**SET1**

**Code1**

kelvin\_to\_celsius <- function(kelvin\_temp) {

celsius\_temp <- kelvin\_temp - 273.15

return(celsius\_temp)

}

values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

mean\_value <- mean(values)

median\_value <- median(values)

mode\_value <- names(table(values))[table(values) == max(table(values))]

print(mean\_value)

print(median\_value)

print(mode\_value)

values <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

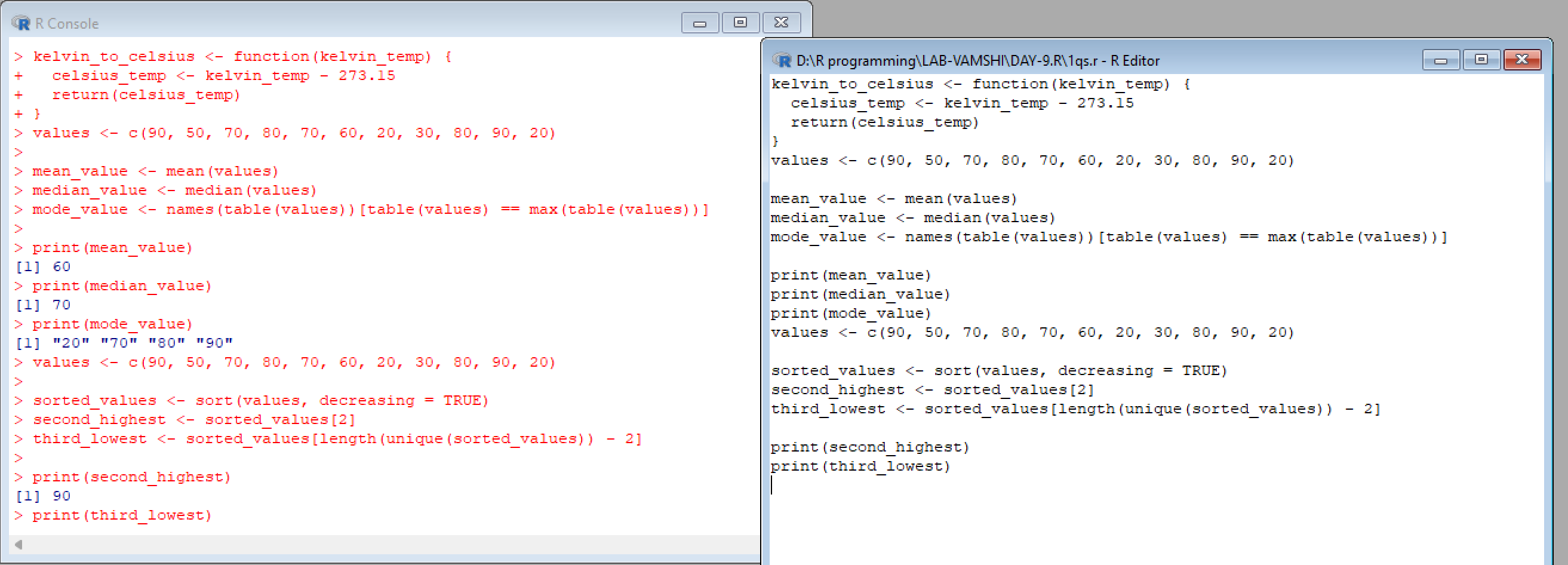
sorted\_values <- sort(values, decreasing = TRUE)

second\_highest <- sorted\_values[2]

third\_lowest <- sorted\_values[length(unique(sorted\_values)) - 2]

print(second\_highest)

print(third\_lowest)



**Code2**

# Load the airquality dataset

data(airquality)

# i. Compute the mean temperature (without using built-in function)

mean\_temperature <- sum(airquality$Temp) / length(airquality$Temp)

print(mean\_temperature)

# ii. Extract the first five rows from airquality

first\_five\_rows <- airquality[1:5, ]

print(first\_five\_rows)

# iii. Extract all columns from airquality except Temp and Wind

subset\_airquality <- airquality[, !(names(airquality) %in% c("Temp", "Wind"))]

print(subset\_airquality)

# iv. Find the coldest day during the period

coldest\_day <- airquality$Day[which.min(airquality$Temp)]

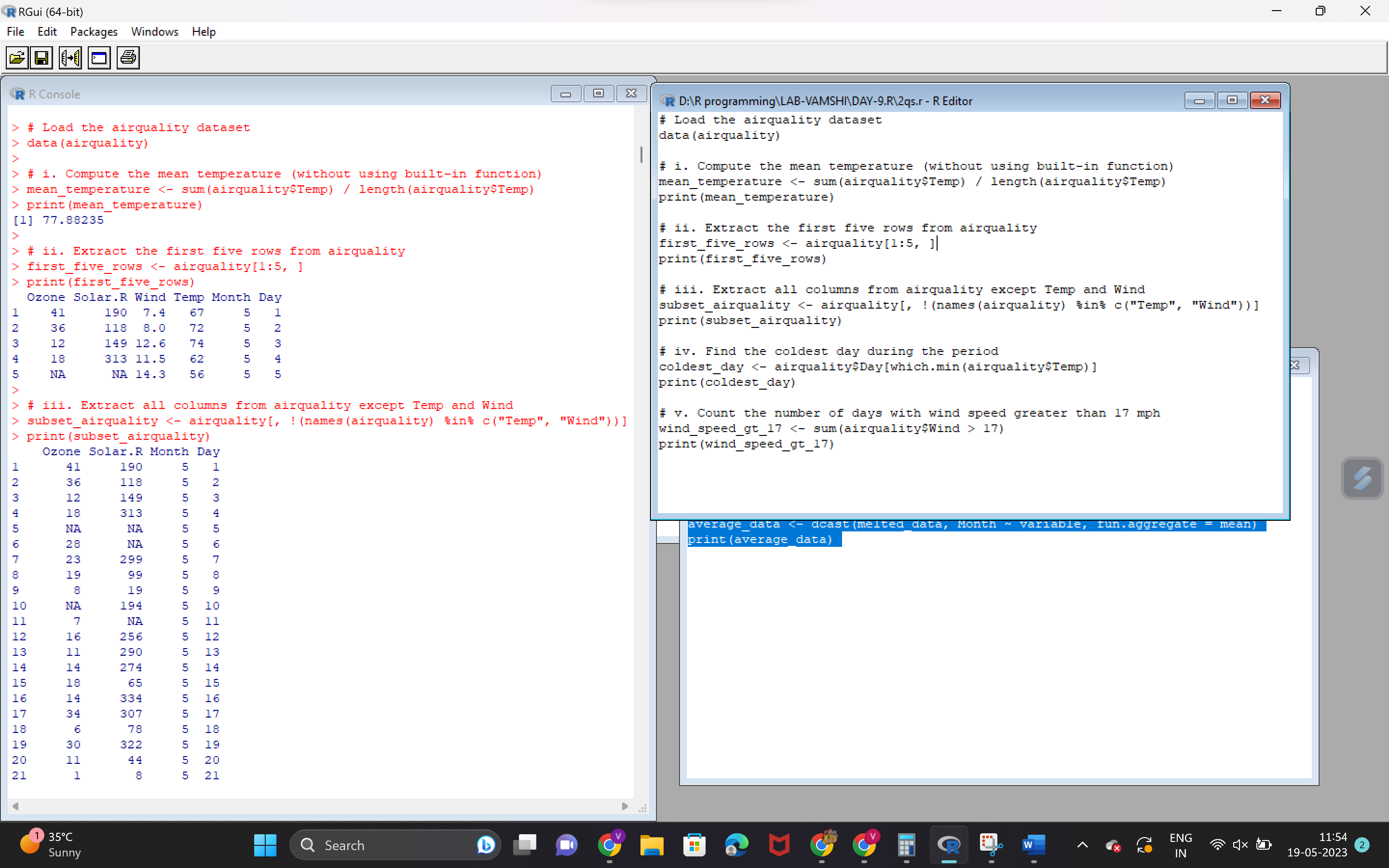
print(coldest\_day)

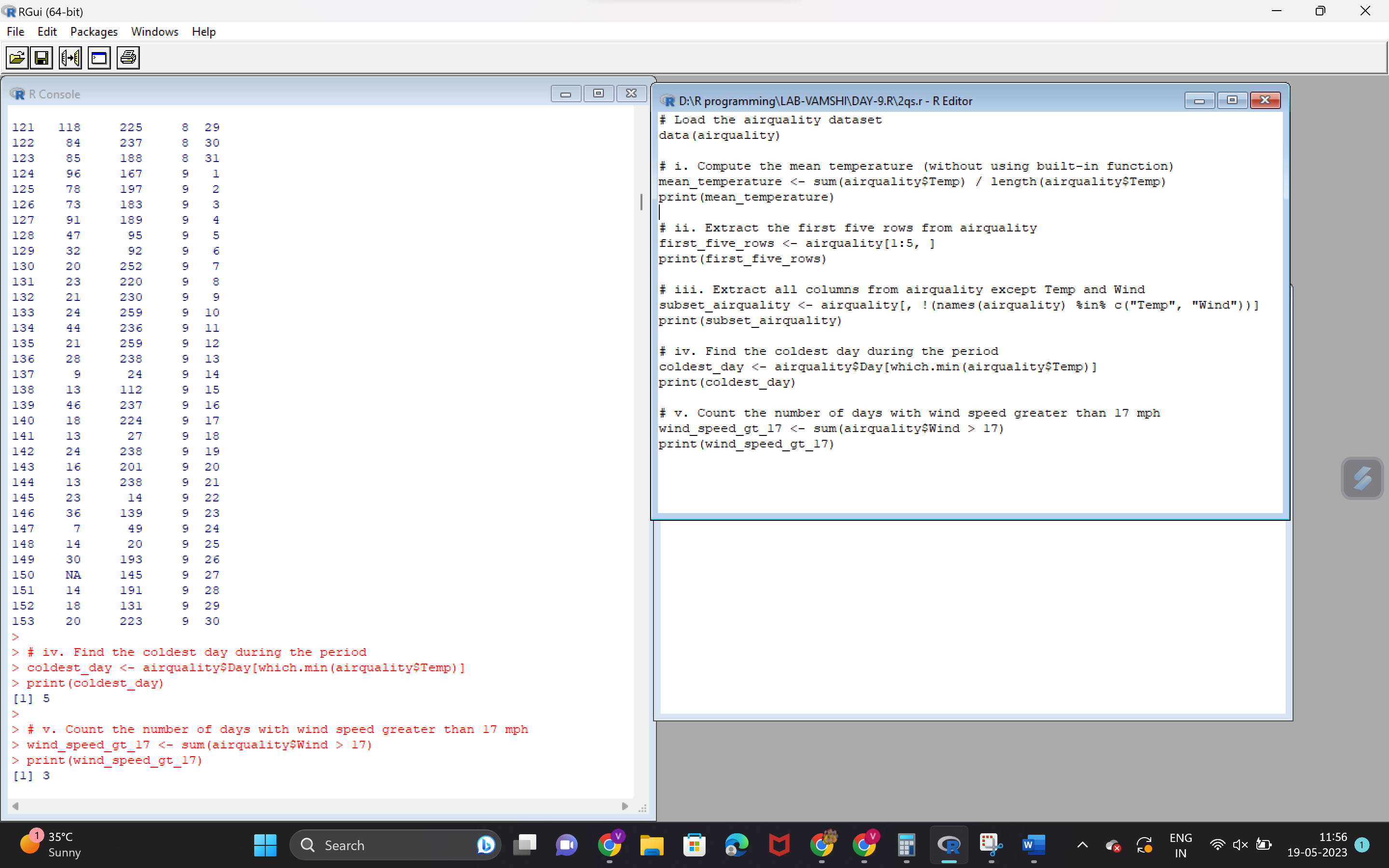
# v. Count the number of days with wind speed greater than 17 mph

wind\_speed\_gt\_17 <- sum(airquality$Wind > 17)

print(wind\_speed\_gt\_17)

**output**





**Code3**

summary(airquality)

library(reshape2)

melted\_data <- melt(airquality)

print(melted\_data)

melted\_data <- melt(airquality, id.vars = c("Month", "Day"))

print(melted\_data)

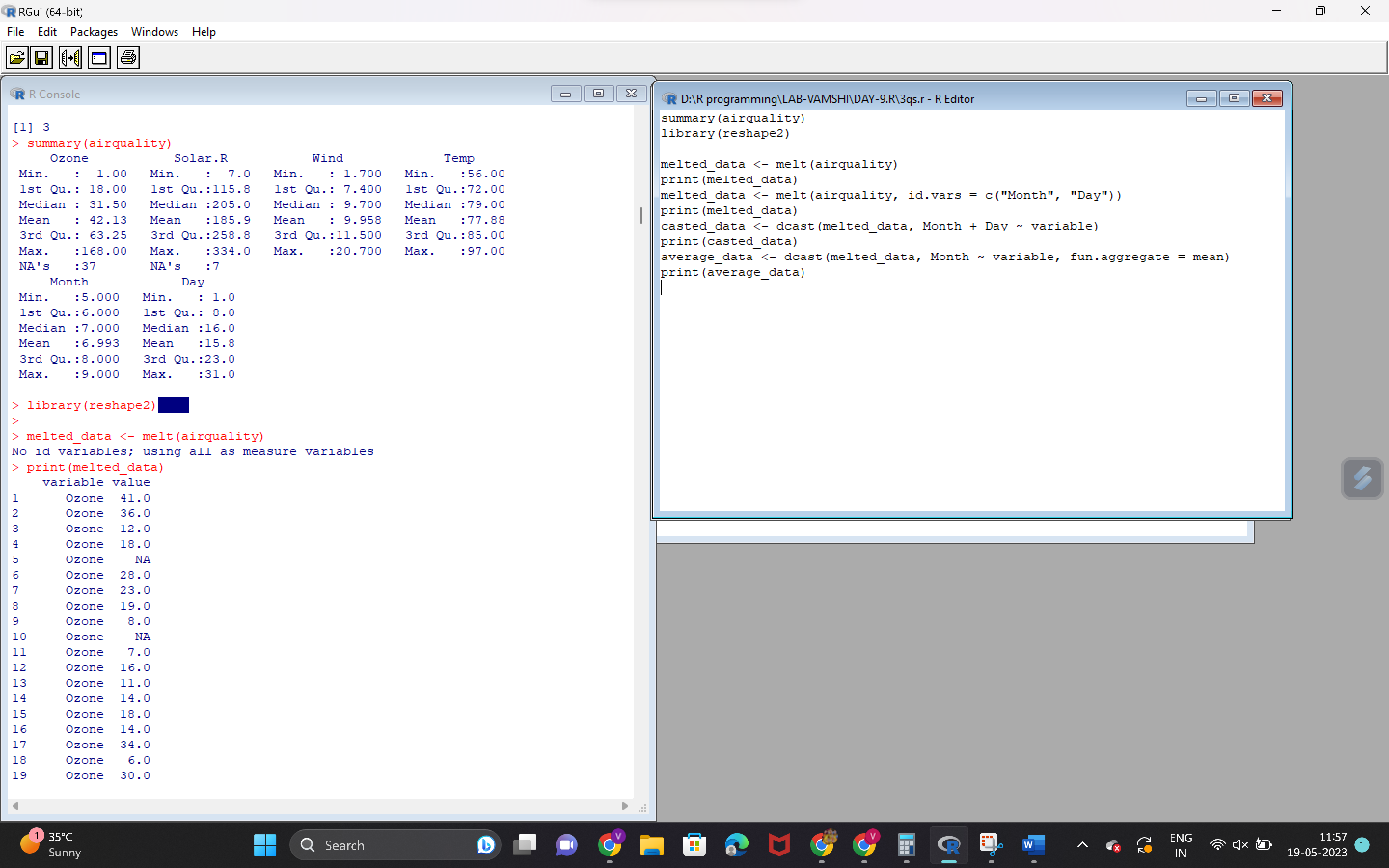
casted\_data <- dcast(melted\_data, Month + Day ~ variable)

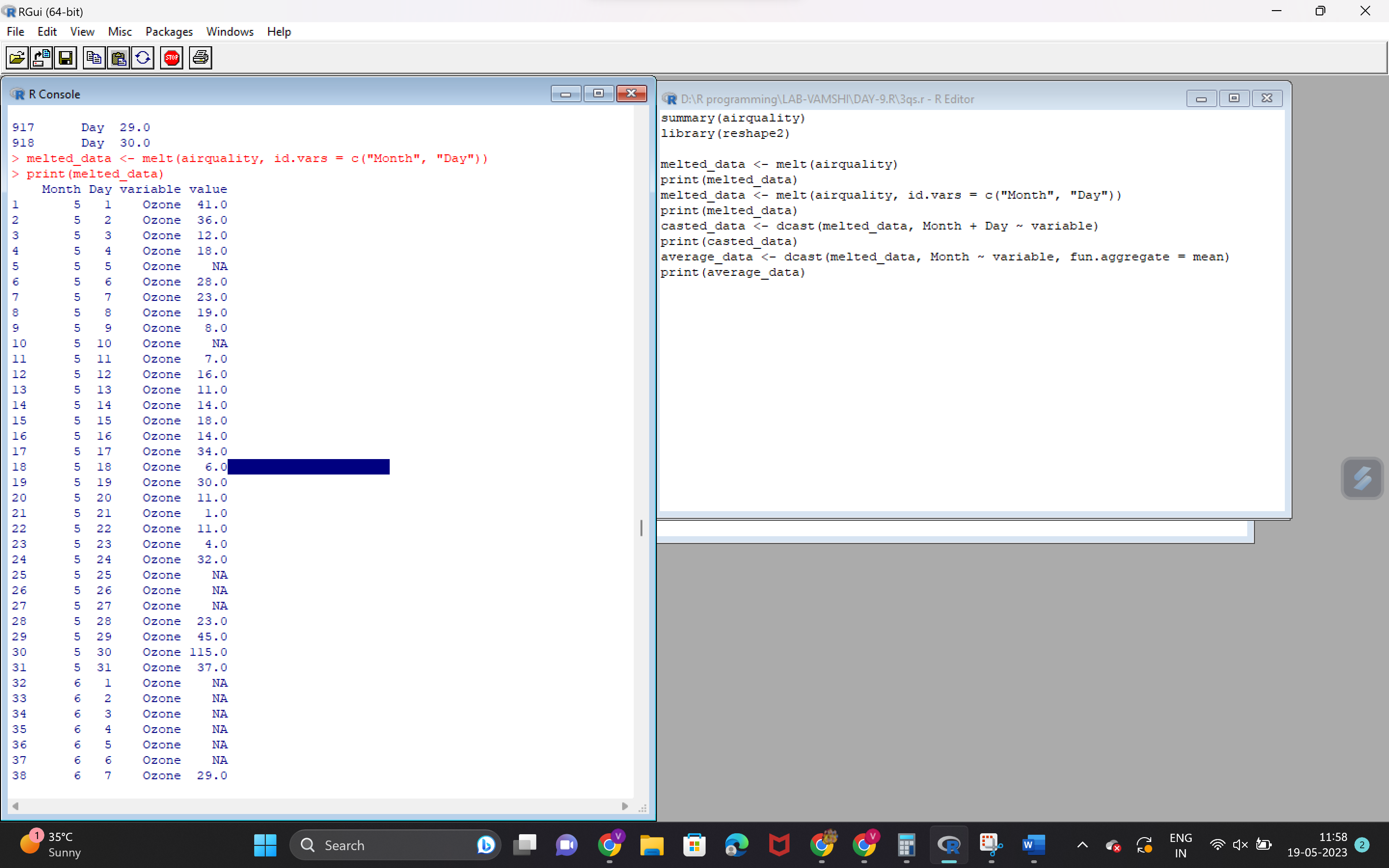
print(casted\_data)

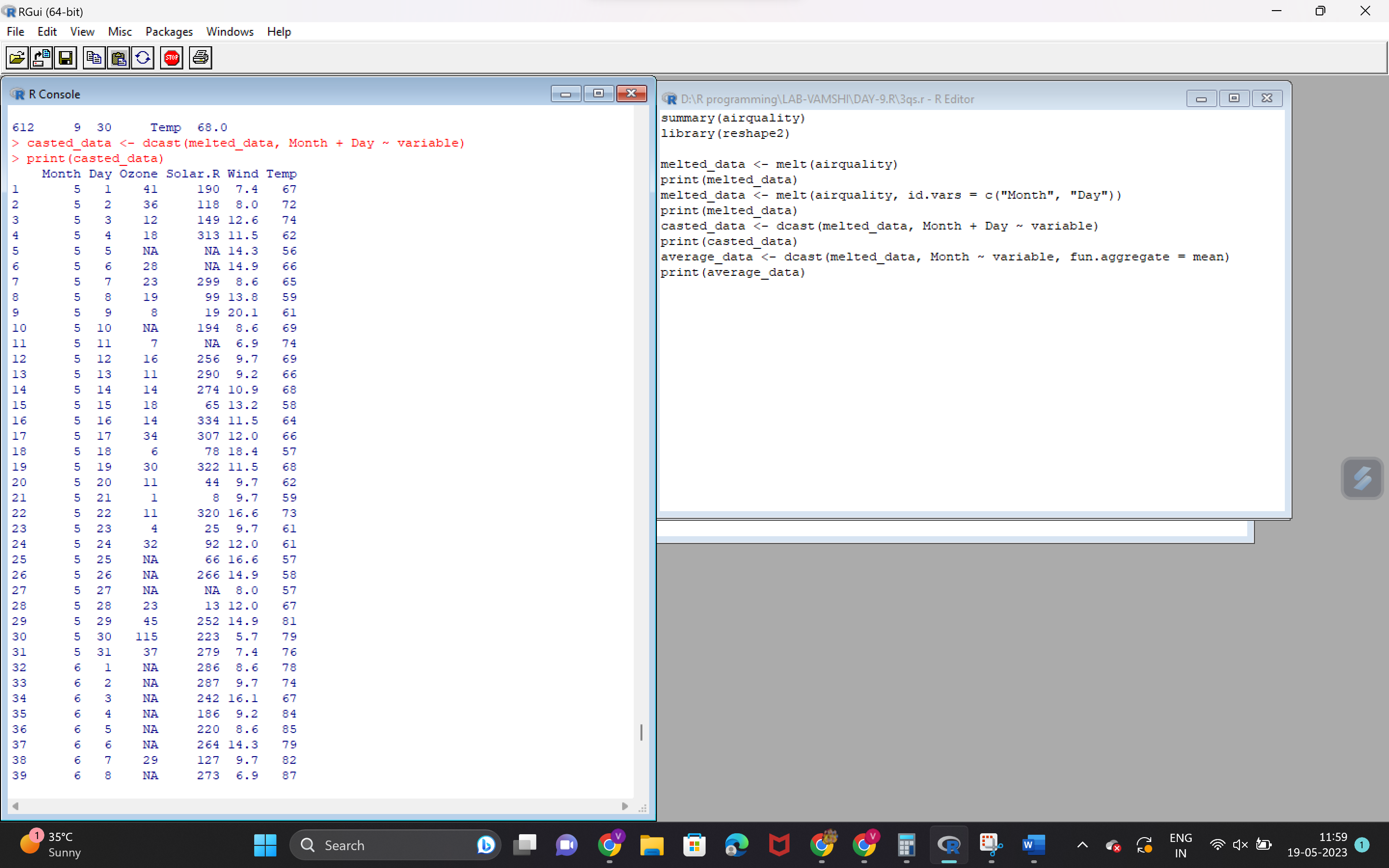
average\_data <- dcast(melted\_data, Month ~ variable, fun.aggregate = mean)

print(average\_data)

**output**







**Code4**

# Check for missing values in the dataset

missing\_values <- sum(is.na(airquality))

missing\_percentage <- missing\_values / nrow(airquality) \* 100

# If missing values are less than 10%, drop them. Otherwise, replace with mean.

if (missing\_percentage < 10) {

airquality <- na.omit(airquality)

} else {

airquality[is.na(airquality)] <- mean(airquality, na.rm = TRUE)

}

# Fit linear regression model

model <- lm(Ozone ~ Solar.R, data = airquality)

# Print model summary

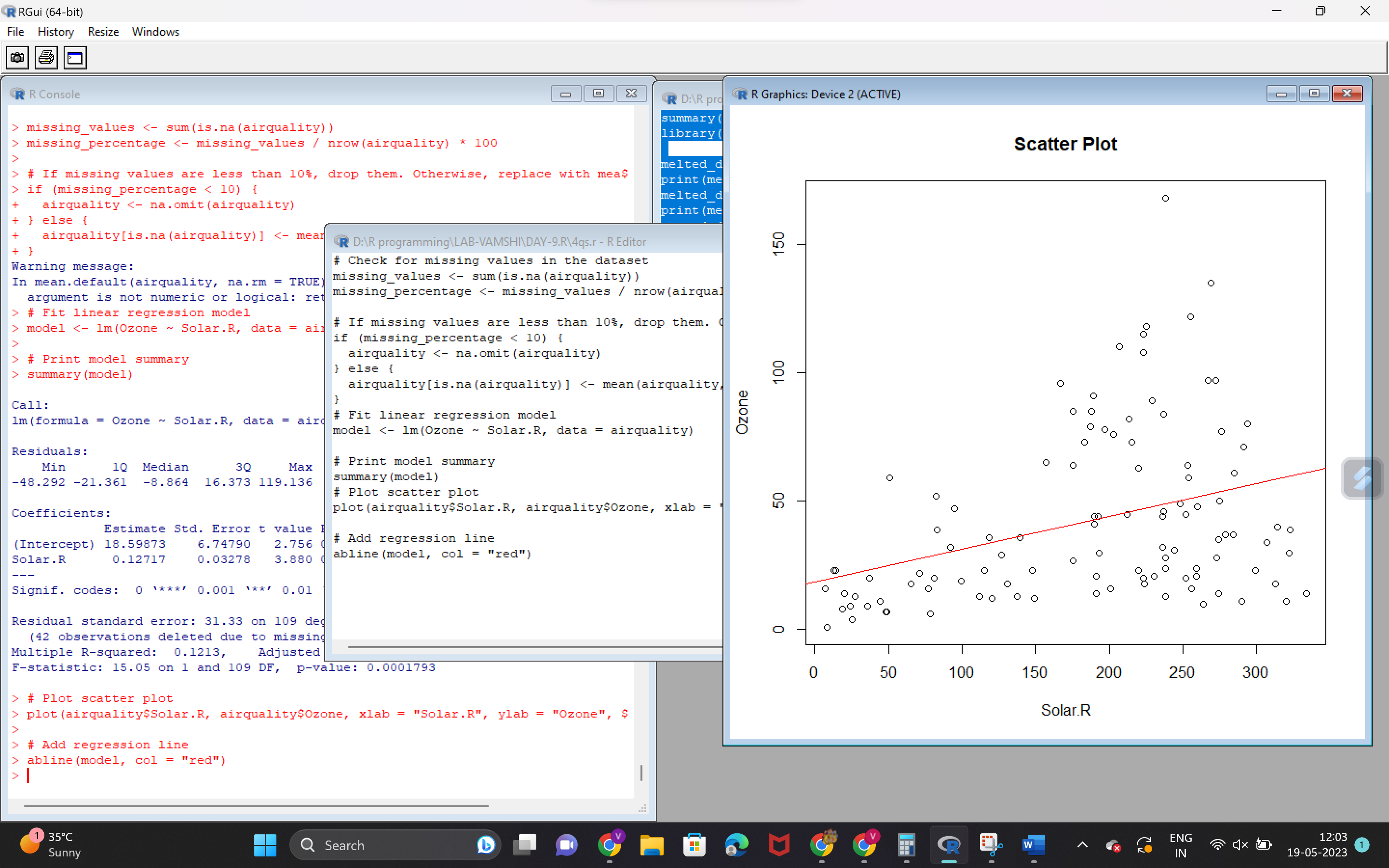
summary(model)

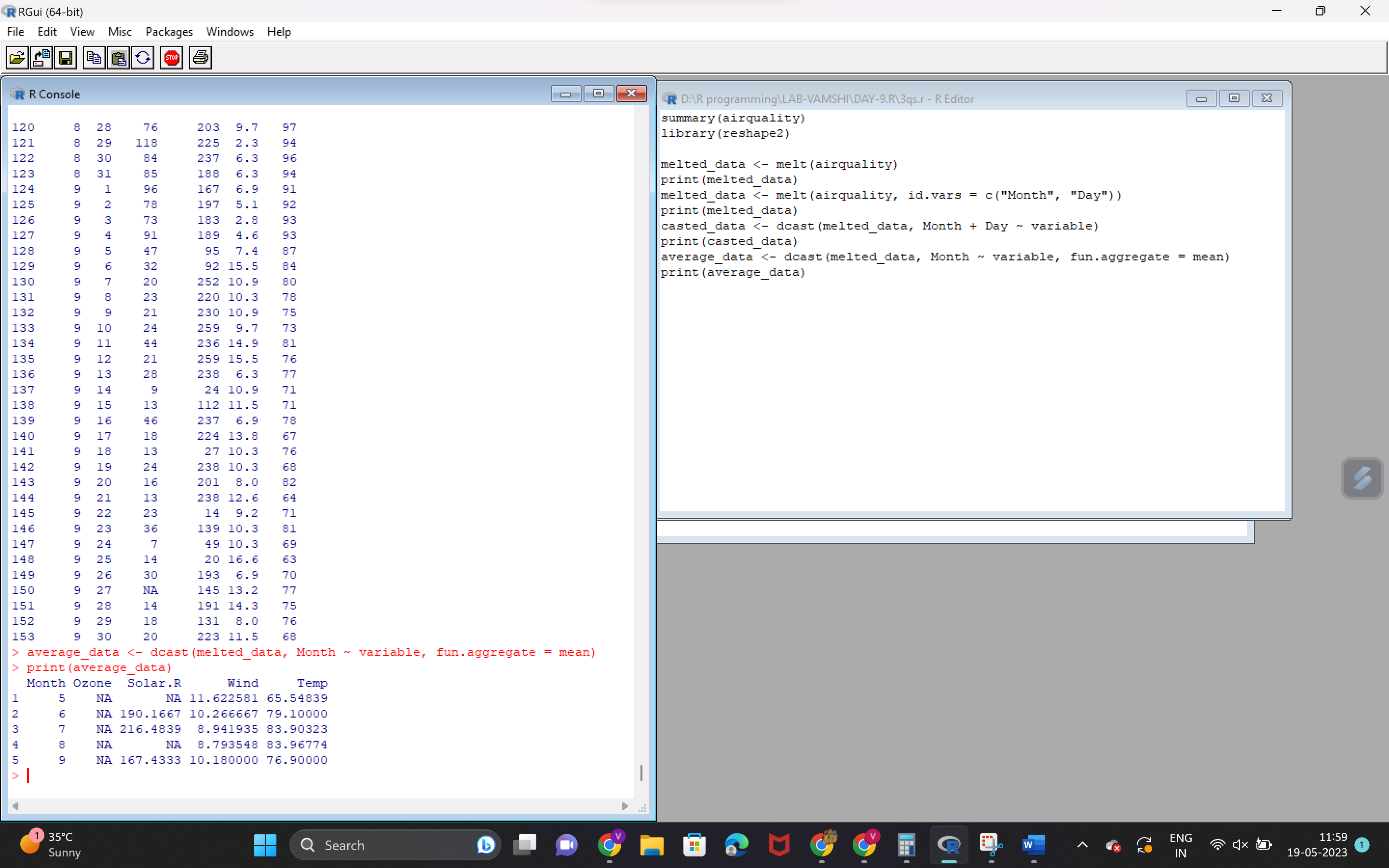
# Plot scatter plot

plot(airquality$Solar.R, airquality$Ozone, xlab = "Solar.R", ylab = "Ozone", main = "Scatter Plot")

# Add regression line

abline(model, col = "red")





**SET2**

**CODE1**

factorial <- function(n) {

result <- 1

for (i in 1:n) {

result <- result \* i

}

return(result)

}

set.seed(42) # Set seed for reproducibility

# Create matrix

matrix\_data <- matrix(sample(1:100, 12, replace = TRUE), nrow = 3, ncol = 4, byrow = TRUE,

dimnames = list(c("x", "y", "z"), c("uno", "dos", "tres", "cuatro")))

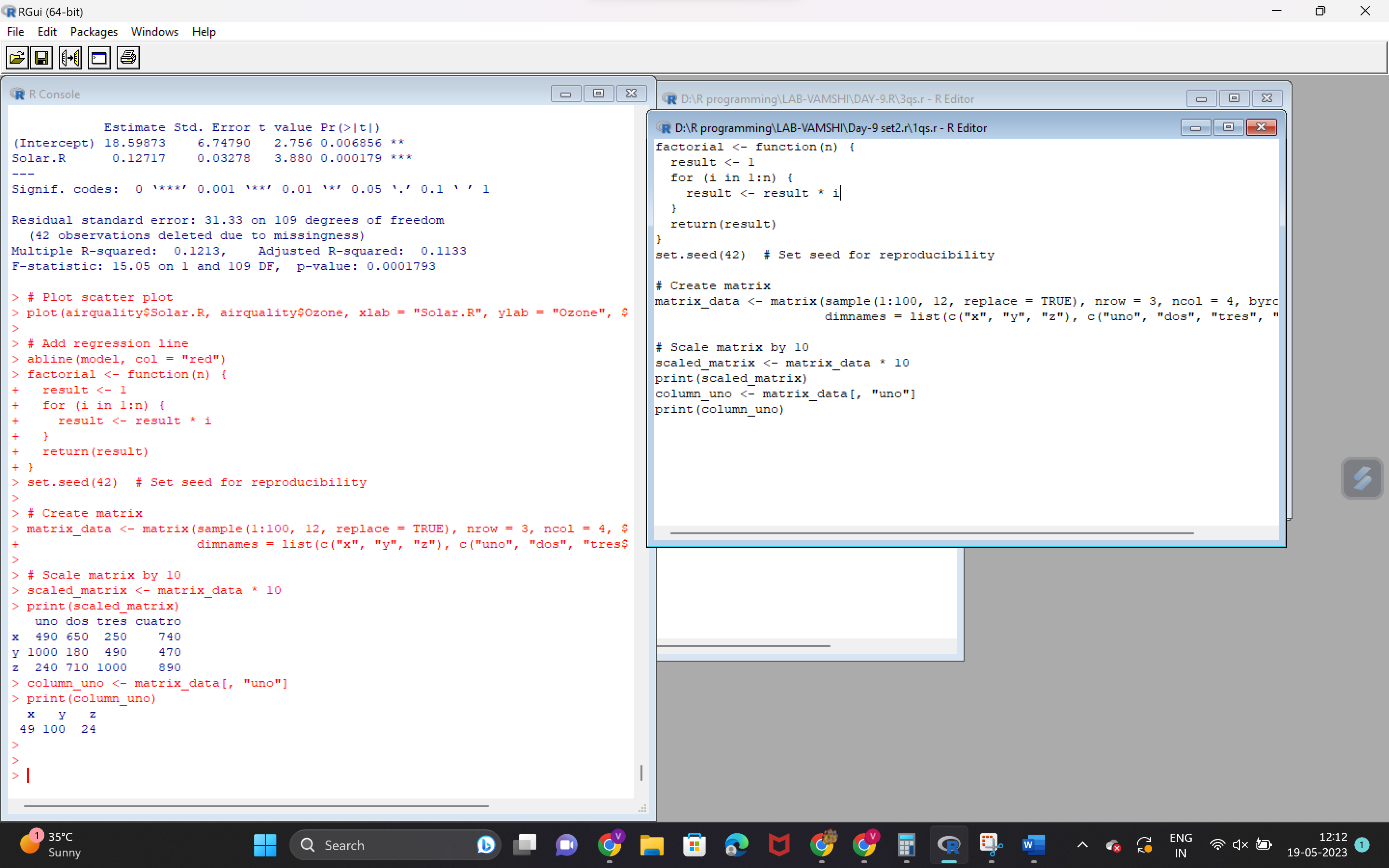
# Scale matrix by 10

scaled\_matrix <- matrix\_data \* 10

print(scaled\_matrix)

column\_uno <- matrix\_data[, "uno"]

print(column\_uno)



CODE2

factorial <- function(n) {

result <- 1

for (i in 1:n) {

result <- result \* i

}

return(result)

}

set.seed(42) # Set seed for reproducibility

# Create matrix

matrix\_data <- matrix(sample(1:100, 12, replace = TRUE), nrow = 3, ncol = 4, byrow = TRUE,

dimnames = list(c("x", "y", "z"), c("uno", "dos", "tres", "cuatro")))

# Scale matrix by 10

scaled\_matrix <- matrix\_data \* 10

print(scaled\_matrix)

column\_uno <- matrix\_data[, "uno"]

print(column\_uno)

