTIVE

Create pie charts and bar charts using R.

BRIEF DISCUSSION AND EXPLANATION

In R, you can use the **pie()** and **barplot()** functions to create pie charts and bar charts, respectively. Here's an example for both:

1. Pie Chart:

```
# Data for the pie chart
pie_data <- c(30, 40, 20, 10)
labels <- c("Category A", "Category B", "Category C", "Category D")

# Create a pie chart
pie(pie_data, labels = labels, main = "Pie Chart Example")
```

2. Bar Chart:

 # Data for the bar chart

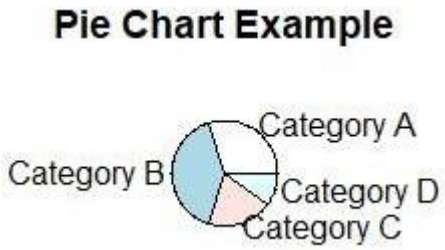
bar_data <- c(25, 40, 15, 30)

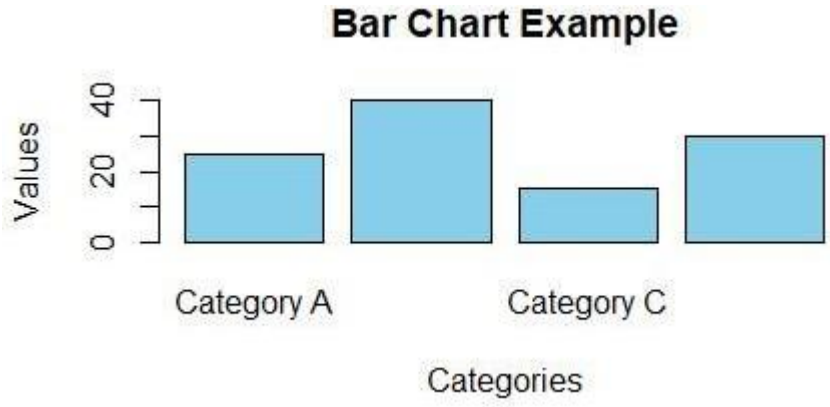
categories <- c("Category A", "Category B", "Category C", "Category D")

 # Create a bar chart

barplot(bar_data, names.arg = categories, col = "skyblue", main = "Bar Chart Example", xlab = "Categories", ylab = "Values")

OUTPUT





POST EXPERIMENT QUESTIONS

Q1. How effective were the pie charts and bar charts in communicating the information you intended?

Q2. What insights did you gain from the visualizations that were not immediately apparent in the raw data.?
Q3. Were the visualizations accurate and precise in representing the underlying data?

LAB EXPERIMENT 10

PRE EXPERIMENT QUESTIONS

Q1. What is the main objective of your research or experiment?

Q2. What are the key variables of interest in your analysis?

OBJECTIVE

Create a data set and do statistical analysis on the data using R.

BRIEF DISCUSSION AND EXPLANATION

Let's create a simple dataset and perform some basic statistical analysis using R. In this example, I'll create a dataset with two variables, 'Height' and 'Weight', and then calculate descriptive statistics and conduct a t-test.

```
# Create a dataset

set.seed(123) # Setting seed for reproducibility

height <- rnorm(50, mean = 170, sd = 10)

weight <- rnorm(50, mean = 70, sd = 5)


# Combine variables into a data frame

my_data <- data.frame(Height = height, Weight = weight)


# Display the first few rows of the dataset

print("First few rows of the dataset:")
```

```
print(head(my_data))

# Descriptive statistics
print("Descriptive statistics:")
print(summary(my_data))

# T-test for comparing means of 'Height' between two groups
group1 <- my_data$Height[1:25]
group2 <- my_data$Height[26:50]

t_test_result <- t.test(group1, group2)
print("T-test for comparing means of 'Height' between two groups:")
print(t_test_result)
```

OUTPUT

[1] "First few rows of the dataset:"

```
    Height   Weight
1  173.4392 66.00146
2  160.7693 70.73500
3  174.3770 74.89562
4  163.6827 73.63305
5  175.4874 71.47079
6  167.8282 70.05956
```

[1] "Descriptive statistics:"

```
    Height         Weight
```

Min. :144.8 Min. :57.22

1st Qu.:164.4 1st Qu.:66.22

Median :170.1 Median :70.32

Mean :169.8 Mean :70.52

3rd Qu.:176.1 3rd Qu.:74.20

Max. :192.5 Max. :79.49

[1] "T-test for comparing means of 'Height' between two groups:"

Welch Two Sample t-test

data: group1 and group2

t = -0.89027, df = 47.943, p-value = 0.3785

alternative hypothesis: true difference in means is not equal to 0

95 percent confidence interval:

-3.658126 1.451653

sample estimates:

mean of x mean of y

169.8056 170.7962

POST EXPERIMENT QUESTIONS

Q1 What insights did you gain from examining the first few rows of the dataset?

Q2 What does the t-test result suggest about the difference in means between the two groups?