**PART-A**

**Program 1 (.csv mean, median)**

# Importing the example data

examples <- read.csv("D:/example.csv")

# Taking the mean

mean\_value <- mean(examples[, 1])

cat("Mean =", mean\_value)

# Taking the median

median\_value <- median(examples[, 1])

cat("Median =", median\_value)

# Taking the variance

variance\_value <- var(examples[, 1])

cat("Variance =", variance\_value)

# Taking the standard deviation

sd\_value <- variance\_value^0.5

cat("Standard Deviation =", sd\_value)

# Taking the range

range\_value <- range(examples[, 1])

cat("Range =", range\_value)

# Taking the quartile

quartile\_1 <- quantile(examples[, 1], 0.25)

cat("1st Quartile =",quartile\_1)

quartile\_2 <- quantile(examples[, 1], 0.5)

cat("2nd Quartile =",quartile\_2)

quartile\_3 <- quantile(examples[, 1], 0.75)

cat("3rd Quartile =",quartile\_3)

**Program 2 (file operations)**

# Create a file

file.create("D:/AEC\_File1.txt")

# Writing into a file

write.table(x = iris[1:10, ], file = "AEC\_File1.txt")

# Reading a text file

myData = read.table(file = "AEC\_File1.txt ")

print(myData)

# Renaming a file

file.rename("AEC\_File1.txt", "AEC\_renamed.txt")

# Listing the table

list.files()

# Copy a file

file.copy("D:/AEC\_File1.txt", "D:/programs")

list.files("D:/programs")

**Program 3 (string operations, reverse, palindrome)**

# Create two strings

str1 = "Artificial"

str2 = "Intelligence"

# using paste() to concatenate two strings

result = paste(str1, str2)

print(result)

# Compare both strings

result = toupper(str1) == toupper(str2)

print(result)

# Reverse the string

text = "Maths is fun"

reversed\_text = rev(strsplit(text, "")[[1]])

reversed\_text = paste(reversed\_text, collapse = "")

cat("Reversed String:", reversed\_text, "\n")

# Check if a given string is a palindrome

text2 = "mom"

# Reverse the string

reversed\_string = rev(strsplit(text2, "")[[1]])

reversed\_string = paste(reversed\_string, collapse = "")

# Check if the input string is equal to its reverse

if (text2 == reversed\_string) {

cat("The string is a palindrome.\n")

} else {

cat("The string is not a palindrome.\n")

}

**Program 4 (string manipulation)**

string = "Hello World"

# nchar function

nchar(string)

# toupper function

toupper(string)

# tolower function

tolower(string)

# substr function

substr(string, 5, 20)

# grep function

grep("wor", string)

# paste function

paste("hello", "world", string, sep = "-")

# strsplit function

strsplit(string,'e')

# sprintf function

age = 40L

name = "Ram"

sprintf("%s is %d years old", name, age)

# cat function

cat("hello","world",sep = "-")

# sub function

sub("World", "there!", string)

**PART-B**

**Program 5 (scatter, line, bar, histo)**

# Sample data

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

y = c(2, 4, 6, 8, 10)

# Scatter Plot

plot(x, y, type = "p", col = "red", pch = 16, xlab = "X-axis", ylab = "Y-axis", main = "Scatter

Plot Example")

# Line Plot

plot(x, y, type = "l", col = "blue", lwd = 2, xlab = "X-axis", ylab = "Y-axis", main = "Line Plot

Example")

# Bar Plot

barplot(y, names.arg = x, col = "purple", xlab = "X-axis", ylab = "Y-axis", main = "Bar Plot

Example")

# Histogram

hist(y, col = "red", xlab = "X-axis", ylab = "Frequency", main = "Histogram Example")

**Program 6 (2D, 3D pie chart)**

marks<-c(100,65,89,90,85)

subjects<-c("Math","DAA","R","OS","ARM")

percentage<-round(100 \* marks/sum(marks),1)

pie(marks, labels=marks, main="Marks scored by a student", col=rainbow(length(marks)))

legend("topleft",subjects,cex=1,fill =rainbow(length(marks)))

library(plotrix)

pie3D(marks, labels=marks, main="Marks scored by a person", col=rainbow(length(marks)),explode=0.2)

legend("bottom",subjects,cex=1,fill =rainbow(length(marks)))

**Program 7 (iris dataset)**

# install.packages("ggplot2")

library(ggplot2)

# Load the iris dataset (it's built-in)

data(iris)

# Bar Plot

bar\_plot = ggplot(iris, aes(x = Species)) + geom\_bar(fill = "skyblue") + labs(title = "Bar Plot

of Iris Species", x = "Species", y = "Count")

print(bar\_plot)

# Line Plot

line\_plot = ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species)) +

geom\_line() + labs(title = "Line Plot of Sepal Length vs Sepal Width", x = "Sepal Length", y

= "Sepal Width")

print(line\_plot)

# Scatter Plot

scatter\_plot = ggplot(iris, aes(x = Petal.Length, y = Petal.Width, color = Species, shape =

Species)) + geom\_point(size = 3, alpha = 0.7) + labs(title = "Scatter Plot of Petal Length vs

Petal Width", x = "Petal Length", y = "Petal Width") + scale\_color\_manual(values =

c("setosa" = "blue", "versicolor" = "green", "virginica" = "red")) +

scale\_shape\_manual(values = c("setosa" = 16, "versicolor" = 17, "virginica" = 18))

print(scatter\_plot)

**Program 8 (histo and box using ggplot2)**

# Load required libraries

library(ggplot2)

# Histogram

# Sample data for Histogram

data = data.frame(values = rnorm(1000))

# Create a histogram

ggplot(data, aes(x = values)) +

geom\_histogram(binwidth = 0.5, fill = "lightblue", color = "black") +

labs(title = "Histogram", x = "Values", y = "Frequency") + theme\_minimal()

library(ggplot2)

#Box plot

# Sample data for box plot

data = data.frame(group = rep(c("A", "B", "C"), each = 50), value = rnorm(150))

# Create a box plot

ggplot(data, aes(x = group, y = value, fill = group)) +

geom\_boxplot() + labs(title = "Box Plot", x = "Group", y = "Value") +

theme\_minimal()

**Program 9 (mtcars and lattice)**

# Load required libraries

library(lattice)

# Create a bar plot of average MPG by number of cylinders

avg\_mpg\_by\_cyl = tapply(mtcars$mpg, mtcars$cyl, mean)

bar\_plot = barchart(avg\_mpg\_by\_cyl, main = "Average MPG by Number of Cylinders",

xlab = "Cylinders", ylab = "Average MPG", col = "orange")

print(bar\_plot)

# Create a scatter plot of MPG vs Horsepower

scatter\_plot = xyplot(mpg ~ hp, data = mtcars, pch = 16, col = "blue", main = "Scatter Plot

of MPG vs. Horsepower", xlab = "Horsepower", ylab = "MPG")

print(scatter\_plot)

# Create a histogram of MPG values

histogram\_plot = histogram (~ mpg, data = mtcars, main = "Histogram of MPG", xlab =

"MPG", ylab = "Frequency", col = "green")

print(histogram\_plot)

# Create a density plot of MPG values

density\_plot <- densityplot(~ mpg, data = mtcars, main = "Density Plot of MPG", xlab =

"MPG", ylab = "Density", col = "purple")

print(density\_plot)

**Program 10 (3D wireframe)**

# Load the lattice package for advanced visualizations

library(lattice)

# Create numeric vectors 'a' and 'b'

a = 1:10

b = 1:15

# Generate a data frame with all combinations of 'a' and 'b'

eg =- expand.grid(x=a, y=b)

# Calculate a new variable 'z' based on the formula

eg$z = eg$x^2 + eg$x \* eg$y

# Create a 3D wireframe plot to visualize 'z' vs 'x' and 'y'

wireframe(z ~ x+y, eg)

#Level plot

x = seq(-pi, pi, length.out = 100)

y = seq(-pi, pi, length.out = 100)

z = outer(x, y, function(x, y) sin(sqrt(x^2 + y^2)))

levelplot(z, xlab = "x", ylab = "y", main = "2D Sin Function")