PROBABILITY, STATISTICS AND SIMULATION- LABORATORY MANUAL

R-2023

IV SEMESTER

MA23435- PROBABILITY, STATISTICS AND SIMULATION

Common to IV Sem. B.E. – Computer Science and Engineering & Computer Science and

Engineering

(Cyber Security) and B.Tech. – Information Technology



NAME :----------------------------------------

REG. NO :-------------------------------------

BRANCH :--------------------------------------

DEPARTMENT OF HUMANITIES AND SCIENCES-MATHEMATICS

RAJALAKSHMI ENGINEERING COLLEGE (AUTONOMOUS)

THANDALAM –602105

I .BASIC FUNCTIONS IN R AND PLOTTING

Lab Manual for Basic Functions in R and Plotting

Aim

The aim of this lab manual is to introduce students to basic functions in R for data manipulation,

arithmetic operations, statistical functions, and plotting. Students will learn how to perform basic

calculations, use built-in functions, create simple plots, and interpret the results.

Procedure

1. Introduction to Basic Functions in R

o Explain the importance of basic functions for data analysis and manipulation.

o Introduce various types of functions in R: arithmetic, statistical, mathematical,

etc.

o Discuss how plotting functions can help visualize data and analyze trends.

2. Setting Up R Environment

o Ensure R and RStudio are installed.

o Familiarize students with the RStudio interface and the R console.

o Load necessary packages for plotting (e.g., ggplot2 for advanced plotting).

3. Arithmetic Operations and Mathematical Functions

o Example: Basic Arithmetic Operations

# Addition

result\_add <- 5 + 3

cat("Addition:", result\_add, "\n")

# Subtraction

result\_sub <- 7 - 2

cat("Subtraction:", result\_sub, "\n")

# Multiplication

result\_mult <- 4 \* 6

cat("Multiplication:", result\_mult, "\n")

# Division

result\_div <- 10 / 2

cat("Division:", result\_div, "\n")

o Example: Mathematical Functions

# Square root

sqrt\_result <- sqrt(25)

cat("Square root:", sqrt\_result, "\n")

# Exponential function

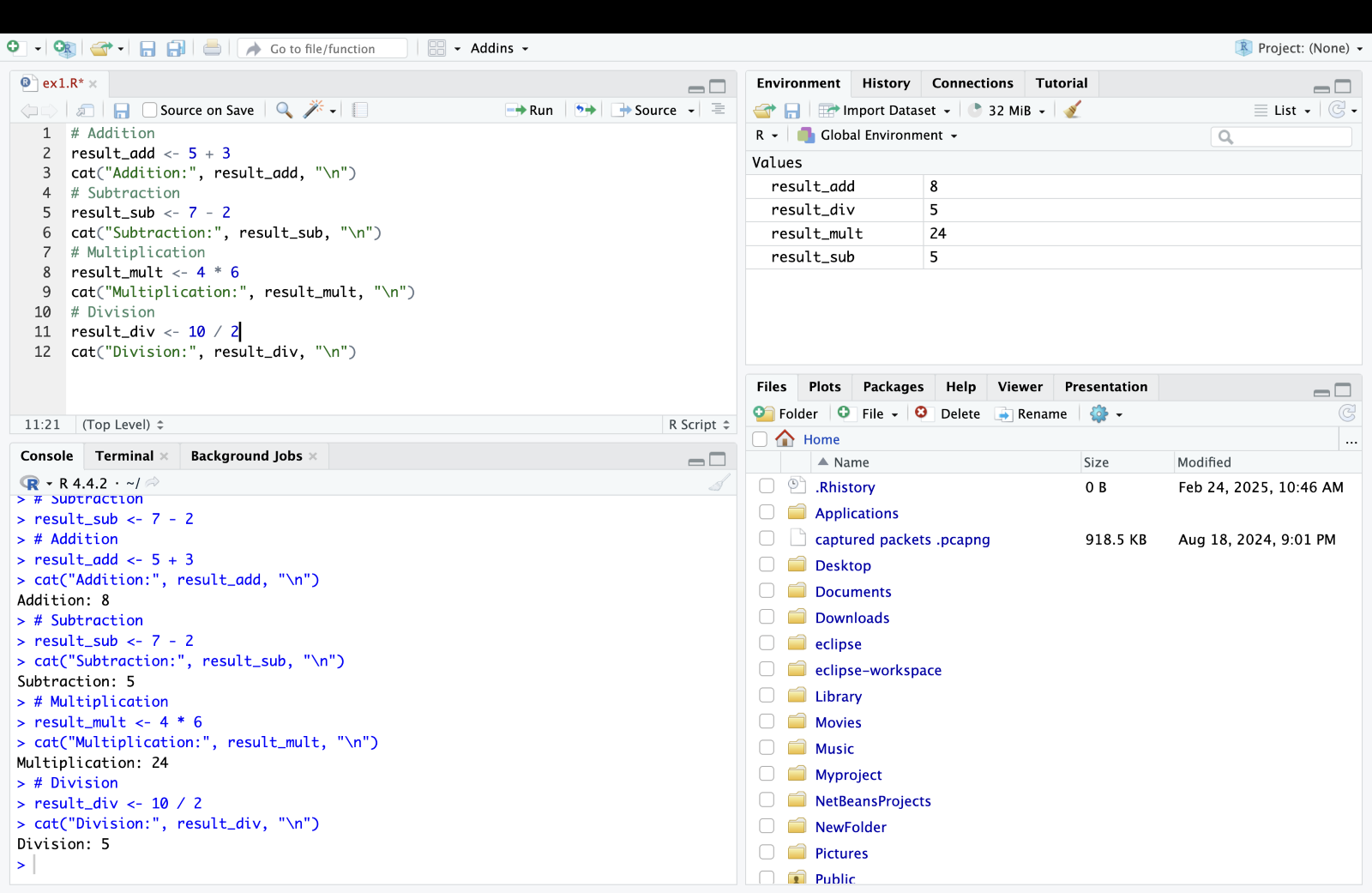
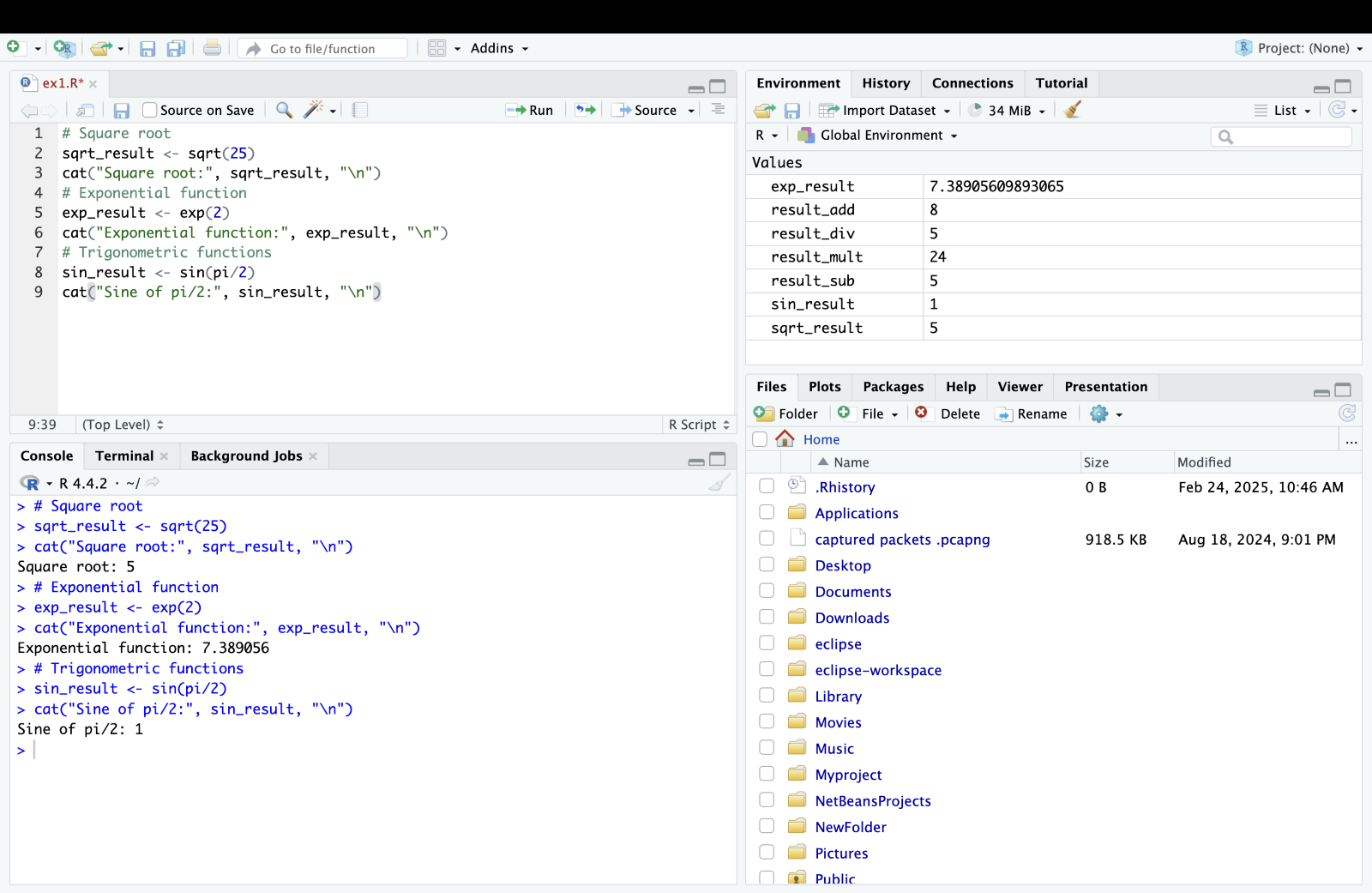
exp\_result <- exp(2)

cat("Exponential function:", exp\_result, "\n")

# Trigonometric functions

sin\_result <- sin(pi/2)

cat("Sine of pi/2:", sin\_result, "\n")

4. Statistical Functions

o Example: Using Statistical Functions

# Generate random data

set.seed(123)

data <- rnorm(100)

# Mean

mean\_result <- mean(data)

cat("Mean:", mean\_result, "\n")

# Standard deviation

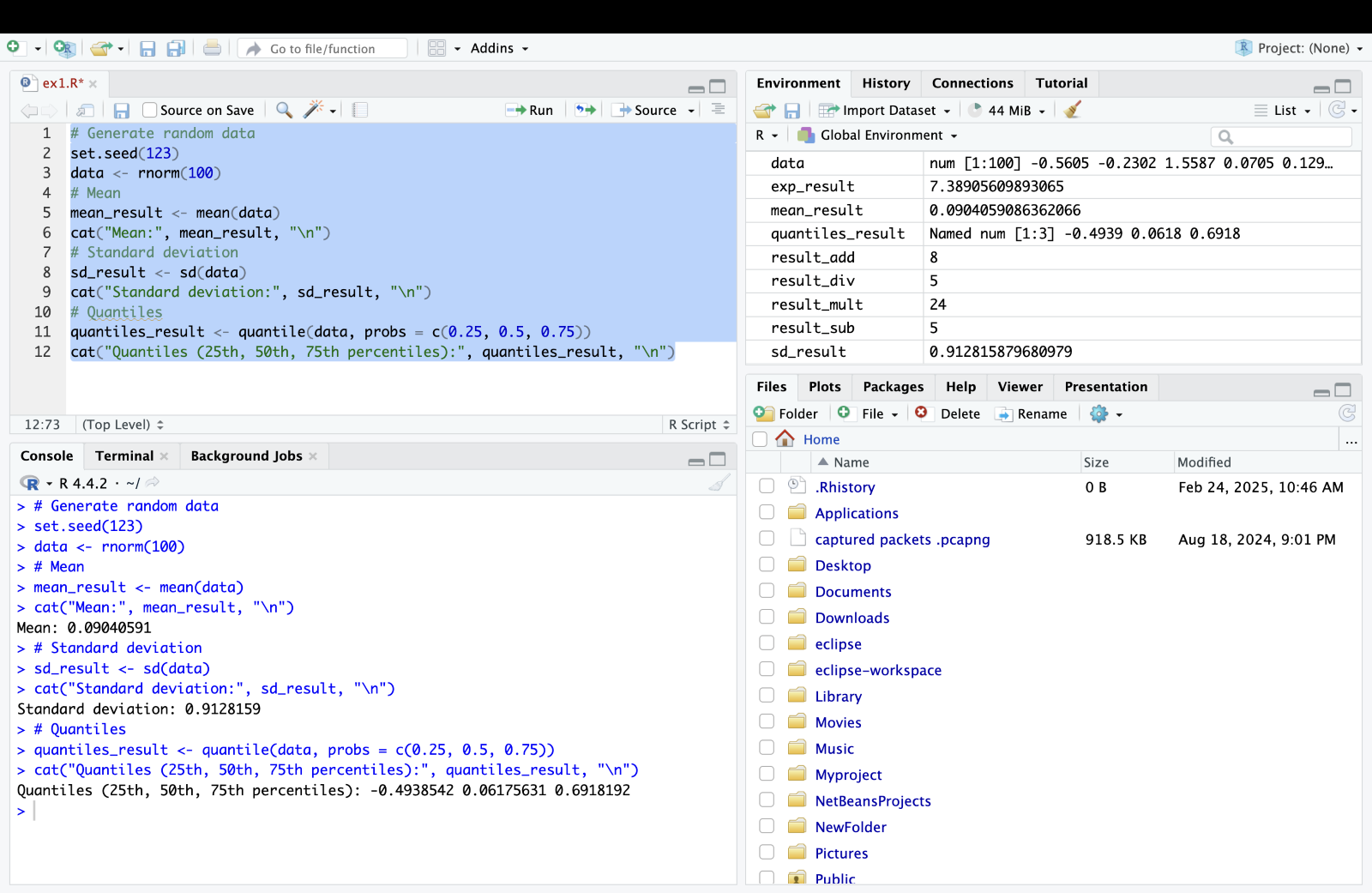
sd\_result <- sd(data)

cat("Standard deviation:", sd\_result, "\n")

# Quantiles

quantiles\_result <- quantile(data, probs = c(0.25, 0.5, 0.75))

cat("Quantiles (25th, 50th, 75th percentiles):", quantiles\_result, "\n")



5. Plotting Data

o Example: Creating Basic Plots

# Generate data for plotting

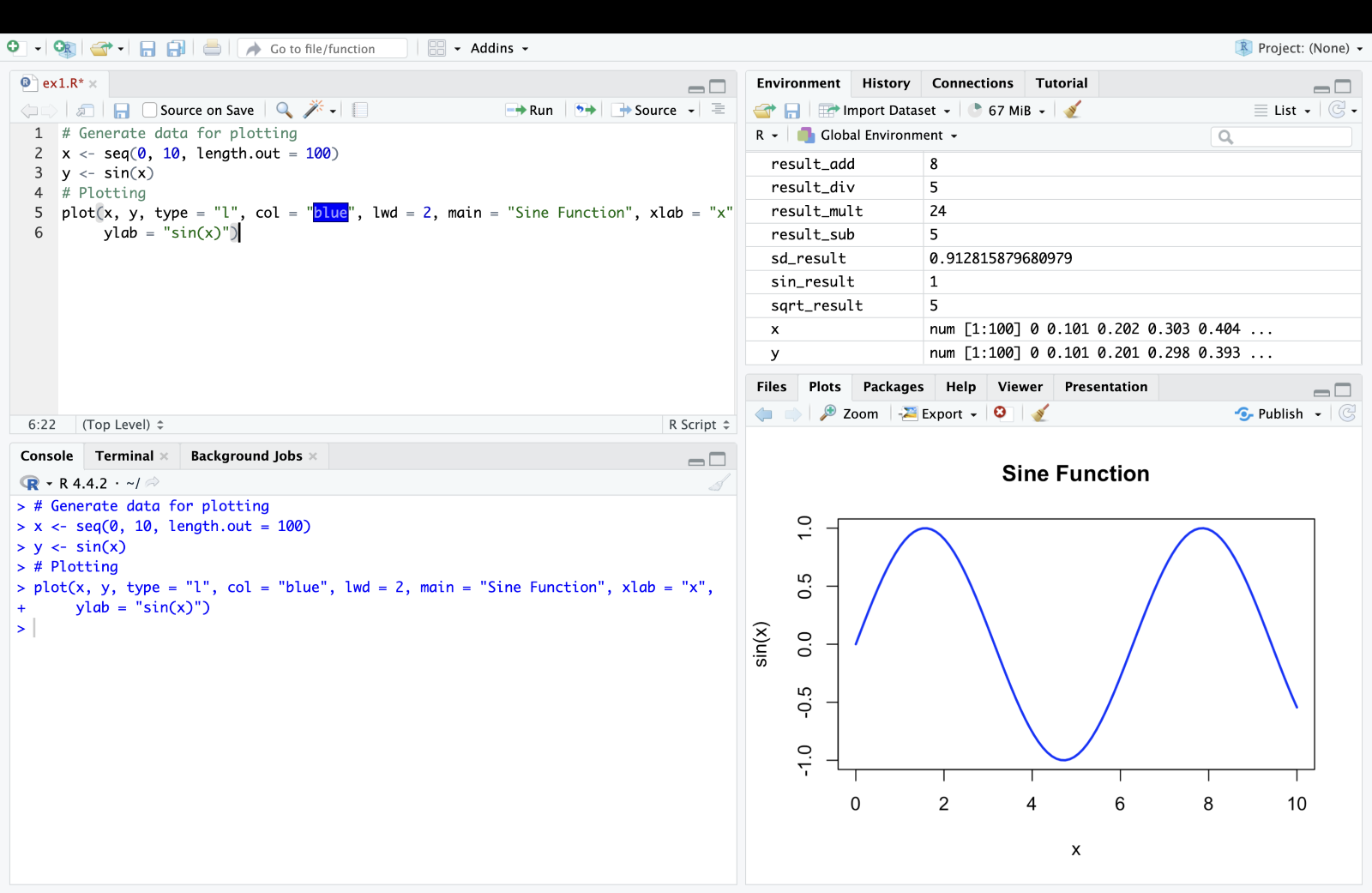
x <- seq(0, 10, length.out = 100)

y <- sin(x)

# Plotting

plot(x, y, type = "l", col = "blue", lwd = 2, main = "Sine Function", xlab = "x",

ylab = "sin(x)")



Exercises

1. Exercise 1: Arithmetic Operations

o Task: Perform arithmetic operations to calculate the area of a rectangle with

length 8 units and width 5 units. Print the result.

o Expected Output: Display the calculated area of the rectangle.

# Calculate area of rectangle

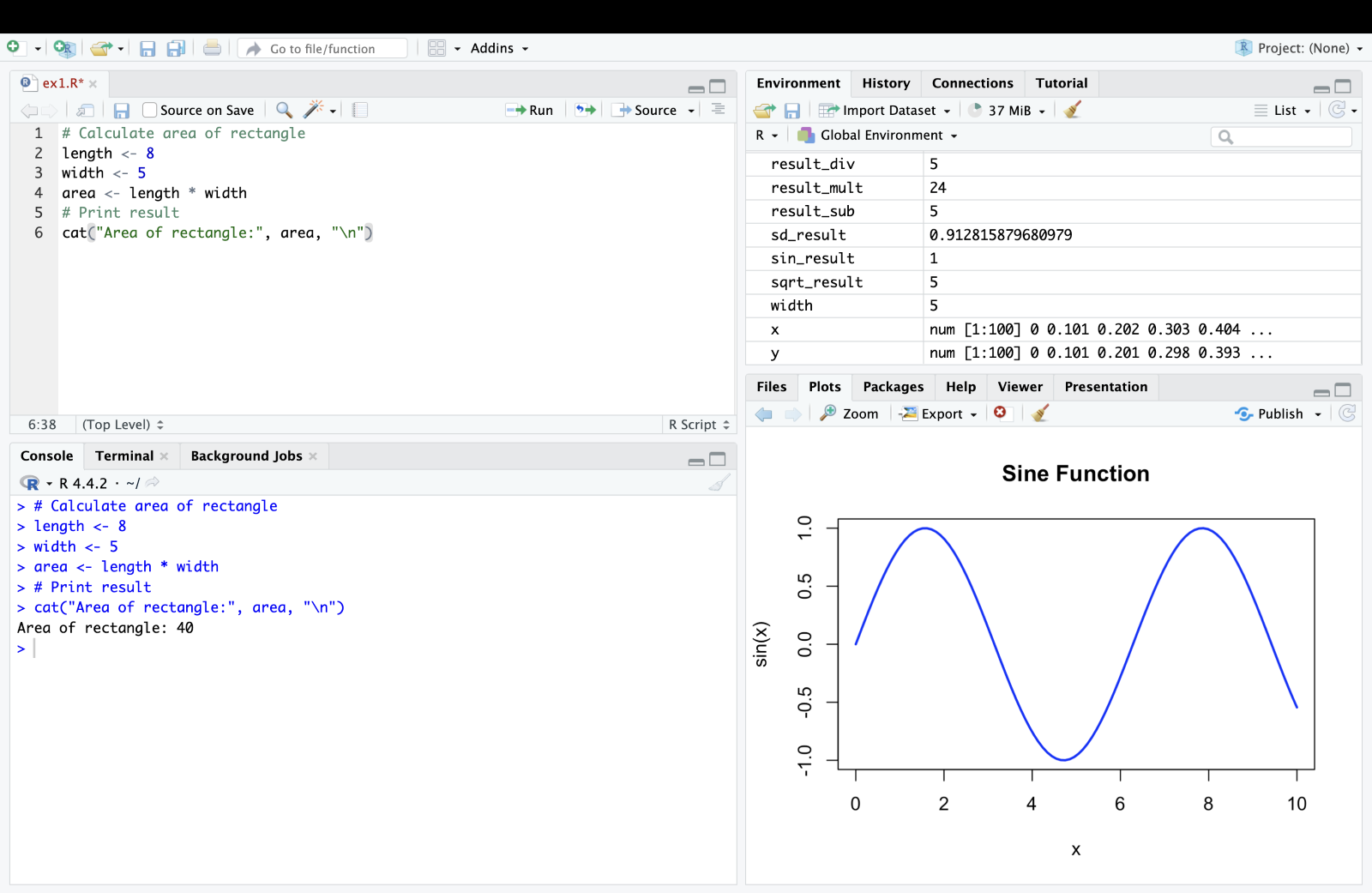
length <- 8

width <- 5

area <- length \* width

# Print result

cat("Area of rectangle:", area, "\n")



2. Exercise 2: Mathematical Functions

o Task: Use R functions to calculate the factorial of 6 and the natural logarithm of

10. Print the results.

o Expected Output: Display the factorial and logarithm results.

# Calculate factorial and logarithm

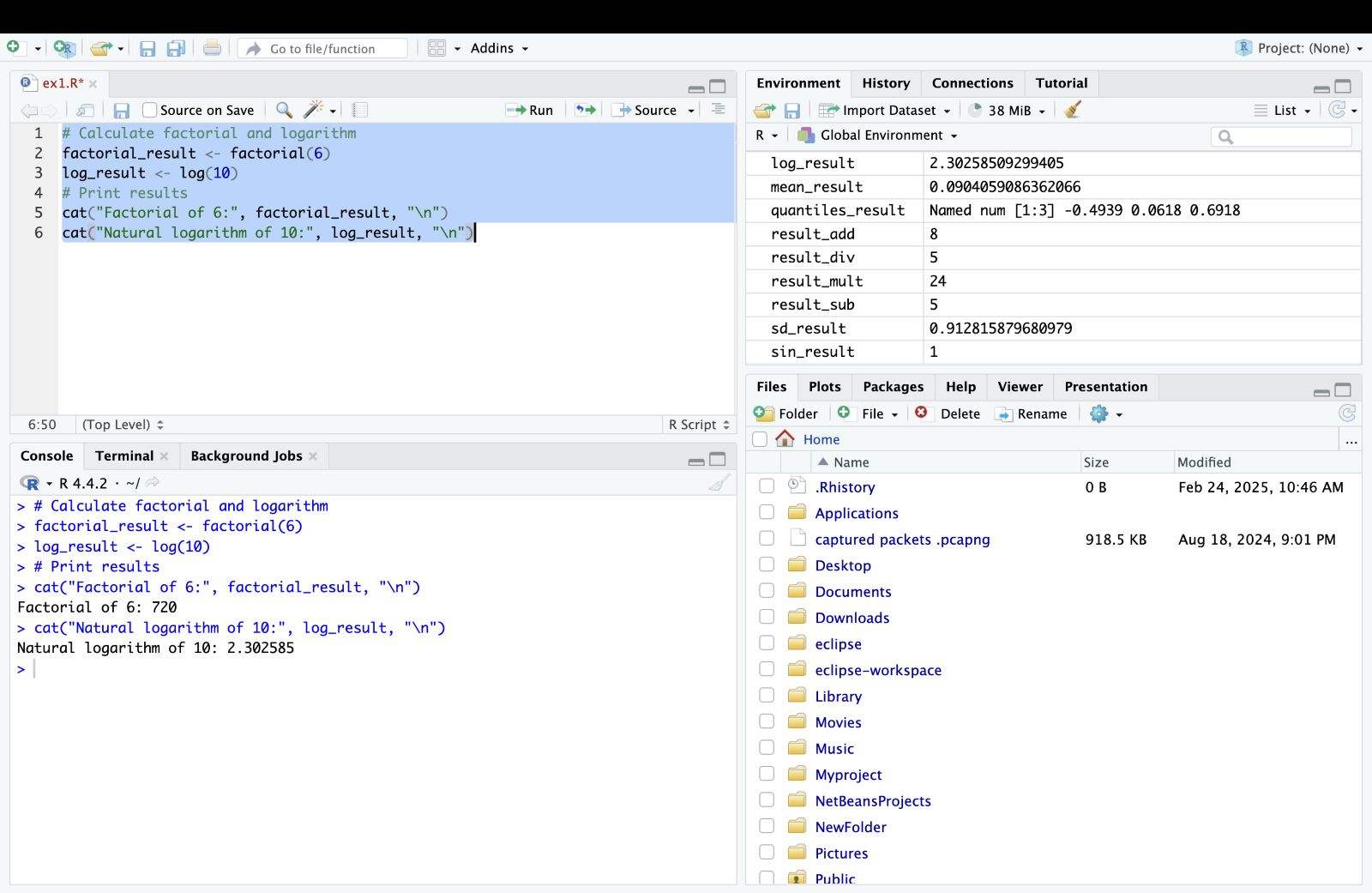
factorial\_result <- factorial(6)

log\_result <- log(10)

# Print results

cat("Factorial of 6:", factorial\_result, "\n")

cat("Natural logarithm of 10:", log\_result, "\n")



3. Exercise 3: Statistical Functions and Plotting

o Task: Generate 100 random numbers from a normal distribution with mean 50

and standard deviation 10. Calculate the mean and standard deviation of the

generated data. Plot a histogram of the data.

o Expected Output: Display the calculated mean and standard deviation, and

visualize the histogram.

# Generate random data

set.seed(456)

data <- rnorm(100, mean = 50, sd = 10)

# Calculate mean and standard deviation

mean\_result <- mean(data)

sd\_result <- sd(data)

# Print results

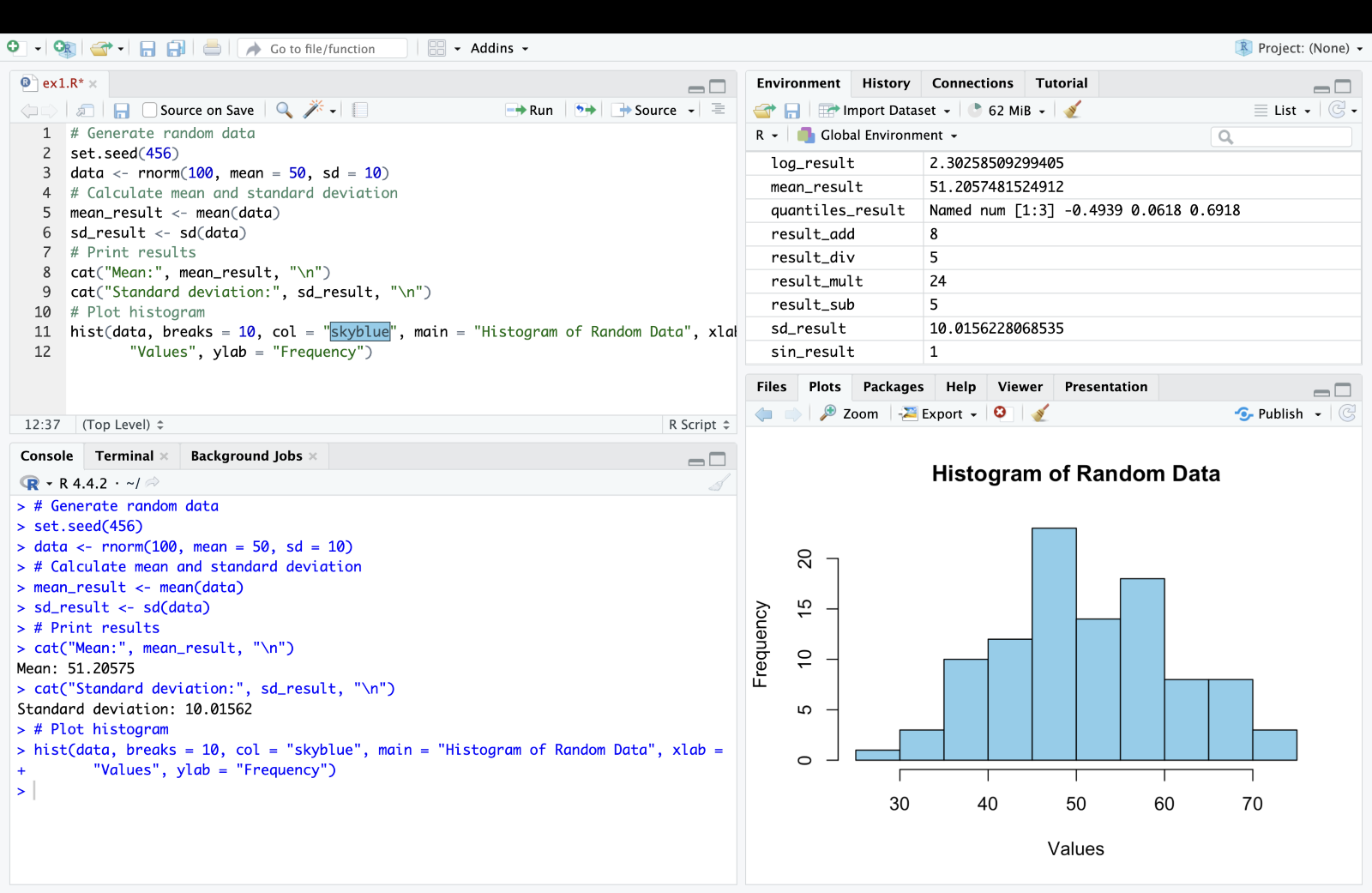
cat("Mean:", mean\_result, "\n")

cat("Standard deviation:", sd\_result, "\n")

# Plot histogram

hist(data, breaks = 10, col = "skyblue", main = "Histogram of Random Data", xlab =

"Values", ylab = "Frequency")



Expected Outputs

1. Exercise 1 Output

# Area of rectangle output

2. Exercise 2 Output

# Factorial and logarithm output

3. Exercise 3 Output

# Mean and standard deviation output

# Histogram plot of random data

By the end of this lab, students should be able to confidently perform basic calculations, use

mathematical and statistical functions in R, create simple plots to visualize data, and interpret

results, providing a solid foundation for further exploration in data analysis and visualization

tasks in computer science and information technology.

VIVA QUESTIONS:

1 What is a function in R?

2. How do you create a function in R?

3. What are the arguments in a function?

4. How do you call a function in R?

5.What is the purpose of the return() function in R?

6. How do you handle default argument values in R functions?

7. What is the scope of variables in R functions?

II. Mathematical functions in R – Integration

Aim

The aim of this lab manual is to provide students with hands-on experience in using R for

performing mathematical integrations. Students will learn how to utilize R's built-in functions for

numerical integration, visualize the results, and apply these techniques to solve real-world

problems relevant to computer science and information technology.

Exercises

1. Exercise 1: Integrate a Polynomial Function o Task: Integrate the function f(x)=3x3+2x2+x+1f(x) = 3x^3 + 2x^2 + x + 1f(x)=3x3+2x2+x+1 from 0 to 2. o Visualize: Plot the function and the area under the curve. o Expected Output:

# Define the function

f <- function(x) { 3*x^3 + 2*x^2 + x + 1 }

# Perform integration

result <- integrate(f, lower = 0, upper = 2)

print(result$value)

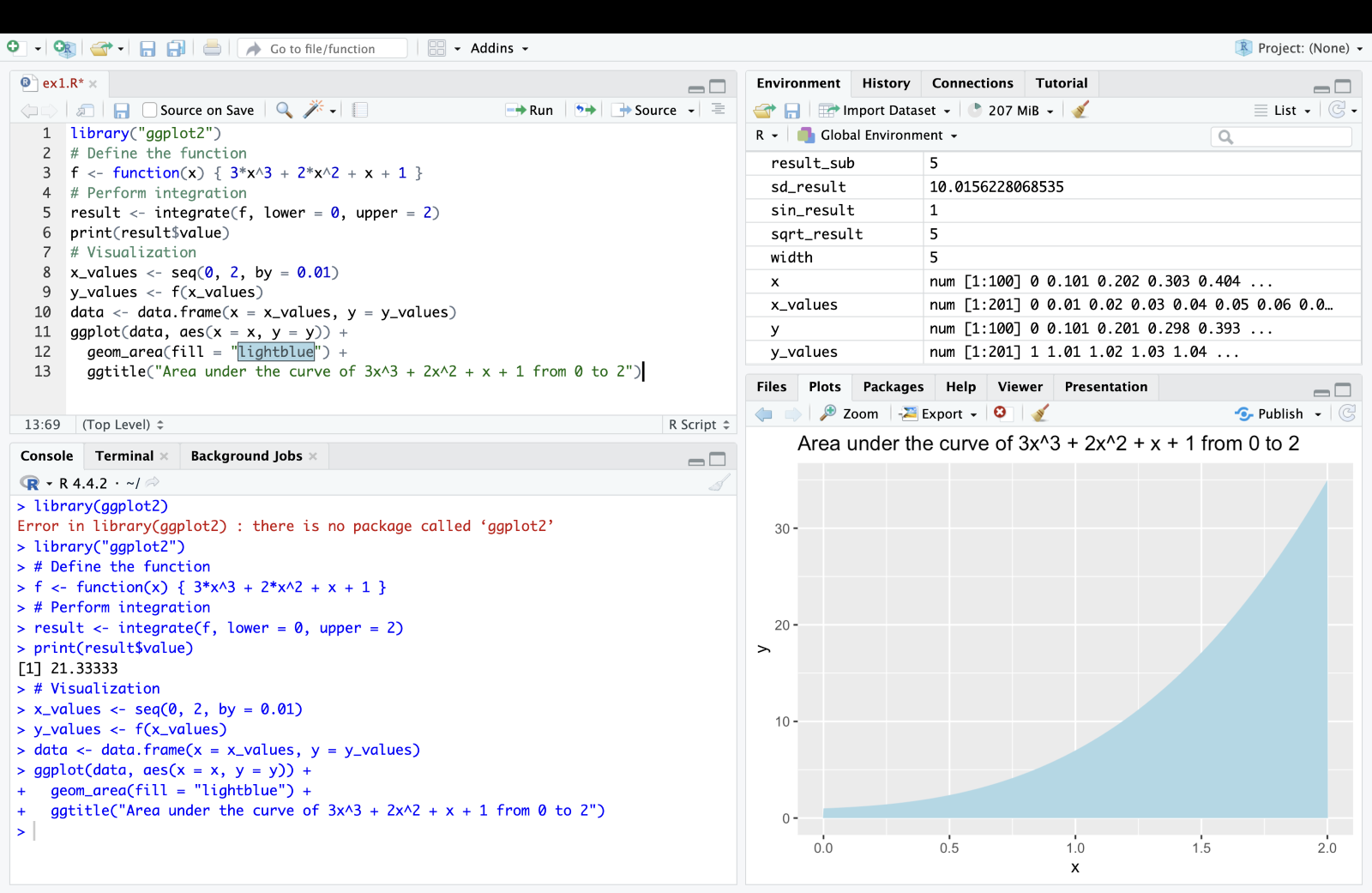
# Visualization

# x\_values <- seq(0, 2, by = 0.01)

y\_values <- f(x\_values)

data <- data.frame(x = x\_values, y = y\_values)

ggplot(data, aes(x = x, y = y)) + geom\_area(fill = "lightblue") + ggtitle("Area under the curve of 3x^3 + 2x^2 + x + 1 from 0 to 2")



Result [1] 21.33333

1. Exercise 2: Integration of Exponential Function o Task: Integrate the function f(x)=exf(x) = e^xf(x)=ex from 1 to 3. o Compare: Compare the result with the analytical solution. o Expected Output:

f <- function(x) { exp(x) }

# **Perform integration**

result <- integrate(f, lower = 1, upper = 3)

print(result$value)

# **Analytical solution**

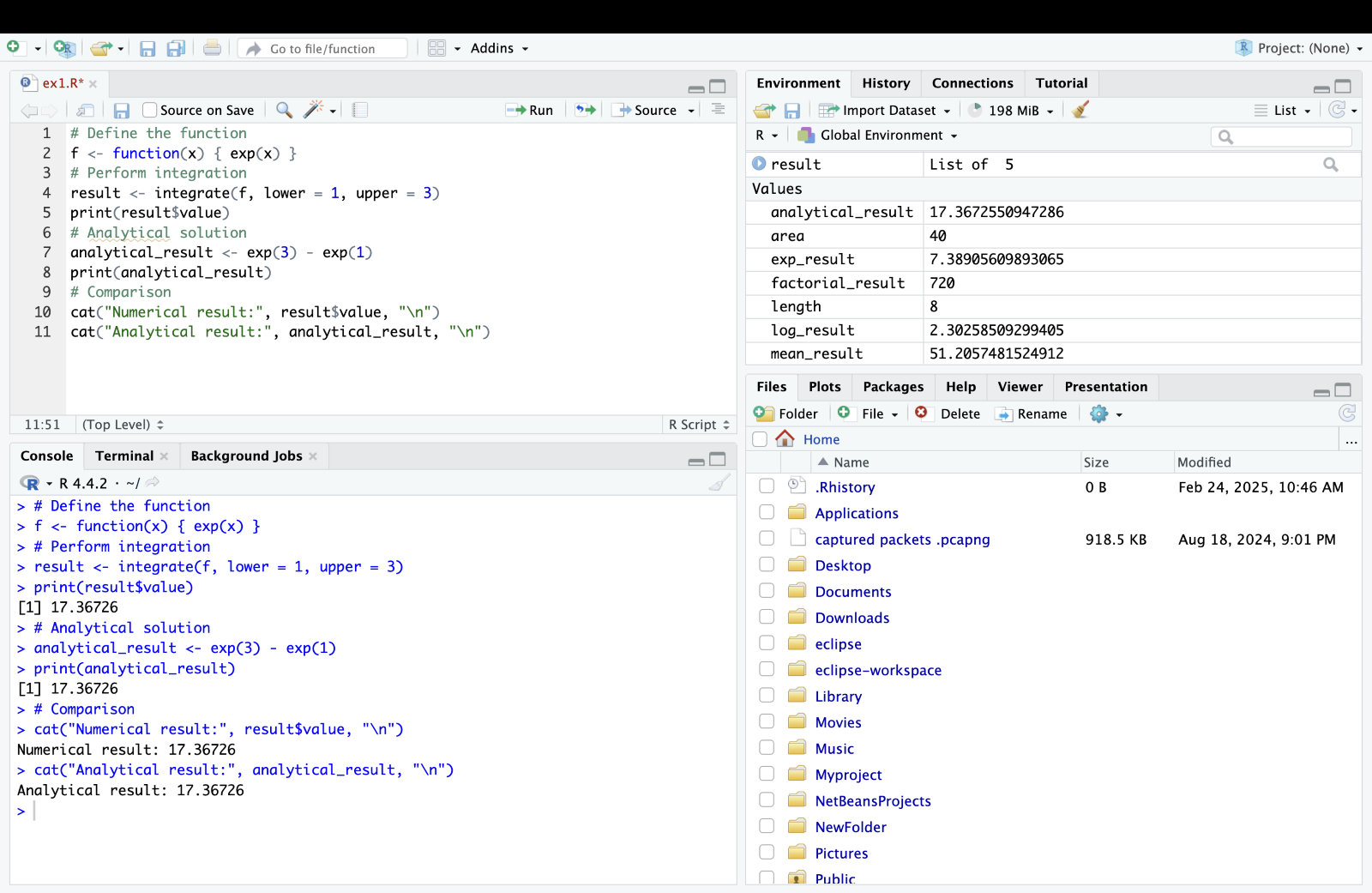
analytical\_result <- exp(3) - exp(1)

print(analytical\_result)

comparision

cat("Numerical result:", result$value, "\n")

cat("Analytical result:", analytical\_result, "\n")



1. Exercise 3: Double Integration

o Task: Perform double integration of the function f(x,y)=xy over the region [0, 1] Expected Output:

# **Define the function**

f <- function(x, y) { x \* y }

# **Perform double integration**

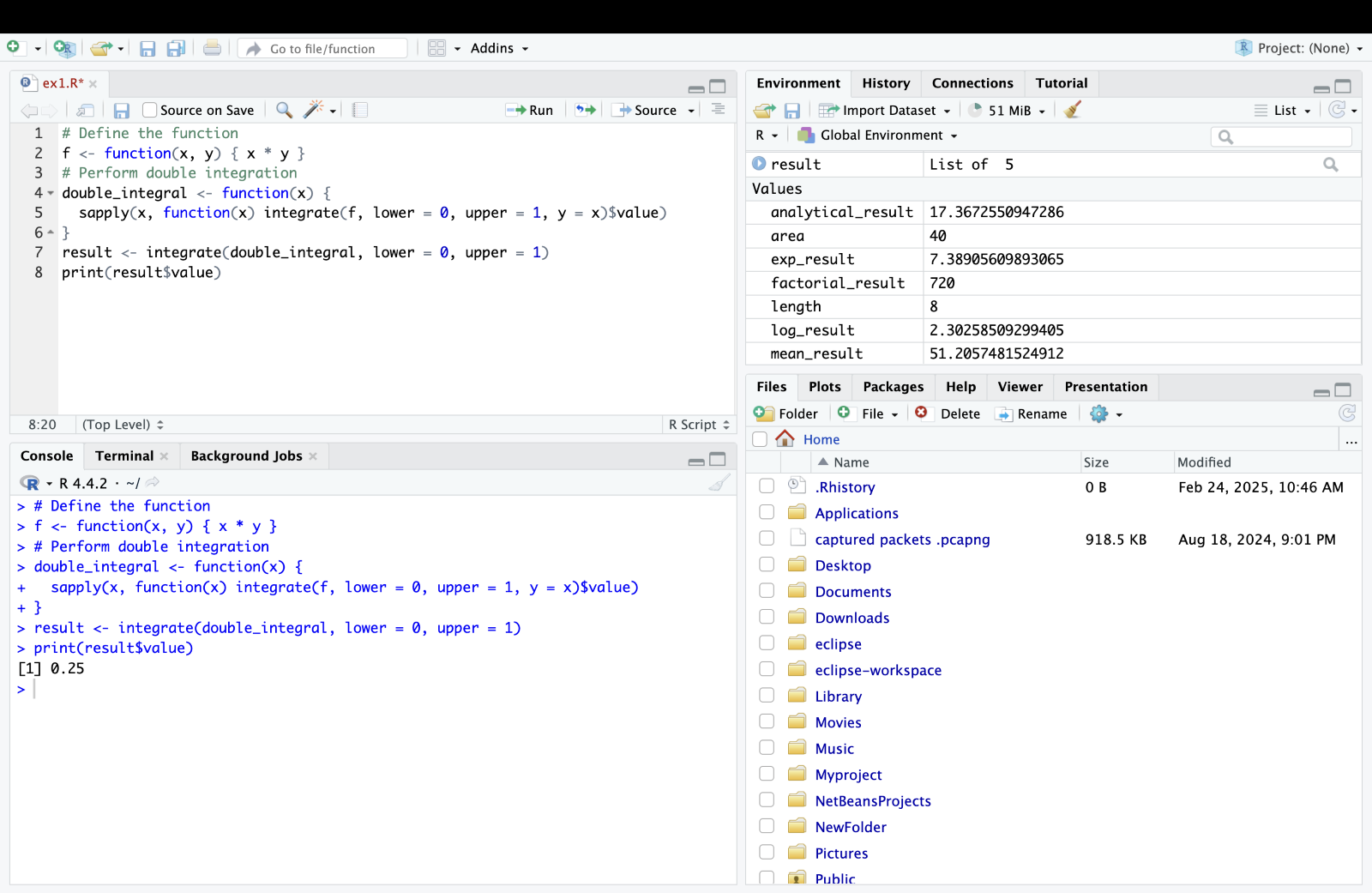
double\_integral <- function(x) {

sapply(x, function(x) integrate(f, lower = 0, upper = 1, y = x)$value)

}

result <- integrate(double\_integral, lower = 0, upper = 1)

print(result$value)



1. Exercise 4: Integration with Real Data o Task: Load a dataset containing time and velocity of a moving object. Compute the distance traveled by integrating the velocity over time. Visualize the velocity and the distance traveled. o Expected Output:

# **Example dataset**

time <- seq(0, 10, by = 0.1)

18

velocity <- 3 \* time^2 - 2 \* time + 1 # Example velocity function

# **Create a data frame**

data <- data.frame(time = time, velocity = velocity)

# **Compute distance traveled using integration**

distance\_function <- function(t) { 3 \* t^2 - 2 \* t + 1 }

result <- integrate(distance\_function, lower = 0, upper = 10)

print(result$value)

# **Cumulative distance**

cumulative\_distance <- cumsum(velocity) \* 0.1

# **Visualization**

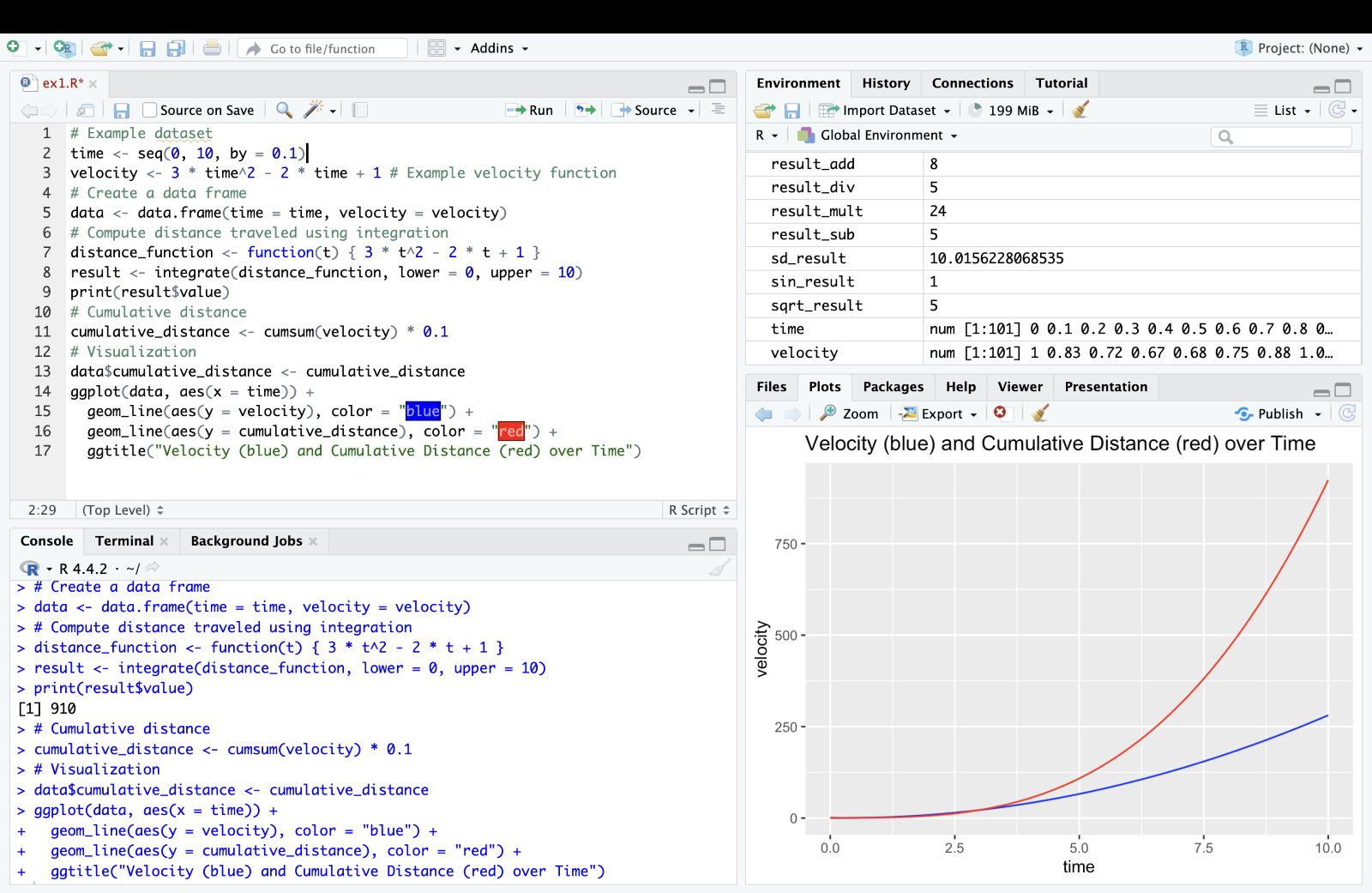
data$cumulative\_distance <- cumulative\_distance

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

geom\_line(aes(y = velocity), color = "blue") +

geom\_line(aes(y = cumulative\_distance), color = "red") +

ggtitle("Velocity (blue) and Cumulative Distance (red) over Time")



Output

By the end of this lab, students should be able to understand and apply numerical integration

techniques in R, visualize the results, and solve practical problems using these methods.

VIVA QUESTIONS:

1. What is numerical integration, and why is it used?

2. What are some of the common numerical integration methods available in R?

3. How do you perform definite integration of a function in R?

4. Explain the use of the integrate() function in R.

5. What arguments does the integrate() function take, and what does it return?

6.How can you handle improper integrals using the integrate() function?

7. Describe how to integrate a function over an infinite interval in R.

III. CONTROL FLOW LOOPS IN R

Aim

The aim of this lab manual is to provide students with hands-on experience in using control flow

loops in R. Students will learn how to utilize for, while, and repeat loops to perform iterative

tasks, understand the application of these loops in real-world problems, and develop their coding

skills in R.

Exercises

1. Exercise 1:

Sum of Even Numbers

o Task: Write a for loop to calculate the sum of even numbers from 1 to 20.

o Expected Output:

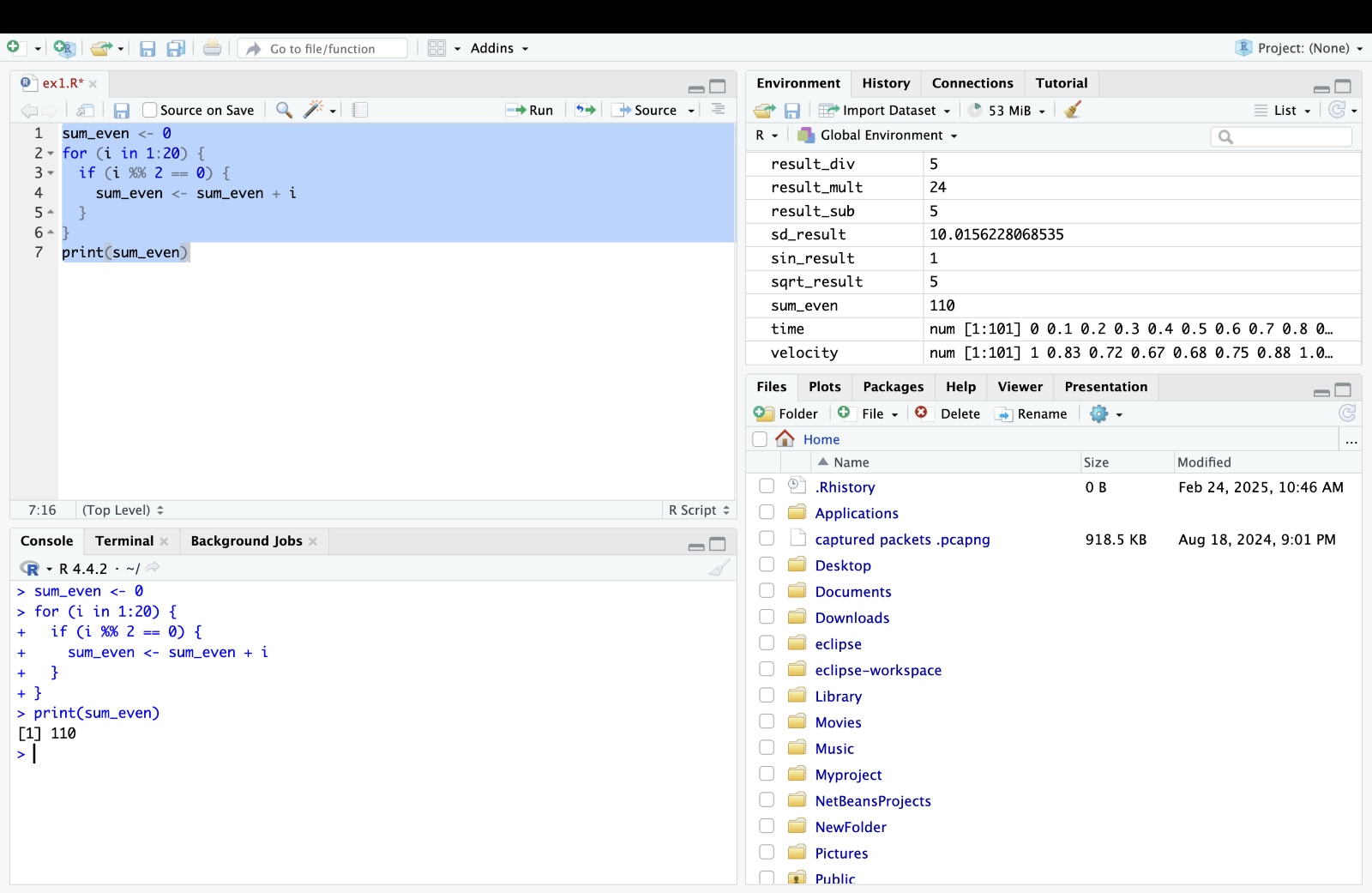
sum\_even <- 0 for (i in 1:20) {

if (i %% 2 == 0) { sum\_even <- sum\_even + i

} }

print(sum\_even)

# Output should be 110



1. Exercise 2: Prime Number Checker

o Task: Write a while loop to check if a given number is prime.

o Expected Output:

num <- 29

is\_prime <- TRUE

i <- 2

while (i <= sqrt(num)) {

if (num %% i == 0) {

is\_prime <- FALSE

break

}

i <- i + 1

}

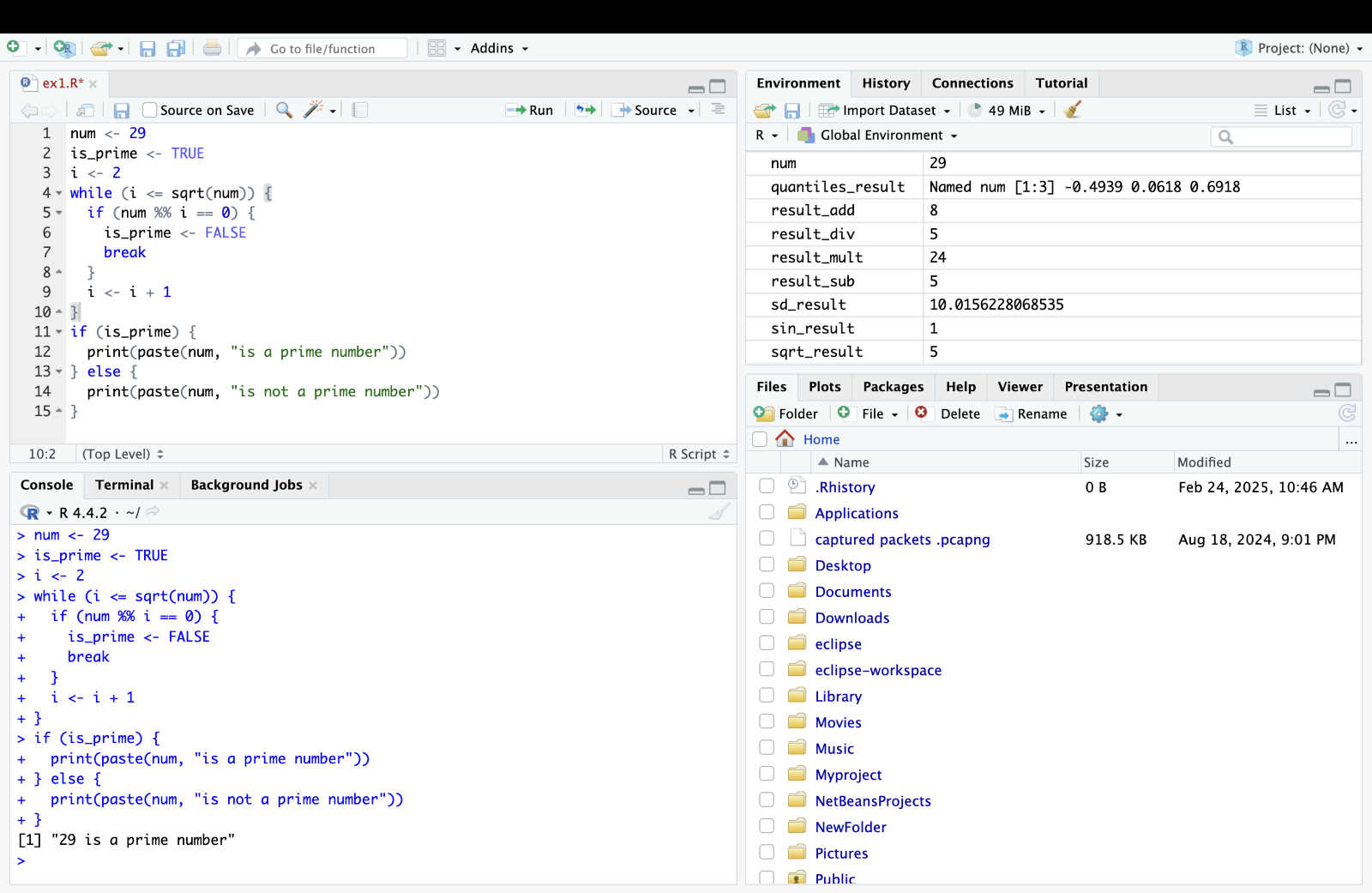
if (is\_prime) {

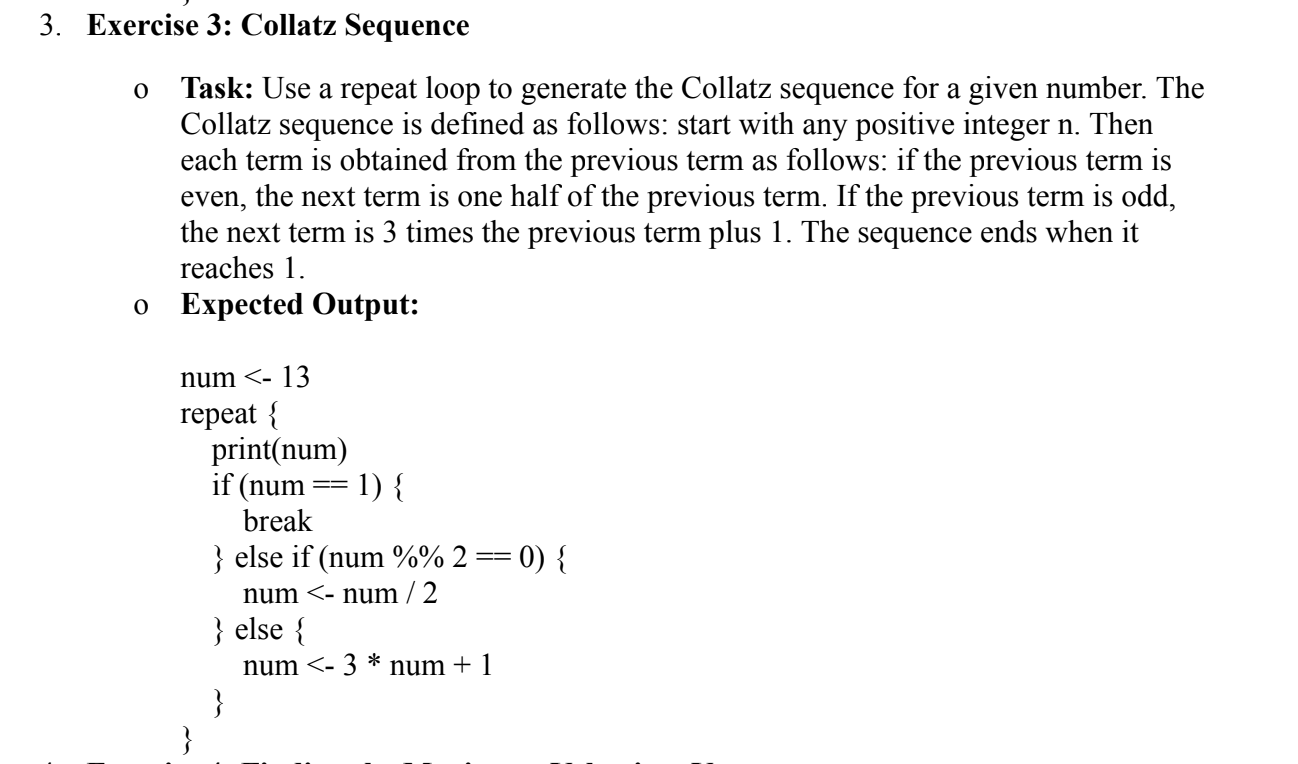
print(paste(num, "is a prime number"))

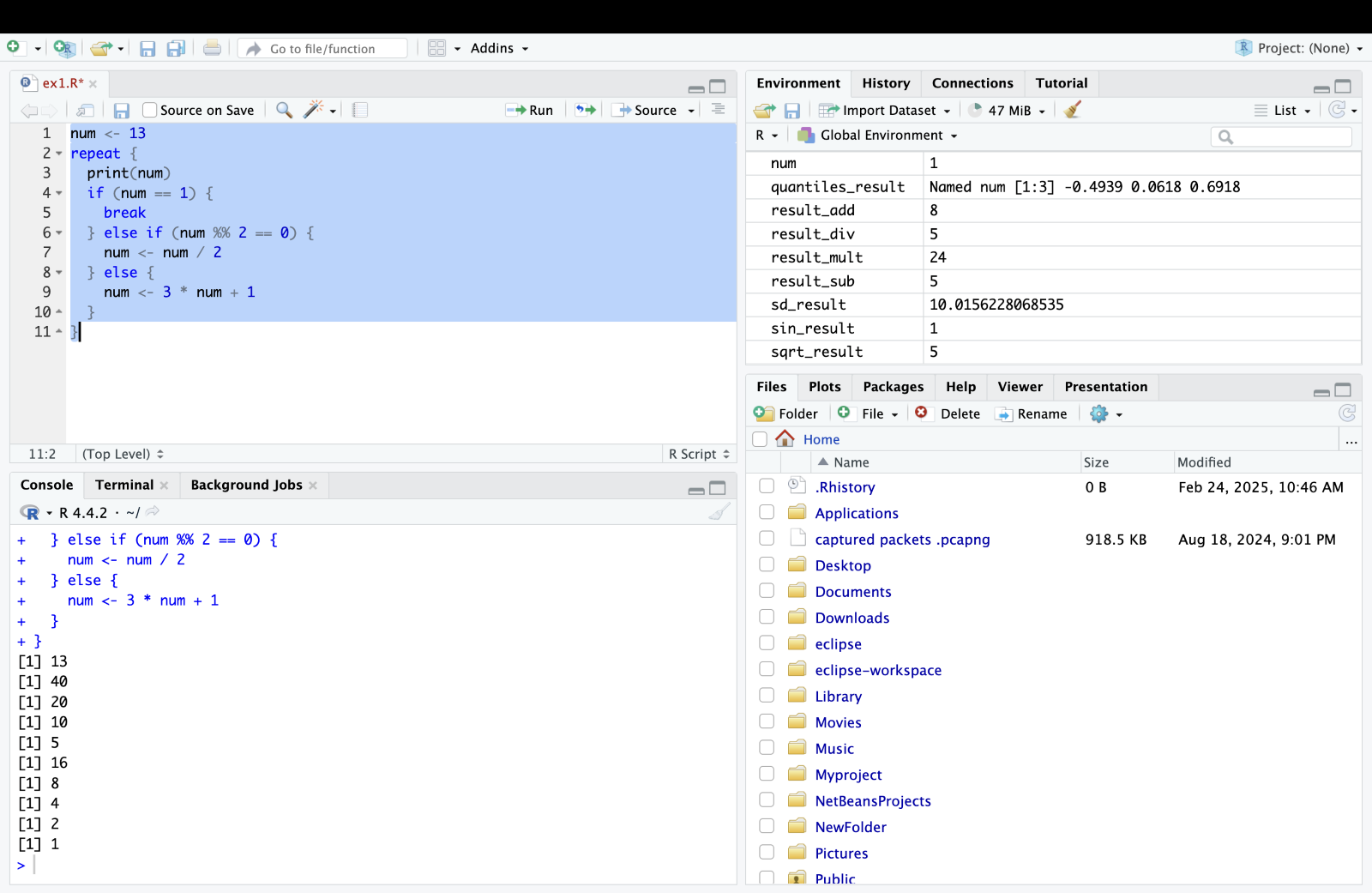
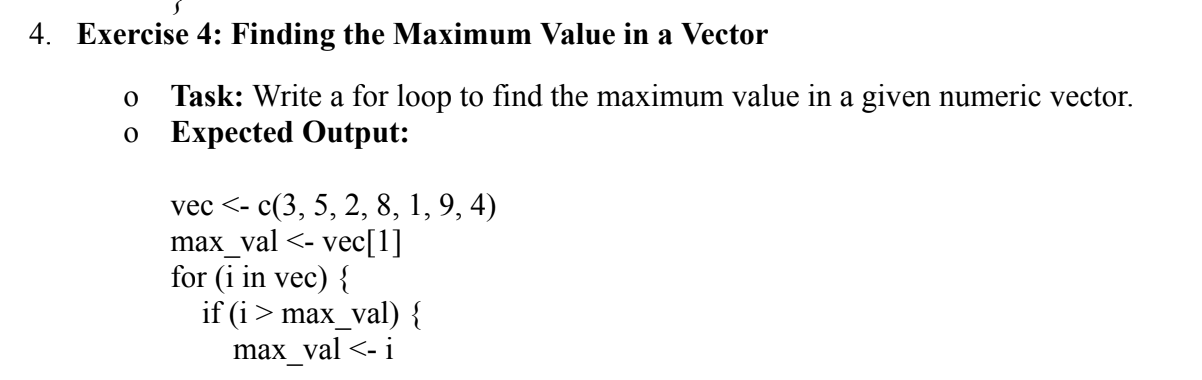
}

else {

print(paste(num, "is not a prime number")) }



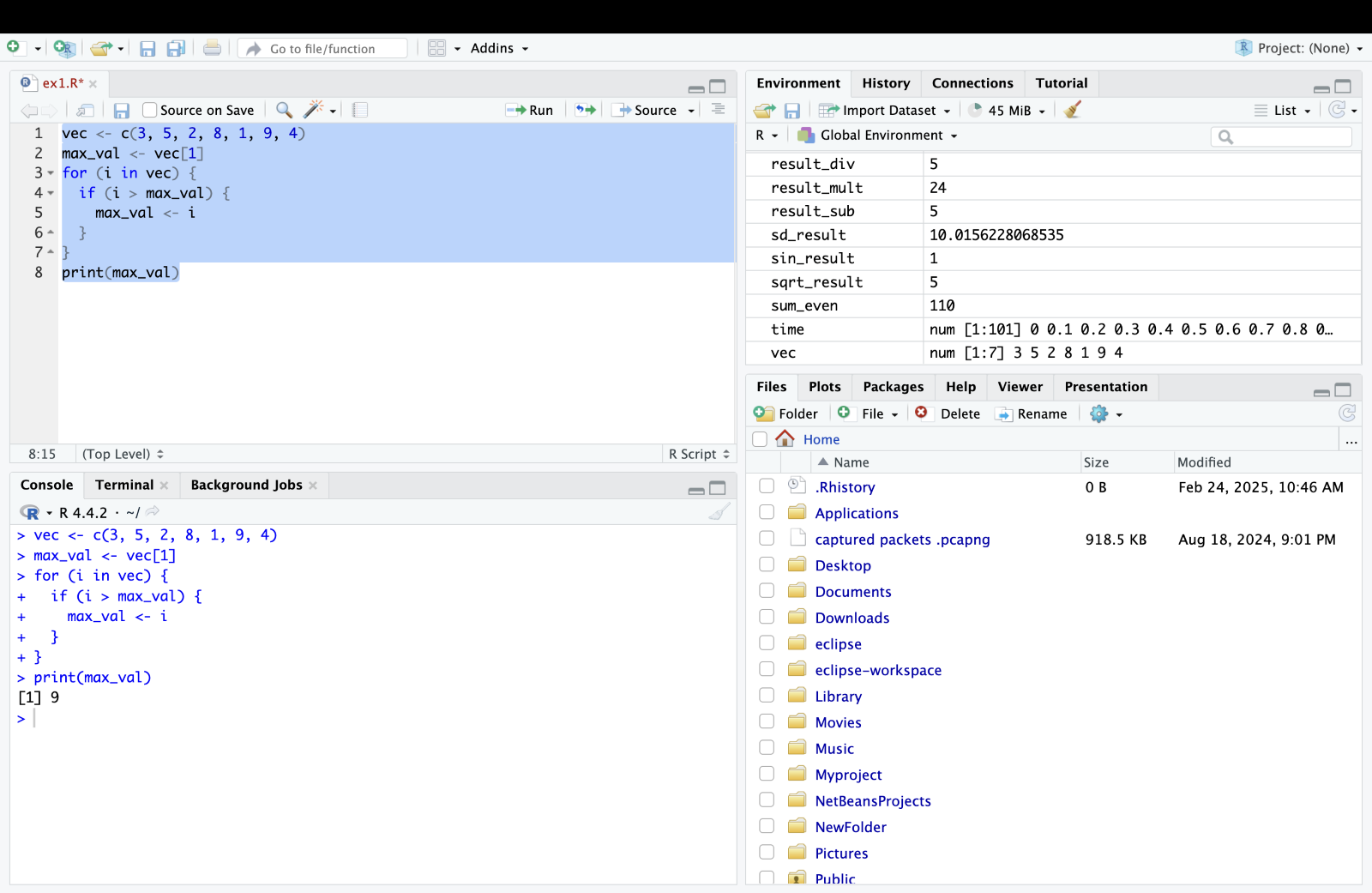


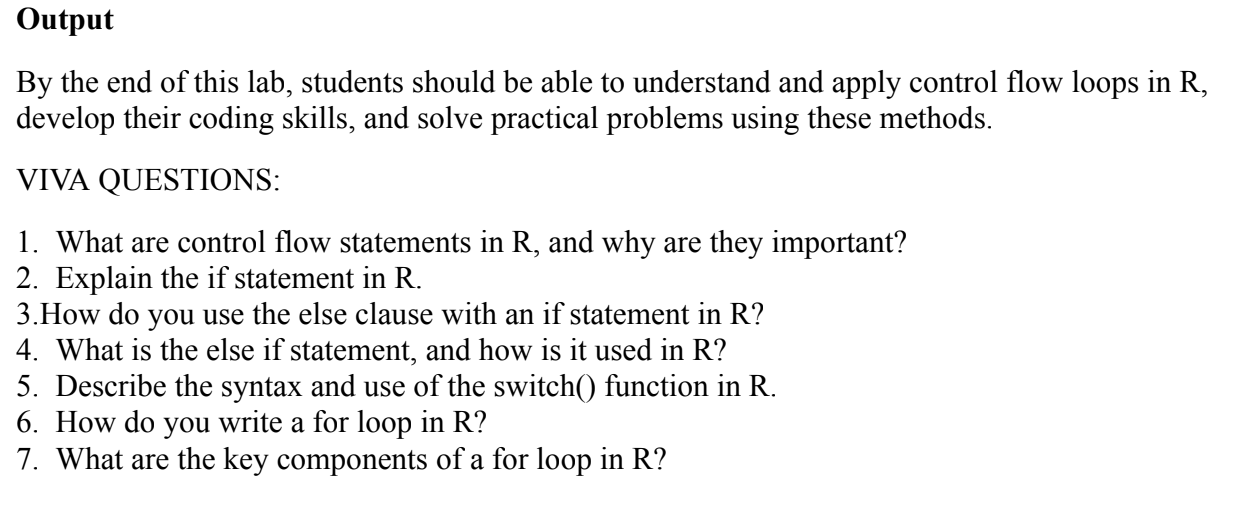
  
 }

}

print(max\_val)

# Output should be 9





**V. TESTING OF HYPOTHESIS – Z AND T TESTING**

**Aim**

The aim of this lab manual is to introduce students to the concept of hypothesis testing using the

Z-test in statistics. Students will learn how to perform one-sample and two-sample Z-tests,

Interpret results, and apply these skills using R.

Procedure

1. Introduction to Hypothesis Testing

* Explain the concept of hypothesis testing.
* Discuss the steps involved: formulation of null and alternative hypotheses,

Choosing a significance level, conducting the test, and interpreting the results.

2. Z-Test Overview

* Introduce the Z-test.
* Explain when to use the Z-test, typically when the sample size is large (n ≥ 30)

And / or the population standard deviation (σ) is known.

3. Setting Up R Environment

* Ensure R and RStudio are installed.
* Open RStudio and create a new R script for the lab exercises.

4. Coding Examples

* Example 1: One-Sample Z-Test

# Example data

sample\_data <- c(12, 15, 18, 14, 16, 19, 17, 13, 15, 18, 16, 15, 17, 16, 14)

# Population parameters (if known)

mu <- 16 # Population mean

sigma <- 2 # Population standard deviation

# Calculate Z-statistic

z\_stat <- (mean(sample\_data) - mu) / (sigma / sqrt(length(sample\_data)))

# Calculate p-value (two-tailed)

p\_value <- 2 \* (1 - pnorm(abs(z\_stat)))

# Print results

cat("Z-statistic:", z\_stat, "\n")

cat("P-value:", p\_value, "\n")

# Interpretation

if (p\_value < 0.05) {

cat("Reject null hypothesis: The sample mean is significantly different from",

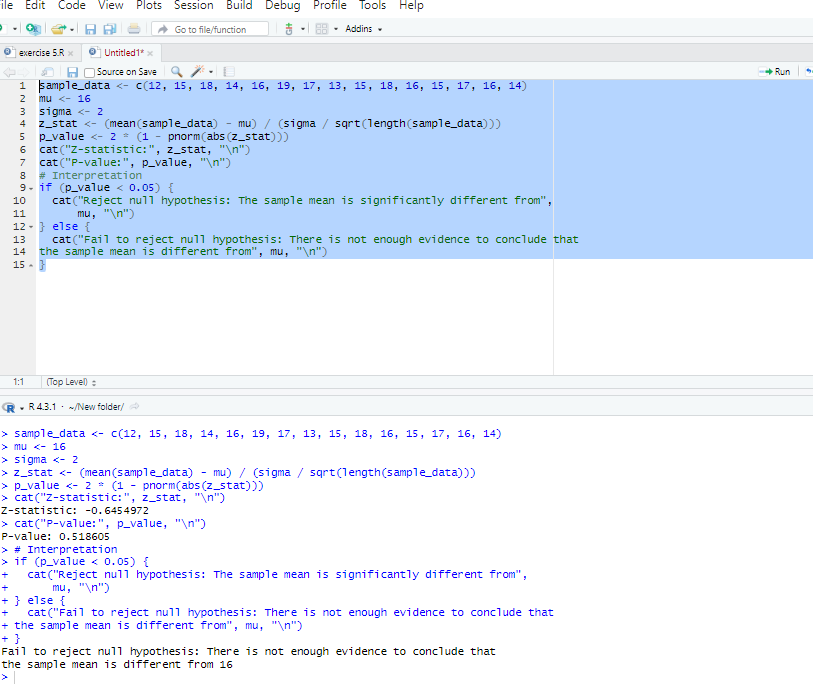
mu, "\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the sample mean is different from", mu, "\n")

}



* Example 2: Two-Sample Z-Test

group1 <- c(72, 75, 78, 71, 74, 77, 76, 73, 75, 78)

group2 <- c(68, 71, 73, 69, 72, 70, 72, 67, 71, 74)

# Calculate Z-statistic for two-sample test

z\_stat <- (mean(group1) - mean(group2)) / sqrt(var(group1)/length(group1) +

var(group2)/length(group2))

# Calculate p-value (two-tailed)

p\_value <- 2 \* (1 - pnorm(abs(z\_stat)))

# Print results

cat("Z-statistic:", z\_stat, "\n")

cat("P-value:", p\_value, "\n")

# Interpretation

if (p\_value < 0.05) {

cat("Reject null hypothesis: The means of two groups are significantly different

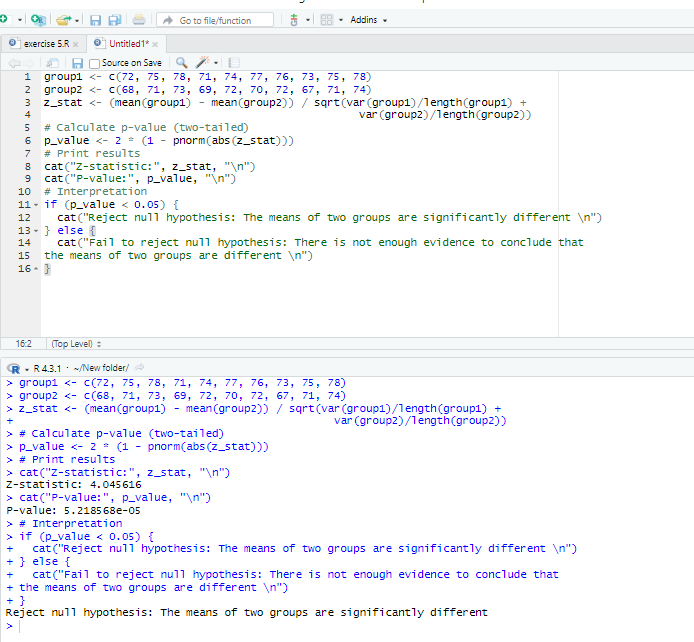
\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the means of two groups are different \n")

}



**Exercises**

1. Exercise 1: One-Sample Z-Test

* Task: Perform a one-sample Z-test to determine if the mean weight of a sample of

50 individuals is significantly different from 65 kg. Use a significance level of

0.05.

* **Expected Output:**

**# Example data**

sample\_data <- rnorm(50, mean = 68, sd = 5) # Simulated sample data

# Population parameters (if known)

mu <- 65 # Population mean

sigma <- 5 # Population standard deviation

# Calculate Z-statistic

z\_stat <- (mean(sample\_data) - mu) / (sigma / sqrt(length(sample\_data)))

# Calculate p-value (two-tailed)

p\_value <- 2 \* (1 - pnorm(abs(z\_stat)))

# Print results

cat("Z-statistic:", z\_stat, "\n")

cat("P-value:", p\_value, "\n")

# Interpretation

if (p\_value < 0.05) {

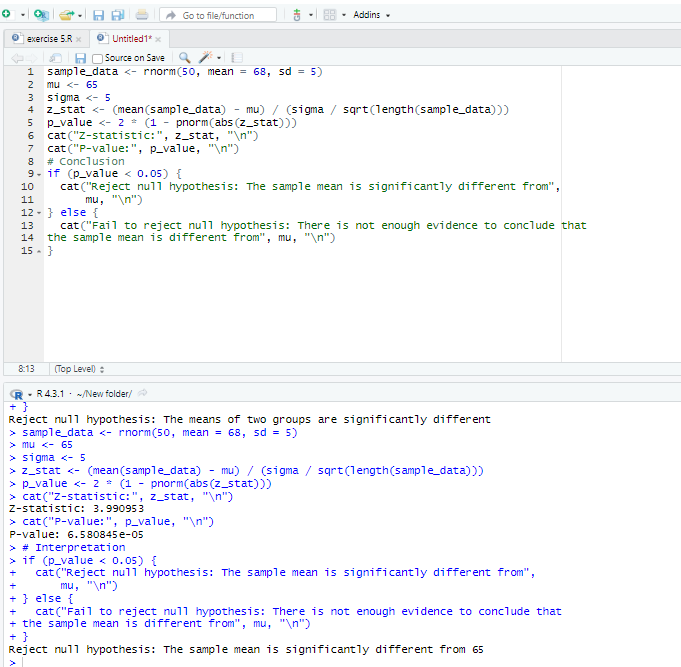
cat("Reject null hypothesis: The sample mean is significantly different from",

mu, "\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the sample mean is different from", mu, "\n")}



2. Exercise 2: Two-Sample Z-Test

* Task: Perform a two-sample Z-test to compare the mean scores of two groups:

Group A and Group B. Use the following data:

**# Example data**

groupA <- c(85, 89, 92, 78, 86, 88, 90, 82, 87, 84)

groupB <- c(80, 81, 85, 79, 83, 81, 84, 78, 82, 80)

# Calculate Z-statistic for two-sample test

z\_stat <- (mean(groupA) - mean(groupB)) / sqrt(var(groupA)/length(groupA) +

var(groupB)/length(groupB))

# Calculate p-value (two-tailed)

p\_value <- 2 \* (1 - pnorm(abs(z\_stat)))

# Print results

cat("Z-statistic:", z\_stat, "\n")

cat("P-value:", p\_value, "\n")

# Interpretation

if (p\_value < 0.05) {

cat("Reject null hypothesis: The means of two groups are significantly different

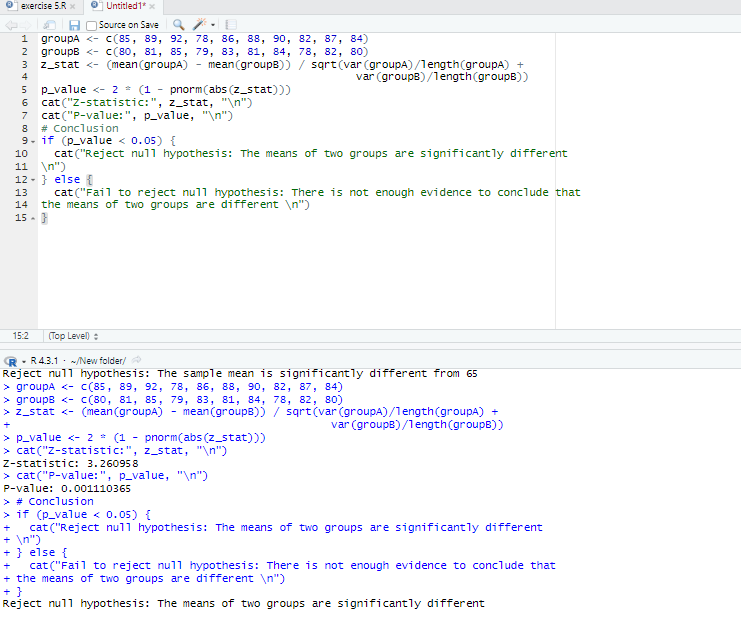
\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the means of two groups are different \n")

}



**T -Test Hypothesis Testing**

**Aim**

The aim of this lab manual is to introduce students to the concept of hypothesis testing using the

t-test in statistics. Students will learn how to perform one-sample and two-sample t-tests,

interpret results, and apply these skills using R.

Procedure

1. Introduction to Hypothesis Testing

* Explain the concept of hypothesis testing.
* Discuss the steps involved: formulation of null and alternative hypotheses,

choosing a significance level, conducting the test, and interpreting the results.

2. t-Test Overview

* Introduce the t-test.
* Explain when to use the t-test, typically when the sample size is small (n < 30) or

the population standard deviation (σ) is unknown.

3. Setting Up R Environment

* Ensure R and RStudio are installed.
* Open RStudio and create a new R script for the lab exercises.

4. Coding Examples

* Example 1: One-Sample t-Test

**# Example data**

sample\_data <- c(12, 15, 18, 14, 16, 19, 17, 13, 15, 18, 16, 15, 17, 16, 14)

# Population parameters (if known)

mu <- 16 # Population mean (null hypothesis)

# Conduct one-sample t-test

t\_test <- t.test(sample\_data, mu = mu)

# Print test result

print(t\_test)

# Interpretation

if (t\_test$p.value < 0.05) {

cat("Reject null hypothesis: The sample mean is significantly different from",

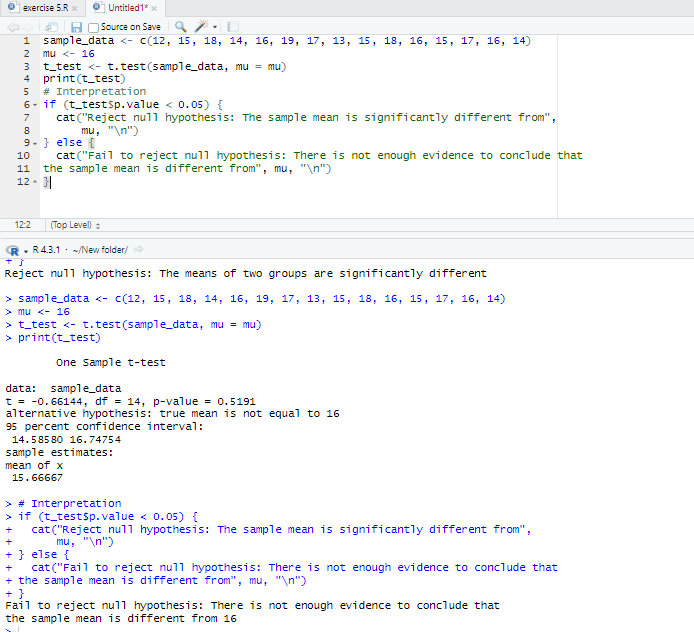
mu, "\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the sample mean is different from", mu, "\n")

}



* **Example 2: Two-Sample t-Test**

**# Example data**

group1 <- c(72, 75, 78, 71, 74, 77, 76, 73, 75, 78)

group2 <- c(68, 71, 73, 69, 72, 70, 72, 67, 71, 74)

# Conduct two-sample t-test

t\_test <- t.test(group1, group2)

# Print test result

print(t\_test)

# Interpretation

if (t\_test$p.value < 0.05) {

cat("Reject null hypothesis: The means of two groups are significantly different

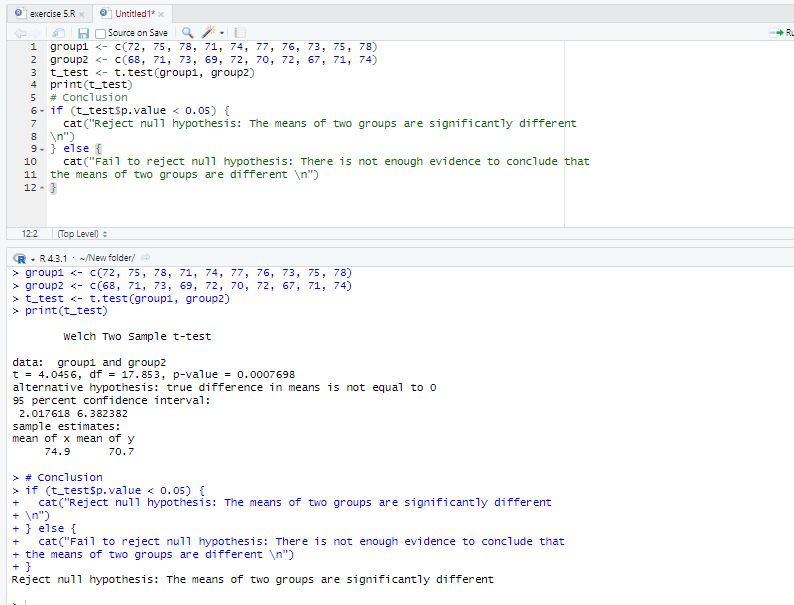
\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the means of two groups are different \n")

}



**Exercises**

**1. Exercise 1: One-Sample t-Test**

* Task: Perform a one-sample t-test to determine if the mean weight of a sample of

50 individuals is significantly different from 65 kg. Use a significance level of

0.05.

* **Expected Output:**

**# Example data**

sample\_data <- rnorm(50, mean = 68, sd = 5) # Simulated sample data

# Population parameters (if known)

mu <- 65 # Population mean (null hypothesis)

# Conduct one-sample t-test

t\_test <- t.test(sample\_data, mu = mu)

# Print test result

print(t\_test)

# Interpretation

if (t\_test$p.value < 0.05) {

cat("Reject null hypothesis: The sample mean is significantly different from",

mu, "\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the sample mean is different from", mu, "\n")

}



Exercise 2: Two-Sample t-Test

* Task: Perform a two-sample t-test to compare the mean scores of two groups:

Group A and Group B. Use the following data:

**Example data**

groupA <- c(85, 89, 92, 78, 86, 88, 90, 82, 87, 84)

groupB <- c(80, 81, 85, 79, 83, 81, 84, 78, 82, 80)

# Conduct two-sample t-test

t\_test <- t.test(groupA, groupB)

# Print test result

print(t\_test)

# Interpretation

if (t\_test$p.value < 0.05) {

cat("Reject null hypothesis: The means of two groups are significantly different

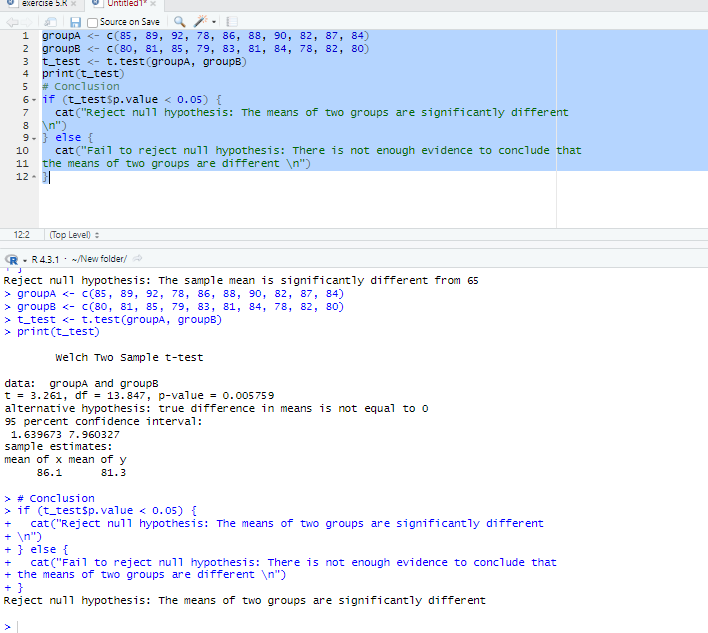
\n")

} else {

cat("Fail to reject null hypothesis: There is not enough evidence to conclude that

the means of two groups are different \n")

}



**VIVA QUESTIONS:**

1. What is a probability distribution, and why is it important in statistics?

2. Explain the concept of the probability density function (PDF).

3. What is the cumulative distribution function (CDF)?

4 How do you generate random samples from a Binomial distribution in R?

5. What functions are used to calculate the PDF and CDF of a Binomial distribution in R?

6. Describe how to plot the PDF and CDF of a Binomial distribution in R.

7. How do you generate random samples from a Poisson distribution in R?

8. What functions are used to calculate the PDF and CDF of a Poisson distribution in R?