The Basics of R Programming

```r
# 1. Data Types
numeric_var <- 10.5 # Numeric data type
integer_var <- 7L # Integer data type
char_var <- "R Programming" # Character data type
logical_var <- TRUE # Logical data type
# Display Data Types
cat("Data Types:\n")
cat("Numeric Variable:", numeric_var, "Type:", typeof(numeric_var), "\n")
cat("Integer Variable:", integer_var, "Type:", typeof(integer_var), "\n")
cat("Character Variable:", char_var, "Type:", typeof(char_var), "\n")
cat("Logical Variable:", logical_var, "Type:", typeof(logical_var), "\n\n")
# 2. Variables
a <- 5
b <- 3
# Arithmetic Operations
sum <- a + b
difference <- a - b
product <- a * b
quotient <- a / b
cat("Arithmetic Operations:\n")
cat("Sum:", sum, "\n")
cat("Difference:", difference, "\n")
cat("Product:", product, "\n")
cat("Quotient:", quotient, "\n\n")

# 3. Operators
# Relational Operations
is_equal <- (a == b)
is_greater <- (a > b)
cat("Relational Operations:\n")
cat("Is Equal:", is_equal, "\n")
cat("Is Greater:", is_greater, "\n\n")
# Logical Operations
logical_and <- (a > 2) & (b < 5)
logical_or <- (a < 2) | (b > 1)
cat("Logical Operations:\n")
cat("Logical AND:", logical_and, "\n")
cat("Logical OR:", logical_or, "\n\n")
# 4. Control Statements
cat("Control Statements:\n")
# If-Else Statement
```

```
if (a > b) {
cat("a is greater than b\n")
} else {
cat("a is not greater than b\n")
}
# For Loop
cat("For Loop:\n")
for (i in 1:3) {
cat("Iteration:", i, "\n")
}
# 5. Functions
# Define a Function
add_numbers <- function(x, y) {
return(x + y)
}
# Use the Function
result <- add_numbers(10, 15)

cat("Function Result:\n")
cat("Sum of 10 and 15 is:", result, "\n")
```

Program to implement different data structures in R

```
# 1. Creating Vectors
numeric_vector <- c(1.2, 3.4, 5.6, 7.8) # Numeric vector
cat("Vector Created:\n")
cat("Numeric Vector:", numeric_vector, "\n")
# 2. Accessing Elements
cat("Accessing Elements:\n")
cat("First element of numeric_vector:", numeric_vector[1], "\n")
cat("Elements greater than 3 in numeric_vector:", numeric_vector[numeric_vector >
3], "\n")
cat("Second to third elements in numeric_vector:", numeric_vector[2:3], "\n\n")
# 3. Vector Operations
# Arithmetic Operations on Numeric Vectors
sum_vector <- numeric_vector + 2
product_vector <- numeric_vector * 2
cat("Vector Operations:\n")
cat("Sum of numeric_vector + 2:", sum_vector, "\n")
cat("Product of numeric_vector * 2:", product_vector, "\n\n")
# 4. Manipulating Vectors
cat("Manipulating Vectors:\n")
# Adding a new element
```

```
numeric_vector <- c(numeric_vector, 9.0)
cat("Numeric Vector after adding 9.0:", numeric_vector, "\n")
```

 # Modifying an element

```
numeric_vector[2] <- 4.5
cat("Numeric Vector after modifying second element:", numeric_vector, "\n")
```

 # Removing the first element

```
numeric_vector <- numeric_vector[-1]
cat("Numeric Vector after removing the first element:", numeric_vector, "\n\n")
```

 # 5. Vector Functions

```
cat("Vector Functions:\n")
cat("Sum of elements in numeric_vector:", sum(numeric_vector), "\n")
cat("Length of numeric_vector:", length(numeric_vector), "\n")
cat("Mean of numeric_vector:", mean(numeric_vector), "\n")
```

Program to Read a CSV File and Analyze the Data in R

 # 1. Reading the CSV File "sample_data.csv"

```
data <- read.csv("sample_data.csv")
```

 # 2. Display the Preview and Summary of the data (first five records)

```
cat("Data Preview:\n")
print(head(data,5))
cat("\nSummary of the Data:\n")
print(summary(data))
```

 # 3. Analyzing the Data

 # Calculate mean, median, and mode for a numeric column (Assuming 'Age' column exists)

```
cat("\nBasic Analysis:\n")
age_mean <- mean(data$Age, na.rm = TRUE)
age_median <- median(data$Age, na.rm = TRUE)
age_mode <- as.numeric(names(sort(table(data$Age), decreasing=TRUE)[1]))
cat("Mean Age:", age_mean, "\n")
cat("Median Age:", age_median, "\n")
cat("Mode Age:", age_mode, "\n\n")
```

 # Generate a frequency table for Gender column

```
gender_table <- table(data$Gender)
cat("Gender Distribution:\n")
print(gender_table)
```

 # Visualize the data using a simple bar plot for 'Gender'

```
barplot(gender_table, main = "Gender Distribution", col = "lightblue")
```

sample_data.csv:

```
ID,Name,Age,Gender,Salary
```

1,John,28,Male,50000
2,Emma,32,Female,60000
3,Alex,25,Male,45000
4,Maria,29,Female,52000
5,James,31,Male,55000
6,Sophia,30,Female,58000
7,Chris,26,Male,47000
8,Linda,33,Female,61000
9,David,31,Male,48000

Exercise 4: Program to Find Mean, Median, and Standard Deviation for a Given Discrete Probability Distribution

```
# Step 1: Define the discrete probability distribution
values <- c(1, 2, 3, 4, 5) # Given values
probabilities <- c(0.1, 0.2, 0.3, 0.2, 0.2) # Corresponding probabilities
# Step 2: Calculate the Mean
mean_value <- sum(values * probabilities)
# Step 3: Calculate the Median
median_value <- median(rep(values, probabilities * 100))
# Step 4: Calculate the Standard Deviation
variance <- sum((values - mean_value)^2 * probabilities)
std_dev <- sqrt(variance)
# Output the results
cat("Mean:", mean_value, "\n")
cat("Median:", median_value, "\n")
cat("Standard Deviation:", std_dev, "\n")
```

Exercise 5: Program to Represent a Given Data in the Form of Graphs

```
# Step 1: Define the data
# For Bar Chart
categories <- c("A", "B", "C", "D")
values <- c(10, 20, 15, 25)
# For Histogram
data <- c(5, 7, 8, 9, 10, 10, 12, 14, 15, 16, 17, 20)
# For Scatter Plot
x <- c(1, 2, 3, 4, 5)
y <- c(2, 3, 5, 7, 11)
# Step 2: Create a Bar Chart
barplot(values, names.arg = categories, main = "Bar Chart of Categories", xlab =
"Categories", ylab
```

= "Values", col = "blue")
 # Step 3: Create a Histogram
hist(data, breaks = 5, main = "Histogram of Data", xlab = "Values", ylab = "Frequency", col = "green")
 # Step 4: Create a Scatter Plot
plot(x, y, main = "Scatter Plot of X vs Y", xlab = "X", ylab = "Y", pch = 19, col = "red")

Exercise 5.1: Program to Represent a Given Data in the Form of Graphs

 # 1. Histogram
 # Creating a dataset of student marks
student_marks <- c(55, 65, 70, 72, 73, 90, 95, 100, 75)
 # Plotting a Histogram
hist(student_marks,
main = "Histogram of Student Marks",
xlab = "Marks",
ylab = "Frequency",
col = "lightblue",
border = "black")
 # 2. Bar Plot
 # Creating a dataset of student names and marks
student_names <- c("John", "Alice", "Bob", "Emma", "Lily")
student_scores <- c(45, 55, 65, 55, 35)
 # Plotting a Bar Plot
barplot(student_scores,
names.arg = student_names,
main = "Bar Plot of Student Scores",
xlab = "Students",
ylab = "Scores",
col = "lightgreen",
border = "black")

 # 3. Line Chart
 # Creating a dataset of monthly temperatures
months<-c(1,2,3,4,5)
temperature <- c(10, 15, 5, 20, 28)
 # Plotting a Line Chart
plot(months, temperature,
type = "o",
col = "blue",
main = "Line Chart of Monthly Temperatures",

```
xlab = "Months",
ylab = "Temperature (°C)")
# 4. Pie Chart
# Creating a dataset of department-wise students
dept <- c("IT", "Math", "Physics", "Biology")
students <- c(120, 80, 60, 40)
# Plotting a Pie Chart
pie(students,
labels = dept,
main = "Pie Chart of Department-Wise Students",
col = rainbow(length(dept)))
# 5. Box Plot
# Creating a dataset of exam scores
exam_scores <- c(55, 65, 70, 80, 85, 90, 95, 100, 50, 75)
# Plotting a Box Plot
boxplot(exam_scores,
main = "Box Plot of Exam Scores",
ylab = "Scores",
col = "lightcoral",
border = "black")
# 6. Scatter Plot
# Creating datasets for height and weight
height <- c(150, 160, 165, 170, 175)
weight <- c(50, 55, 60, 65, 70)
# Plotting a Scatter Plot
plot(height, weight,

main = "Scatter Plot of Height vs. Weight",
xlab = "Height (cm)",
ylab = "Weight (kg)",
pch = 16,
col = "darkred")
```

Exercise 6: Simple Linear Regression Analysis in R

```
# 1. Create Dataset
# Independent variable: Hours of study
# Dependent variable: Marks obtained
hours <- c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
marks <- c(50, 55, 60, 65, 70, 75, 80, 85, 90, 95)
# Plot the scatter plot
plot(hours, marks,
main = "Scatter Plot of Hours vs. Marks",
```

```
xlab = "Hours of Study",
ylab = "Marks Obtained",
pch = 16,
col = "blue")
# 2. Perform Simple Linear Regression
model <- lm(marks ~ hours)
# 3. View Model Summary
print(summary(model))
# 4. Plot the Regression Line
abline(model, col = "red")
# 5. Predict Marks for 6.5 hours of study
predicted_marks <- predict(model, data.frame(hours = 6.5))
cat("Predicted Marks for 6.5 hours of study:", predicted_marks, "\n")
```

Exercise 7: Chi-Square Test for Independence in R

```
# 1. Create Contingency Table
# Let's consider a study about the preference of two types of sports (Football
and
Basketball)
# among males and females in a sample of 100 individuals.
# Contingency Table: Rows = Gender, Columns = Sport Preference
gender_sport <- matrix(c(30, 10, 20, 40), nrow = 2, byrow = TRUE,
dimnames = list("Gender" = c("Male", "Female"),
"Sport" = c("Football", "Basketball")))
# Display the contingency table
cat("Contingency Table:\n")
print(gender_sport)
# 2. Perform Chi-Square Test for Independence
chi_square_test <- chisq.test(gender_sport)
# 3. View Test Results
cat("\nChi-Square Test Results:\n")
print(chi_square_test)
# 4. Interpretation
if (chi_square_test$p.value < 0.05) {
cat("\nConclusion: The variables are not independent (reject the null
hypothesis).\n")
} else {
cat("\nConclusion: The variables are independent (fail to reject the null
hypothesis).\n")
}
```

Exercise 8: One-Way ANOVA in R

```
# 1. Create a Dataset
# Example: Test scores of students from three different groups (Group A,
Group B, and
Group C)
scores <- c(88, 90, 85, 92, 95, 78, 82, 87, 91, 86, 83, 89, 94, 80, 77)
groups <- factor(c(rep("Group A", 5), rep("Group B", 5), rep("Group C", 5)))
# Combine into a data frame
data <- data.frame(scores, groups)
# Display the dataset
cat("Dataset:\n")
print(data)
# 2. Perform One-Way ANOVA
anova_result <- aov(scores ~ groups, data = data)
# 3. View the ANOVA Summary
cat("\nOne-Way ANOVA Results:\n")
print(summary(anova_result))
# 4. Interpretation
# Check p-value from the ANOVA summary
p_value <- summary(anova_result)[[1]]["Pr(>F)"][1]
if (p_value < 0.05) {
cat("\nConclusion: There is a significant difference between the group means (reject
the null
hypothesis).\n")
} else {
cat("\nConclusion: There is no significant difference between the group means (fail
to reject the
null hypothesis).\n")
}
```

Exercise 9: Two-Sample T-Test in R

```
# 1. Create a Dataset
# Example: Weights of individuals in Group 1 and Group 2
group1 <- c(68, 72, 65, 70, 74) # Weights in Group 1
group2 <- c(60, 63, 67, 69, 64) # Weights in Group 2
# Display the samples
cat("Group 1 Weights:\n")
print(group1)
cat("Group 2 Weights:\n")
print(group2)
# 2. Perform Two-Sample T-Test
t_test_result <- t.test(group1, group2)
# 3. View the T-Test Results
```

```
cat("\nTwo-Sample T-Test Results:\n")
print(t_test_result)
```
 # 4. Interpretation
 # Check p-value from the t-test result
```
p_value <- t_test_result$p.value
if (p_value < 0.05) {
cat("\nConclusion: There is a significant difference between the means of the two groups (reject
the null hypothesis).\n")
} else {
cat("\nConclusion: There is no significant difference between the means of the two groups (fail to
reject the null hypothesis).\n")
}
```

Exercise 10: Plotting Various Probability Distributions in R

 # 1. Plot Normal Distribution
```
cat("Plotting Normal Distribution:\n")
x_normal <- seq(-10, 10, length = 100)
y_normal <- dnorm(x_normal, mean = 0, sd = 1)
```
 # Plot Normal Distribution
```
plot(x_normal, y_normal, type = "l", col = "blue", lwd = 2,
main = "Normal Distribution", xlab = "x", ylab = "Density")
```
 # 2. Plot Binomial Distribution
```
cat("Plotting Binomial Distribution:\n")
x_binom <- 0:10
y_binom <- dbinom(x_binom, size = 10, prob = 0.5)
```
 # Plot Binomial Distribution
```
barplot(y_binom, names.arg = x_binom, col = "green",
main = "Binomial Distribution", xlab = "Number of Successes", ylab = "Probability")
```
 # 3. Plot Poisson Distribution
```
cat("Plotting Poisson Distribution:\n")
x_pois <- 0:15
y_pois <- dpois(x_pois, lambda = 4)
```
 # Plot Poisson Distribution
```
barplot(y_pois, names.arg = x_pois, col = "red",
main = "Poisson Distribution", xlab = "Number of Events", ylab = "Probability")
```
 # 4. Plot Exponential Distribution
```
cat("Plotting Exponential Distribution:\n")
x_exp <- seq(0, 5, length = 100)

y_exp <- dexp(x_exp, rate = 1)
```

 # Plot Exponential Distribution

```
plot(x_exp, y_exp, type = "l", col = "purple", lwd = 2,
main = "Exponential Distribution", xlab = "x", ylab = "Density")
```