-----------------------------------------------------------

# 1. Data Types

-----------------------------------------------------------

#Logical

v <- TRUE

print (v)

print(class(v))

#Numeric

v <- 23.5

print(class(v))

# Integer

v <- 2L #The L stands for "literal integer" Without the L, R would treat 2 as a numeric, which is actually a double (floating-point number).

print(class(v))

v <- 2 #The L stands for "literal integer" Without the L, R would treat 2 as a numeric, which is actually a double (floating-point number).

print(class(v))

# complex

v <- 2+5i

print(class(v))

# character

v <- "TRUE"

print(class(v))

v <- '23.4'

print(class(v))

# raw

v <- charToRaw("Hello")

v

print(class(v))

---------------------------------------------------------------

# 2. Vectors

---------------------------------------------------------------

# Numeric vector

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

print(num\_vector)

my\_vector <- c(5, 10, 15, 20, 25)

print(my\_vector)

# Basic math with vectors

# Add numbers to the vector

my\_vector\_plus\_ten <- my\_vector + 10

print(my\_vector\_plus\_ten)

# Calculate the mean of the vector

mean\_value <- mean(my\_vector)

print(mean\_value)

# Character vector

char\_vector <- c("apple", "banana", "cherry")

print(char\_vector)

# Logical vector

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

print(log\_vector)

# Accessing the 2nd element in the numeric vector

print(num\_vector[2])

# Accessing the 1st element in the numeric vector

print(num\_vector[1])

# Accessing the size of elements in the character vector

length(char\_vector)

# Accessing the size of elements in the character vector

length(num\_vector)

# Accessing multiple elements (e.g., 1st and 3rd elements)

print(num\_vector[c(1, 3)])

----------------------------------------------------------------

# Creating a list with different types of elements

my\_list <- list(name = "Alice", age = 25, scores = c(90, 85, 88))

# Accessing elements of the list

print(my\_list$name)

print(my\_list$age)

print(my\_list$scores)

----------------------------------------------------------------

# Create a 3x3 matrix

my\_matrix <- matrix(1:9, nrow = 3, byrow = TRUE)

# Print the matrix

print(my\_matrix)

# Accessing a specific element (2nd row, 3rd column)

print(my\_matrix[2, 3])

# Accessing a specific row or column

print(my\_matrix[1, ]) # First row

print(my\_matrix[, 2]) # Second column

# Define a matrix with 3 rows and 4 columns

my\_matrix <- matrix(

data = 1:12, # Fill with numbers from 1 to 12

nrow = 3, # Number of rows

ncol = 4, # Number of columns

byrow = TRUE # Fill by row (left to right)

)

# Print the matrix

print(my\_matrix)

# 2. Variables

----------------------------------------------------------------

# we define variable x and y

x <- 40

x

y <-30

y

# Call x and y and apply subtration

x-y

----------------------------------------------------------------

# Creating a data frame

my\_data <- data.frame(

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

Age = c(25, 30, 35),

Score = c(90, 85, 88)

)

# Print the data frame

print(my\_data)

# Accessing a specific column

print(my\_data$Name)

# Accessing a specific row and column

print(my\_data[2, "Age"])

# Subset of the data

my\_data\_2 <- data.frame(

Name = c("Alice", "Bob", "Charlie", "Joy", "David", "Emma", "Frank", "Grace", "Henry", "Isla"),

Age = c(25, 30, 35, 38, 40, 45, 50, 55, 60, 65),

Score = c(72, 85, 88, 90, 92, 94, 96, 98, 99, 100)

)

subset(my\_data\_2, Age > 25) # Get the subset of the data

# Subset of the data frame or vector

high\_scores <- subset(my\_data\_2, Score > 90)

print(high\_scores)

# Corellation

cor(my\_data\_2$Age, my\_data\_2$Score)

#Frequency table

table(my\_data\_2$Age)

# Plot a scatter plot of Age vs. Score

plot(my\_data\_2$Age, my\_data\_2$Score,

main = "Age vs Score",

xlab = "Age",

ylab = "Score",

col = "blue",

pch = 16)

# Create a data frame

my\_data\_3 <- data.frame(

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

Age = c(25, 30, 35),

Score = c(88, 95, 91)

)

print(my\_data\_3)

# Accessing a specific row and column

print(my\_data\_3[2, "Age"])

----------------------------------------------------------------

add\_numbers <- function(a, b)

{

return(a + b)

}

# Calling the function

result <- add\_numbers(5, 3)

print(result)

# Create a function to find the difference of two numbers

diff\_numbers <- function(a, b) {

return(a - b)

}

# Call the function

result <- diff\_numbers(100, 10)

print(result)

----------------------------------------------------------------

# Built-in Functions:

# Getting the mean of a vector

mean\_val <- mean(num\_vector)

print(mean\_val)

# Getting the sum of a vector

sum\_val <- sum(num\_vector)

print(sum\_val)

# Finding the length of a vector

len\_val <- length(num\_vector)

print(len\_val)

----------------------------------------------------------------

#Control Flow in R

#Conditional Statements (if, else):

x <- 10

if (x > 5) {

print("x is greater than 5")

} else {

print("x is less than or equal to 5")

}

----------------------------------------------------------------

# Loops (for, while): Loops are useful when you need to repeat operations.

#for loop:

for (i in 1:5) {

print(i)

}

#for loop:

for (i in 1:7) {

print(i)

}

#while loop:

i <- 1

while (i <= 5)

{

print(i)

i <- i + 1

}

----------------------------------------------------------------

#Plotting in R

# Simple scatter plot

plot(num\_vector)

# Histogram

hist(num\_vector)

# Boxplot

boxplot(num\_vector)

# Help with a function

help(mean)

# Shortcut for help

?mean

# Combine elements into a vector

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

#Compute mean / average of the vector

my\_vector\_mean <- mean(my\_vector)

print(my\_vector\_mean)

# Calculate sum of elements in the vector

sum(my\_vector)

# Calculate standard deviation of elements in the vector

sd(my\_vector)

# Calculate the length of the vector

length(my\_vector)

#Structure of the vector

str(my\_vector)

#Summary statistics

summary(my\_vector)

#Preview data head/tail

head(my\_vector, 3) # First 3 elements

tail(my\_vector, 3) # Last 3 elements

# Simple plot of the vector

plot(my\_vector)

# Apply function to a matrix or dataframe

apply(my\_matrix, 1, sum) # Sum across rows

apply(my\_matrix, 2, mean) # Mean across columns

# Plot a scatter plot of Age vs. Score

plot(my\_data$Age, my\_data$Score,

main = "Age vs Score",

xlab = "Age",

ylab = "Score",

col = "blue",

pch = 16)

# IF Statements

# Check if a person's score is above 90

score <- 92

if (score > 90) {print("Great job!")

} else {print("Keep trying!")}

# Create a Random Number Generator

# Generate 5 random numbers between 1 and 100

random\_numbers <- sample(1:100, 5)

print(random\_numbers)

# Generate a random number and check if it's odd or even

random\_num <- sample(1:100, 1)

if (random\_num %% 2 == 0) {

print(paste(random\_num, "is even"))

} else {

print(paste(random\_num, "is odd"))

}

-----------------------------------------------------------

#Using paste() to Combine Strings

# Combine two strings

first\_name <- "John"

last\_name <- "Doe"

full\_name <- paste(first\_name,last\_name)

print(full\_name)

# Create a bar plot of scores

barplot(my\_data$Score,

names.arg = my\_data$Name,

main = "Scores of Individuals",

col = "lightblue",

border = "pink")

gender <- factor(c("Male", "Female", "Female", "Male"))

gender

#Using sample() to Shuffle Data

# Shuffle the order of names

shuffled\_names <- sample(my\_data$Name)

print(shuffled\_names)

# Create a histogram of scores

hist(my\_data$Score,

main = "Distribution of Scores",

xlab = "Scores",

col = "green",

border = "black")

# Simple Pie Chart

# Create a pie chart for gender distribution

gender\_counts <- table(gender)

pie(gender\_counts,

main = "Gender Distribution",

col = c("pink", "blue"))

# Example gender vector

gender <- c("Female", "Male", "Female", "Male", "Female", "Male", "Female", "Male", "Male", "Female")

# Count genders

gender\_counts <- table(gender)

# Calculate percentages

gender\_percent <- round(100 \* gender\_counts / sum(gender\_counts), 1)

# Create labels with count and percentage

labels <- paste(names(gender\_counts),

"\n", gender\_counts, " (", gender\_percent, "%)", sep = "")

# Draw the pie chart

pie(gender\_counts,

main = "Gender Distribution",

col = c("pink", "blue"),

labels = labels)