**1.Calculate the number of occurrences of a character in each row of R DataFrame.**

# loading the reqd library

library ("stringr")

# creating a data frame

data\_frame <- data.frame(

col1 = c(1:5), col2 = c("Geeks","for","geeks","CSE","portal"))

# character to search for

ch <- "e"

# counting the occurrences of character

count <- str\_count(data\_frame$col2, ch)

print ("Count of e :")

print (count)

[1] "Count of e :"> print (count)

[1] 2 0 2 0 0

2.Find row and column index of maximum and minimum value in a matrix in R.

# Example matrix

mat <- matrix(c(1, 2, NA, 4, 5, 6, NA, 8, 9), nrow=3, byrow=TRUE)

# Find the maximum value and its indices

max\_value <- max(mat, na.rm = TRUE)

max\_indices <- which(mat == max\_value, arr.ind = TRUE)

# Find the minimum value and its indices

min\_value <- min(mat, na.rm = TRUE)

min\_indices <- which(mat == min\_value, arr.ind = TRUE)

# Print the results

print(paste("Maximum value:", max\_value))

print("Indices of maximum value:")

print(max\_indices)

print(paste("Minimum value:", min\_value))

print("Indices of minimum value:")

print(min\_indices)

OUTPUT:

> # Print the results> print(paste("Maximum value:", max\_value))[1] "Maximum value: 9"

> print("Indices of maximum value:")[1] "Indices of maximum value:"

> print(max\_indices) row col

[1,] 3 3> > print(paste("Minimum value:", min\_value))[1] "Minimum value: 1"

> print("Indices of minimum value:")[1] "Indices of minimum value:"

> print(min\_indices) row col

[1,] 1 1

1. **Multiply a matrix by its transpose while ignoring missing values in R.**

**# Define a custom function to multiply a matrix by its transpose, ignoring NAs**

**multiply\_by\_transpose\_ignore\_na <- function(mat) {**

**n <- nrow(mat)**

**m <- ncol(mat)**

**result <- matrix(0, n, n)**

**for (i in 1:n) {**

**for (j in i:n) {**

**valid\_indices <- !is.na(mat[i,]) & !is.na(mat[j,])**

**if (sum(valid\_indices) > 0) {**

**result[i, j] <- sum(mat[i, valid\_indices] \* mat[j, valid\_indices])**

**result[j, i] <- result[i, j] # Symmetric matrix**

**}**

**}**

**}**

**return(result)**

**}**

**# Example matrix with NA values**

**mat <- matrix(c(1, 2, NA, 4, 5, 6, NA, 8, 9), nrow=3, byrow=TRUE)**

**# Apply the custom function**

**result <- multiply\_by\_transpose\_ignore\_na(mat)**

**# Print the result**

**print(result)**

**Output:**

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

[1,] 5 14 16

[2,] 14 77 94

[3,] 16 94 145

1. **Remove rows with NA in one column of R DataFrame.**

**# Example DataFrame**

**df <- data.frame(**

**A = c(1, 2, NA, 4, 5),**

**B = c(6, 7, 8, NA, 10),**

**C = c(11, NA, 13, 14, 15)**

**)**

**print("Original DataFrame:")**

**print(df)**

**# Using na.omit**

**df\_clean\_na\_omit <- df[!is.na(df$A), ]**

**print("DataFrame after removing rows with NA in column A using na.omit:")**

**print(df\_clean\_na\_omit)**

**# Using subset**

**df\_clean\_subset <- subset(df, !is.na(A))**

**print("DataFrame after removing rows with NA in column A using subset:")**

**print(df\_clean\_subset)**

**Output**

# Example DataFrame

> df <- data.frame(+ A = c(1, 2, NA, 4, 5),+ B = c(6, 7, 8, NA, 10),+ C = c(11, NA, 13, 14, 15)+ )

> > print("Original DataFrame:")[1] "Original DataFrame:"> print(df) A B C

1 1 6 11

2 2 7 NA

3 NA 8 13

4 4 NA 14

5 5 10 15>

> # Using na.omit

> df\_clean\_na\_omit <- df[!is.na(df$A), ]>

> print("DataFrame after removing rows with NA in column A using na.omit:")[1] "DataFrame after removing rows with NA in column A using na.omit:"

> print(df\_clean\_na\_omit) A B C

1 1 6 11

2 2 7 NA

4 4 NA 14

5 5 10 15>

> # Using subset> df\_clean\_subset <- subset(df, !is.na(A))>

> print("DataFrame after removing rows with NA in column A using subset:")[1] "DataFrame after removing rows with NA in column A using subset:"

> print(df\_clean\_subset) A B C

1 1 6 11

2 2 7 NA

4 4 NA 14

5 5 10 15>

1. **How to find common rows and columns between two dataframe in R?**

**# Example DataFrames**

**df1 <- data.frame(**

**A = c(1, 2, 3),**

**B = c(4, 5, 6),**

**C = c(7, 8, 9)**

**)**

**df2 <- data.frame(**

**A = c(1, 2, 10),**

**B = c(4, 5, 11),**

**D = c(7, 8, 12)**

**)**

**print("DataFrame 1:")**

**print(df1)**

**print("DataFrame 2:")**

**print(df2)**

**# Find common columns**

**common\_columns <- intersect(names(df1), names(df2))**

**print("Common Columns:")**

**print(common\_columns)**

**# Find common rows**

**common\_rows <- merge(df1, df2)**

**print("Common Rows:")**

**print(common\_rows)**

**# Find common rows based on specific columns**

**common\_rows\_specific <- merge(df1, df2, by = c("A", "B"))**

**print("Common Rows Based on Specific Columns (A and B):")**

**print(common\_rows\_specific)**

**Output:**

print("DataFrame 1:")[1] "DataFrame 1:"> print(df1) A B C

1 1 4 7

2 2 5 8

3 3 6 9> > print("DataFrame 2:")[1] "DataFrame 2:"> print(df2) A B D

1 1 4 7

2 2 5 8

3 10 11 12> > # Find common columns> common\_columns <- intersect(names(df1), names(df2))> print("Common Columns:")[1] "Common Columns:"> print(common\_columns)[1] "A" "B"> > # Find common rows> common\_rows <- merge(df1, df2)> print("Common Rows:")[1] "Common Rows:"> print(common\_rows) A B C D

1 1 4 7 7

2 2 5 8 8> > # Find common rows based on specific columns> common\_rows\_specific <- merge(df1, df2, by = c("A", "B"))> print("Common Rows Based on Specific Columns (A and B):")[1] "Common Rows Based on Specific Columns (A and B):"> print(common\_rows\_specific) A B C D

1 1 4 7 7

2 2 5 8 8>

1. **Adding Colors to Charts in R Programming.**

# Creating a list

temp<-c(5, 10, 15, 20, 25)

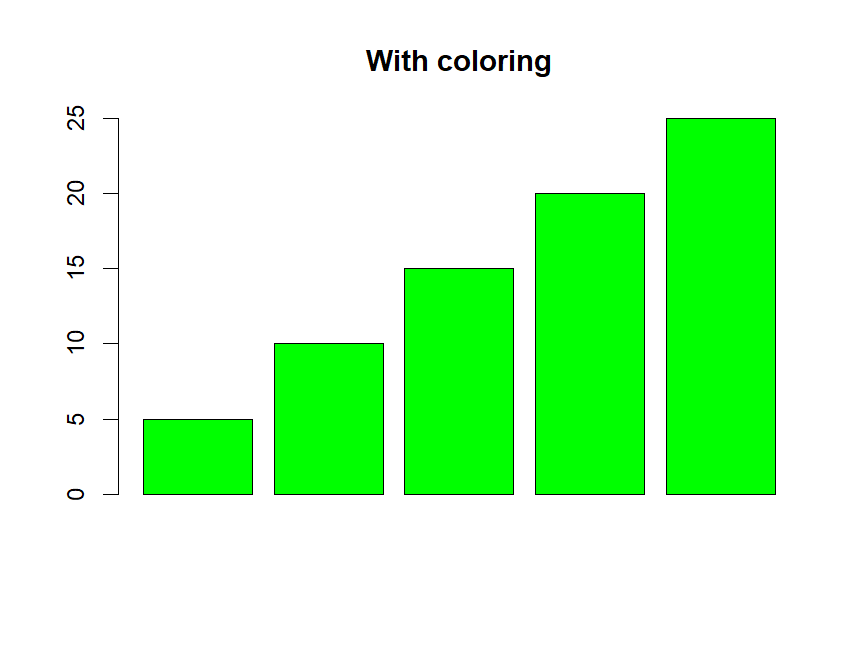
# Barplot with default color

barplot(temp, main ="By default")

# Barplot with green color

barplot(temp, col ="green",

main ="With coloring")



1. Create a Stacked Dot Plot in R.

# Load ggplot2

library(ggplot2)

# Example data

data <- data.frame(

category = rep(c("A", "B", "C"), each = 10),

value = c(sample(1:10, 10), sample(1:10, 10), sample(1:10, 10))

)

# Create the dot plot

ggplot(data, aes(x = category, y = value)) +

geom\_dotplot(binaxis = 'y', stackdir = 'center', dotsize = 0.5) +

labs(title = "Stacked Dot Plot", x = "Category", y = "Value") +

theme\_minimal()

**8.Plot Shaded Area between vertical lines in R.**

x<-rpois(10,5)

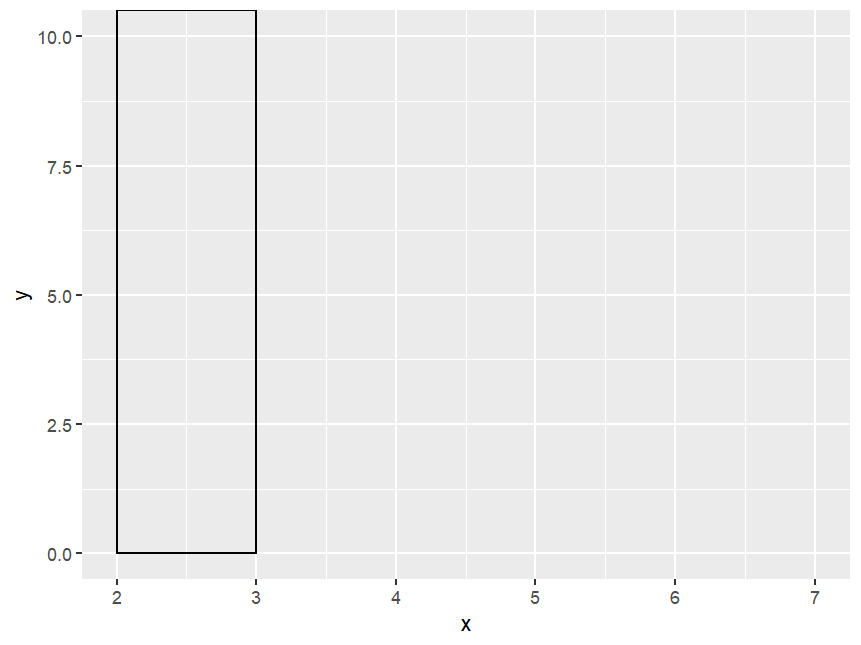
y<-rpois(10,7)

df<-data.frame(x,y)

library(ggplot2)

ggplot(df,aes(x,y))+

geom\_rect(aes(xmin=2,xmax=3,ymin=0,ymax=Inf),color="black",alpha=0)

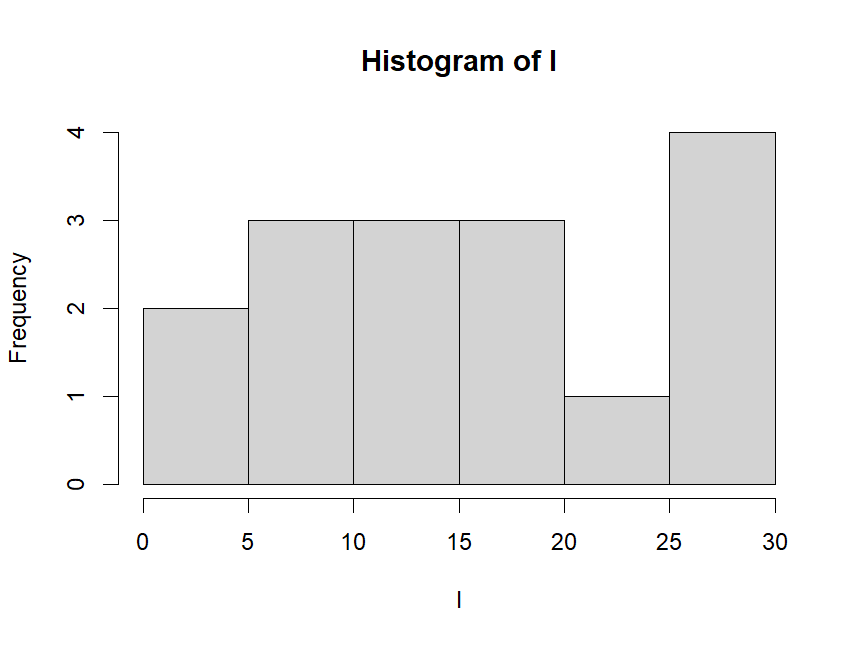


1. **Add Mean and Median to Histogram in.**

**l <- c(2, 3, 7, 8, 10, 13, 14, 15,**

**18, 18, 20, 26, 25, 26, 27, 28)**

**hist(l)**

****

**OR**

**# Sample data**

**set.seed(123)**

**data <- rnorm(1000)**

**# Calculate mean and median**

**mean\_value <- mean(data)**

**median\_value <- median(data)**

**# Create histogram**

**hist(data, main = "Histogram with Mean and Median", xlab = "Value", col = "lightblue", border = "black")**

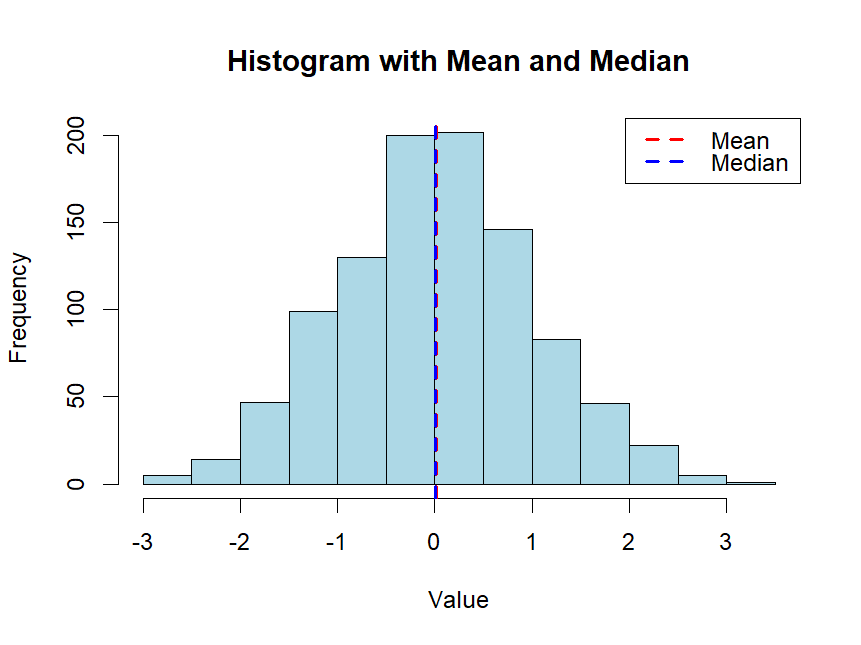
**# Add mean and median lines**

**abline(v = mean\_value, col = "red", lwd = 2, lty = 2) # Mean line**

**abline(v = median\_value, col = "blue", lwd = 2, lty = 2) # Median line**

**# Add legend**

**legend("topright", legend = c("Mean", "Median"), col = c("red", "blue"), lwd = 2, lty = 2)**

****

**10.**

**# Load necessary package**

**library(ggplot2)**

**# Load the CSV file**

**data <- read.csv("data.csv")**

**# Inspect the first few rows of the data**

**print(head(data))**

**# Scatter plot using base R**

**plot(data$x, data$y,**

**main = "Scatter Plot (Base R)",**

**xlab = "X-axis Label",**

**ylab = "Y-axis Label",**

**pch = 19, # Plotting character, 19 is for solid circles**

**col = "blue") # Color of the points**

**# Scatter plot using ggplot2**

**ggplot(data, aes(x = x, y = y)) +**

**geom\_point(color = "blue") +**

**labs(title = "Scatter Plot (ggplot2)", x = "X-axis Label", y = "Y-axis Label") +**

**theme\_minimal()**