**1. Factorial using a for loop**

factorial\_for <- function(n) {

if (n < 0) {

return("Error: Factorial is not defined for negative numbers.")

} else if (n == 0) {

return(1)

} else {

fact <- 1

for (i in 1:n) {

fact <- fact \* i

}

return(fact)

}

}

print(factorial\_for(5))

print(factorial\_for(0))

print(factorial\_for(-3))

**Output:**

[1] 120

[1] 1

[1] "Error: Factorial is not defined for negative numbers."

**2. Fibonacci using a while loop**

fibonacci\_while <- function(n) {

a <- 0

b <- 1

fib\_seq <- c(a, b)

while (b < n) {

temp <- a + b

a <- b

b <- temp

fib\_seq <- c(fib\_seq, b)

}

print(fib\_seq)

print(paste("Length:", length(fib\_seq)))

}

fibonacci\_while(10)

**Output:**

[1] 0 1 1 2 3 5 8 13

[1] "Length: 8"

**3. Grading system**

grade <- function(score) {

if (score >= 90) {

return("A")

} else if (score >= 80) {

return("B")

} else if (score >= 70) {

return("C")

} else if (score >= 60) {

return("D")

} else {

return("F")

}

}

print(grade(85))

print(grade(45))

**Output:**

[1] "B"

[1] "F"

**4. Mean of numeric vectors ignoring non-numeric values**

list\_of\_vectors <- list(c(1, 2, 3), c(4, 5, "a"), c(10, 20, 30))

for (vec in list\_of\_vectors) {

numeric\_values <- as.numeric(vec[!is.na(as.numeric(vec))])

print(mean(numeric\_values))

}

**Output:**

[1] 2

[1] 4.5

[1] 20

**5. Filter rows in a data frame**

df <- data.frame(Name = c("Alice", "Bob", "Charlie"),

Age = c(25, 35, 40))

filtered\_rows <- df[df$Age > 30, ]

print(filtered\_rows)

**Output:**

Name Age

2 Bob 35

3 Charlie 40

**6. Arithmetic operations**

arithmetic <- function(a, b) {

cat("Addition:", a + b, "\n")

cat("Subtraction:", a - b, "\n")

cat("Multiplication:", a \* b, "\n")

cat("Division:", a / b, "\n")

}

# Example

arithmetic(10, 2)

**Output:**

Addition: 12 Multiplication: 20

Division: 5

**7. Attendance or exam pass**

students <- data.frame(Name = c("Alice", "Bob", "Charlie"),

Attendance = c(TRUE, FALSE, TRUE),

PassedExam = c(FALSE, TRUE, TRUE))

qualified <- students[students$Attendance | students$PassedExam, ]

print(qualified)

**Output:**

Name Attendance PassedExam

1 Alice TRUE FALSE

2 Bob FALSE TRUE

3 Charlie TRUE TRUE

**8. Mean, median, and mode**

stats\_function <- function(vec) {

mode\_value <- names(sort(table(vec), decreasing = TRUE))[1]

return(list(mean = mean(vec), median = median(vec), mode = mode\_value))

}

print(stats\_function(c(1, 2, 2, 3, 4)))

**Output:**

$mean

[1] 2.4

$median

[1] 2

$mode

[1] "2"

**9. Recursive factorial**

factorial\_recursive <- function(n) {

if (n == 0) return(1)

else return(n \* factorial\_recursive(n - 1))

}

print(factorial\_recursive(5))

**Output:**

csharp

CopyEdit

[1] 120

**10. Recursive Fibonacci**

fibonacci\_recursive <- function(n) {

if (n <= 1) return(n)

else return(fibonacci\_recursive(n - 1) + fibonacci\_recursive(n - 2))

}

print(fibonacci\_recursive(6))

**Output:**

[1] 8

**11. Vector iteration with a function**

operation <- function(x = 2) {

return(x \* x)

}

vec <- c(1, 2, 3, 4, 5)

for (num in vec) {

if (num %% 2 == 0) {

print(operation(num))

}

}

**Output:**

[1] 4

[1] 16

**12. Rectangle area function**

rectangle\_area <- function(length = 5, width = 3) {

return(length \* width)

}

print(rectangle\_area())

print(rectangle\_area(10, 2))

**Output:**

[1] 15

[1] 20

**13. Prime number check**

is\_prime <- function(n) {

if (n < 2) return(FALSE)

for (i in 2:sqrt(n)) {

if (n %% i == 0) return(FALSE)

}

return(TRUE)

}

print(is\_prime(7))

print(is\_prime(10))

**Output:**

csharp

CopyEdit

[1] TRUE

[1] FALSE

**14. Recursive sum**

recursive\_sum <- function(vec) {

if (length(vec) == 0) return(0)

else return(vec[1] + recursive\_sum(vec[-1]))

}

print(recursive\_sum(c(1, 2, 3, 4, 5)))

**Output:**

[1] 15

**15. Another grading system**

(Same as 3)

**16. Replace elements in a vector**

replace\_signs <- function(vec) {

return(ifelse(vec > 0, "positive", ifelse(vec < 0, "negative", "zero")))

}

print(replace\_signs(c(-5, 0, 10)))

**Output:**

[1] "negative" "zero" "positive"

**17. Loop over categories**

categories <- list(Fruits = c("Apple", "Banana"), Electronics = c("TV", "Phone"))

for (cat in names(categories)) {

print(paste(cat, ":", length(categories[[cat]])))

}

**Output:**

[1] "Fruits : 2"

[1] "Electronics : 2"

**18. Identify duplicated product combinations and unique customer-product pairs**

df <- data.frame(Customer = c("A", "B", "A", "C", "B", "A"),

Product = c("X", "Y", "X", "Z", "Y", "X"))

duplicated\_rows <- df[duplicated(df), ]

unique\_pairs <- unique(df)

print("Duplicated Product Combinations:")

print(duplicated\_rows)

print("Unique Customer-Product Pairs:")

print(unique\_pairs)

**Output:**

[1] "Duplicated Product Combinations:"

Customer Product

3 A X

5 B Y

[1] "Unique Customer-Product Pairs:"

Customer Product

1 A X

2 B Y

4 C Z

**19. Identify duplicated treatments and unique patient-treatment combinations**

df <- data.frame(PatientID = c(101, 102, 101, 103, 102, 104),

Treatment = c("Aspirin", "Ibuprofen", "Aspirin", "Antibiotic", "Ibuprofen", "Vitamin C"))

duplicated\_treatments <- df[duplicated(df), ]

unique\_treatments <- unique(df)

print("Duplicated Treatments:")

print(duplicated\_treatments)

print("Unique Patient-Treatment Combinations:")

print(unique\_treatments)

**Output:**

[1] "Duplicated Treatments:"

PatientID Treatment

3 101 Aspirin

5 102 Ibuprofen

[1] "Unique Patient-Treatment Combinations:"

PatientID Treatment

1 101 Aspirin

2 102 Ibuprofen

4 103 Antibiotic

6 104 Vitamin C

**20. Find and list duplicated treatments for patients, and ensure unique combinations**

df <- data.frame(PatientID = c(201, 202, 201, 203, 204, 202, 205, 201),

Treatment = c("Insulin", "Antibiotic", "Insulin", "Painkiller", "Vitamin D", "Antibiotic", "Painkiller", "Insulin"))

duplicated\_rows <- df[duplicated(df), ]

unique\_combinations <- unique(df)

print("Duplicated Patient-Treatment Combinations:")

print(duplicated\_rows)

print("Table of Unique Patient-Treatment Combinations:")

print(unique\_combinations)

**Output:**

[1] "Duplicated Patient-Treatment Combinations:"

PatientID Treatment

3 201 Insulin

6 202 Antibiotic

[1] "Table of Unique Patient-Treatment Combinations:"

PatientID Treatment

1 201 Insulin

2 202 Antibiotic

4 203 Painkiller

5 204 Vitamin D

7 205 Painkiller