SAVEETHA SCHOOL OF ENGINEERING

SAVEETHA INSTITUTE OF MEDICAL AND TECHNICAL SCIENCES

ITA 0451 - STATISTICS WITH R PROGRAMMING

DAY 4 – LAB ASSESSMENT Part 3

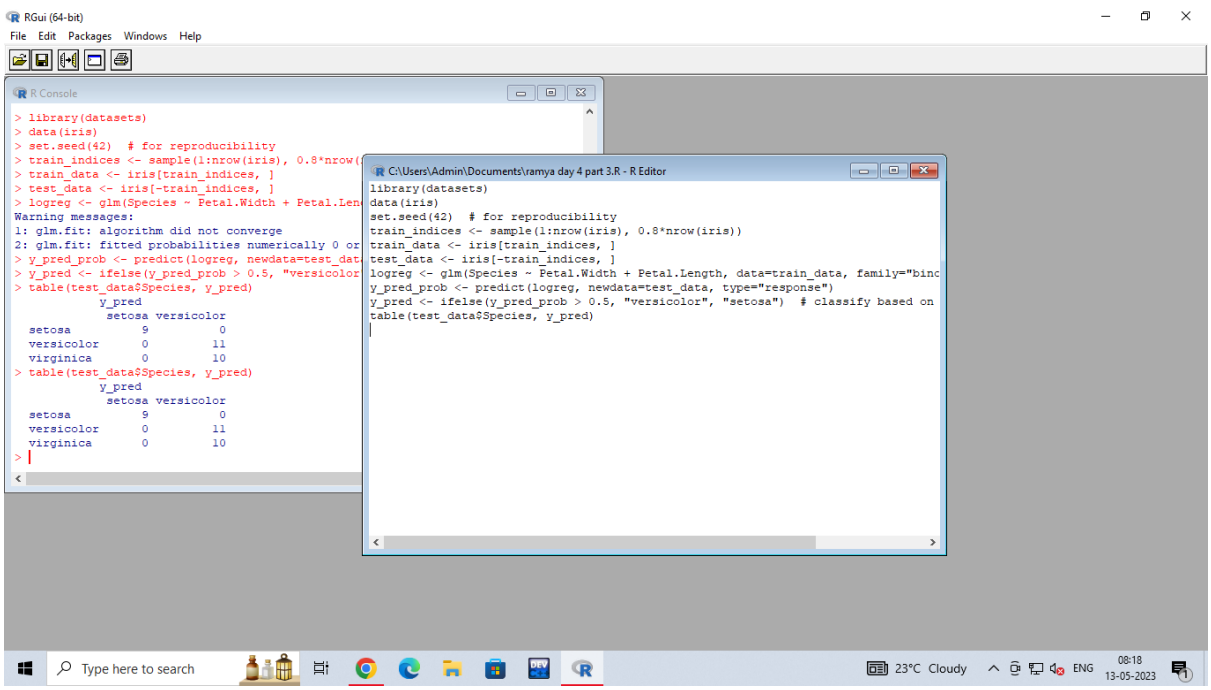
Reg No:192124039

Name:ramya sri.v

1.Randomly Sample the iris dataset such as 80% data for training and 20% for test and create Logistics regression with train data, use species as target and petals width and length as feature variables , Predict the probability of the model using test data, Create Confusion matrix for above test model

program:

```
library(datasets)
data(iris)
set.seed(42)  # for reproducibility
train_indices <- sample(1:nrow(iris), 0.8*nrow(iris))
train_data <- iris[train_indices, ]
test_data <- iris[-train_indices, ]
logreg <- glm(Species ~ Petal.Width + Petal.Length, data=train_data, family="binomial")
y_pred_prob <- predict(logreg, newdata=test_data, type="response")
y_pred <- ifelse(y_pred_prob > 0.5, "versicolor", "setosa")  # classify based on the threshold of 0.5
table(test_data$Species, y_pred)
```

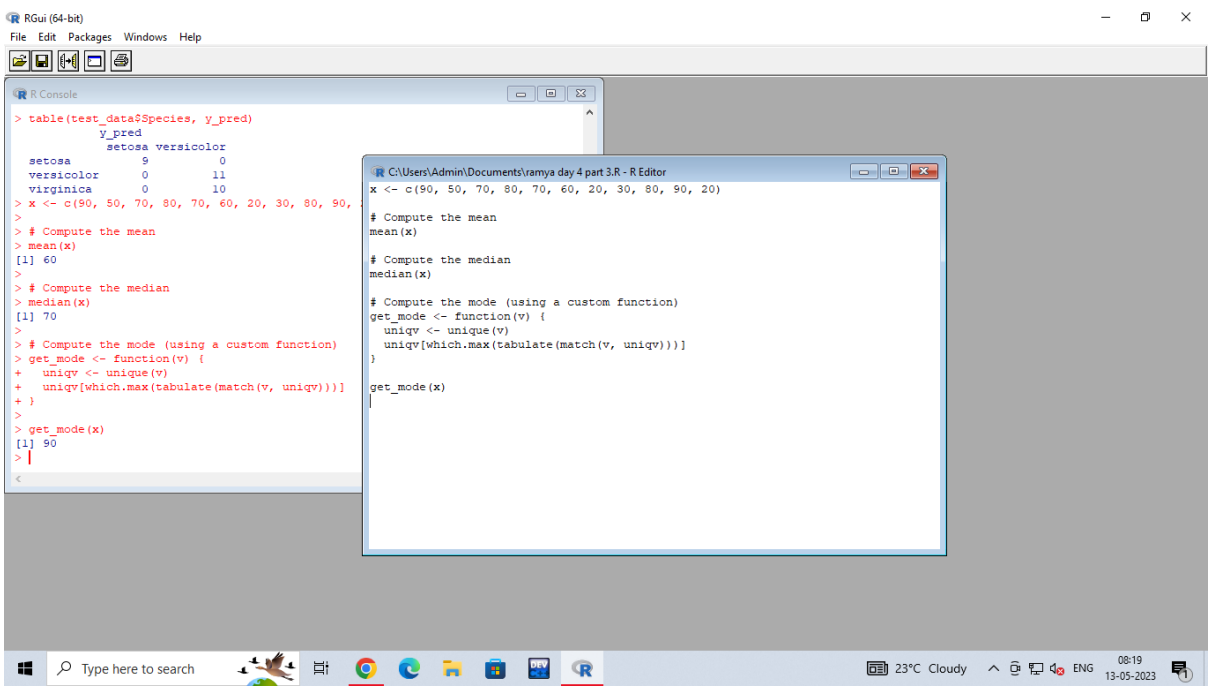


2. (i)Write suitable R code to compute the mean, median ,mode of the following values

c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

program:

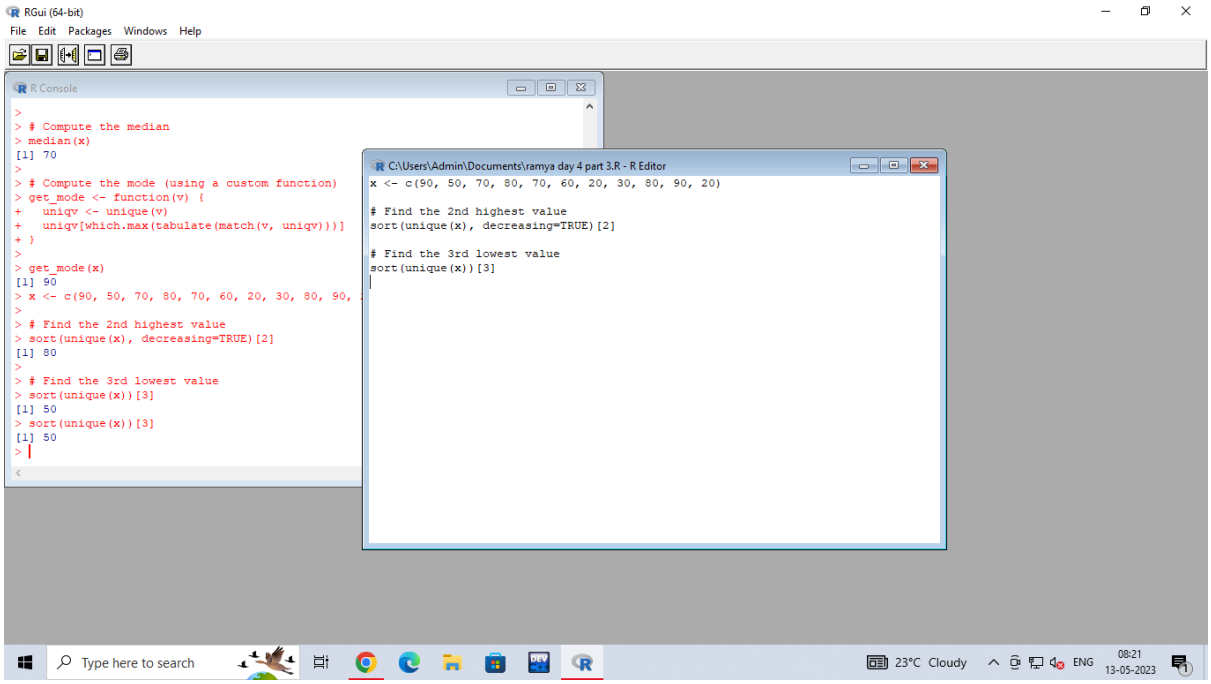
```
x <- c(90, 50, 70, 80, 70, 60, 20, 30, 80, 90, 20)

# Compute the mean
mean(x)

# Compute the median
median(x)

# Compute the mode (using a custom function)
get_mode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

get_mode(x)
```



(ii) Write R code to find 2nd highest and 3rd Lowest value of above problem.

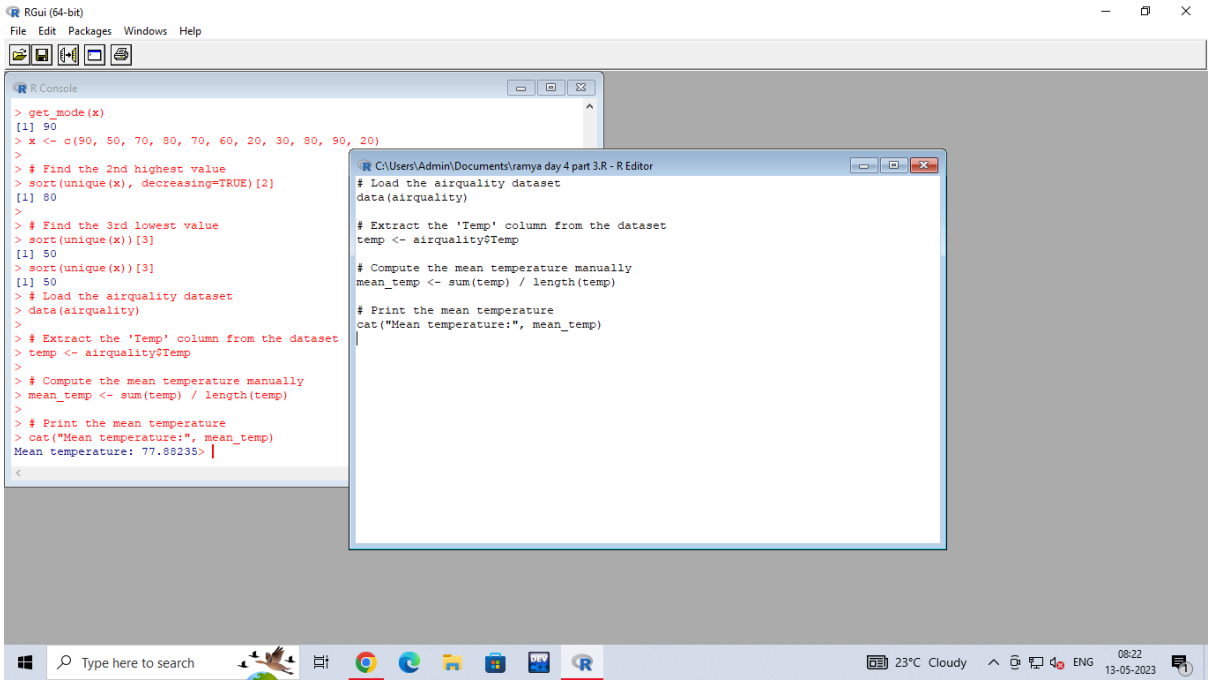
program:

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

# Find the 2nd highest value
sort(unique(x), decreasing=TRUE)[2]

# Find the 3rd lowest value
sort(unique(x))[3]
```



3. Explore the airquality dataset. It contains daily air quality measurements from New York during a period of five months:
• Ozone: mean ozone concentration (ppb), • Solar.R: solar radiation (Langley),
• Wind: average wind speed (mph), • Temp: maximum daily temperature in degrees Fahrenheit,
• Month: numeric month (May=5, June=6, and so on),• Day: numeric day of the month (1 - 4).
i. Compute the mean temperature(don't use build in function)
program:

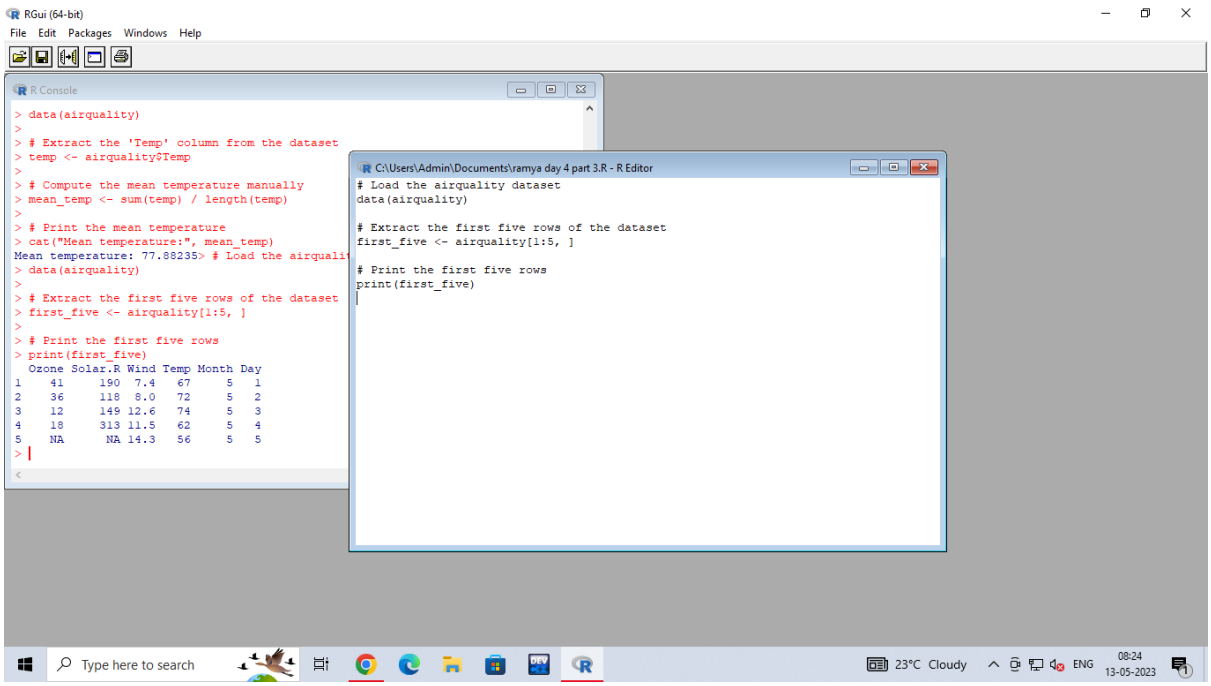
```
# Load the airquality dataset
data(airquality)

# Extract the 'Temp' column from the dataset
temp <- airquality$Temp

# Compute the mean temperature manually
mean_temp <- sum(temp) / length(temp)

# Print the mean temperature
cat("Mean temperature:", mean_temp)
```



ii.Extract the first five rows from airquality.

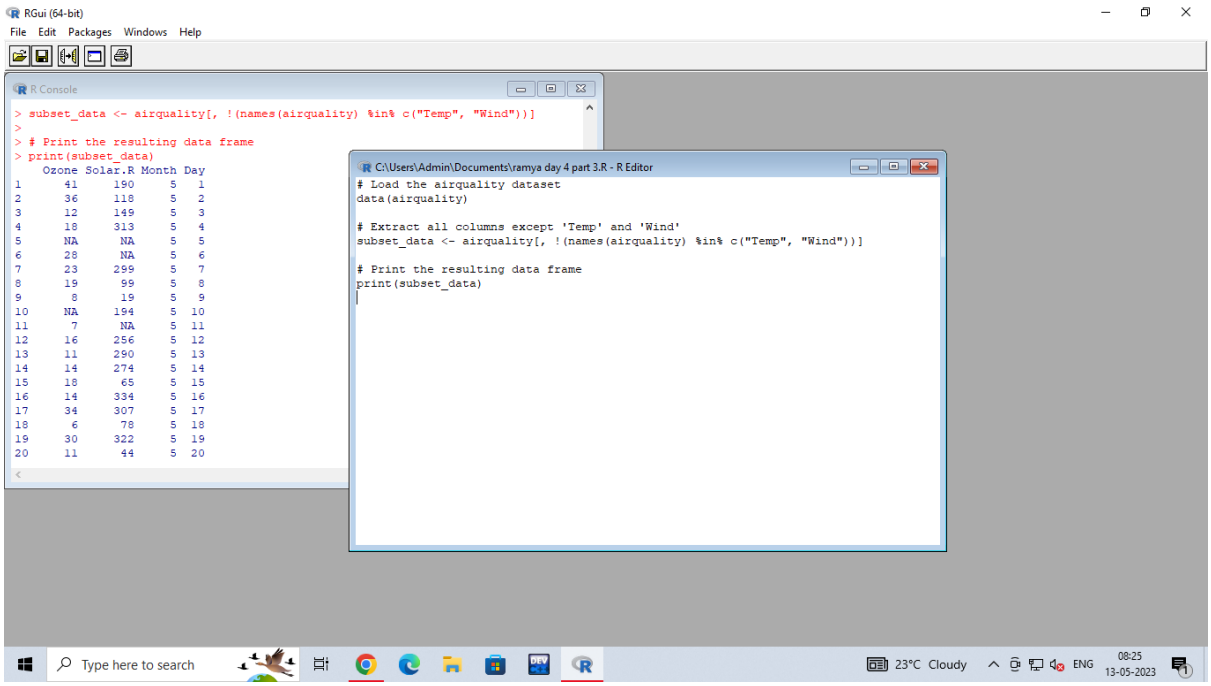
program:

```
# Load the airquality dataset
data(airquality)

# Extract the first five rows of the dataset
first_five <- airquality[1:5, ]

# Print the first five rows
print(first_five)
```



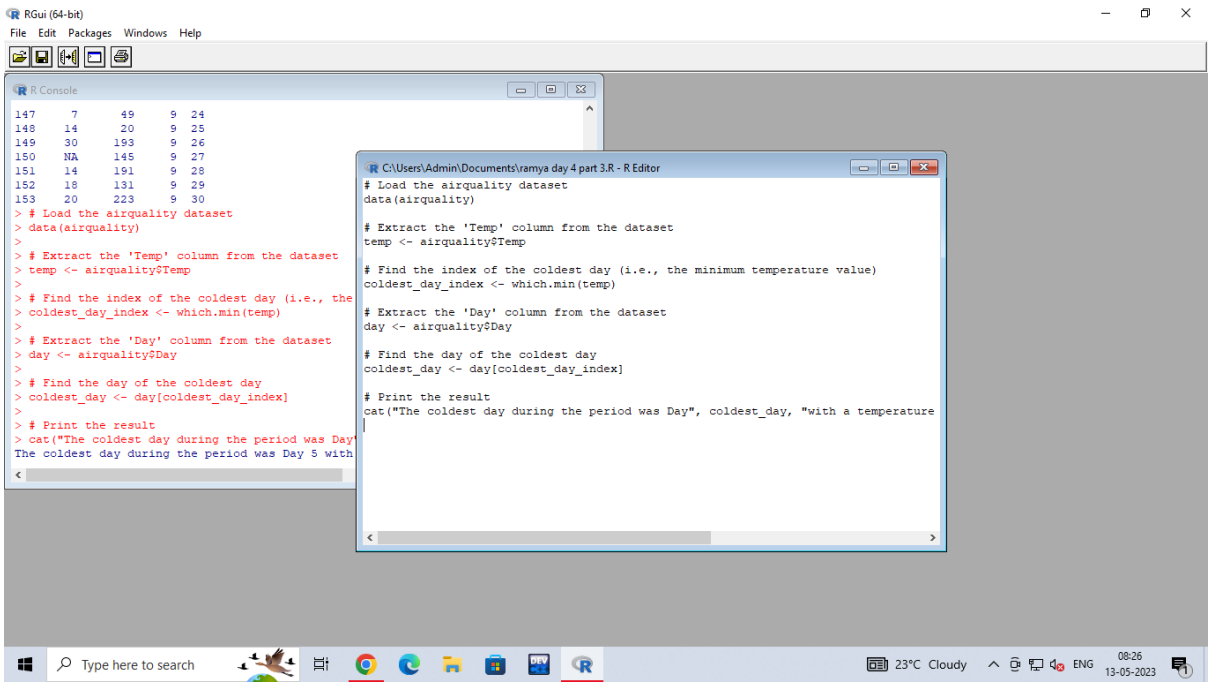
iii.Extract all columns from airquality except Temp and Wind

program:
```
# Load the airquality dataset
data(airquality)

# Extract all columns except 'Temp' and 'Wind'
subset_data <- airquality[, !(names(airquality) %in% c("Temp", "Wind"))]

# Print the resulting data frame
print(subset_data)
```



iv.Which was the coldest day during the period?
program:
```
# Load the airquality dataset
data(airquality)

# Extract the 'Temp' column from the dataset
temp <- airquality$Temp

# Find the index of the coldest day (i.e., the minimum temperature value)
coldest_day_index <- which.min(temp)

# Extract the 'Day' column from the dataset
day <- airquality$Day

# Find the day of the coldest day
coldest_day <- day[coldest_day_index]

# Print the result
```

cat("The coldest day during the period was Day", coldest_day, "with a temperature of", temp[coldest_day_index], "degrees Fahrenheit.")

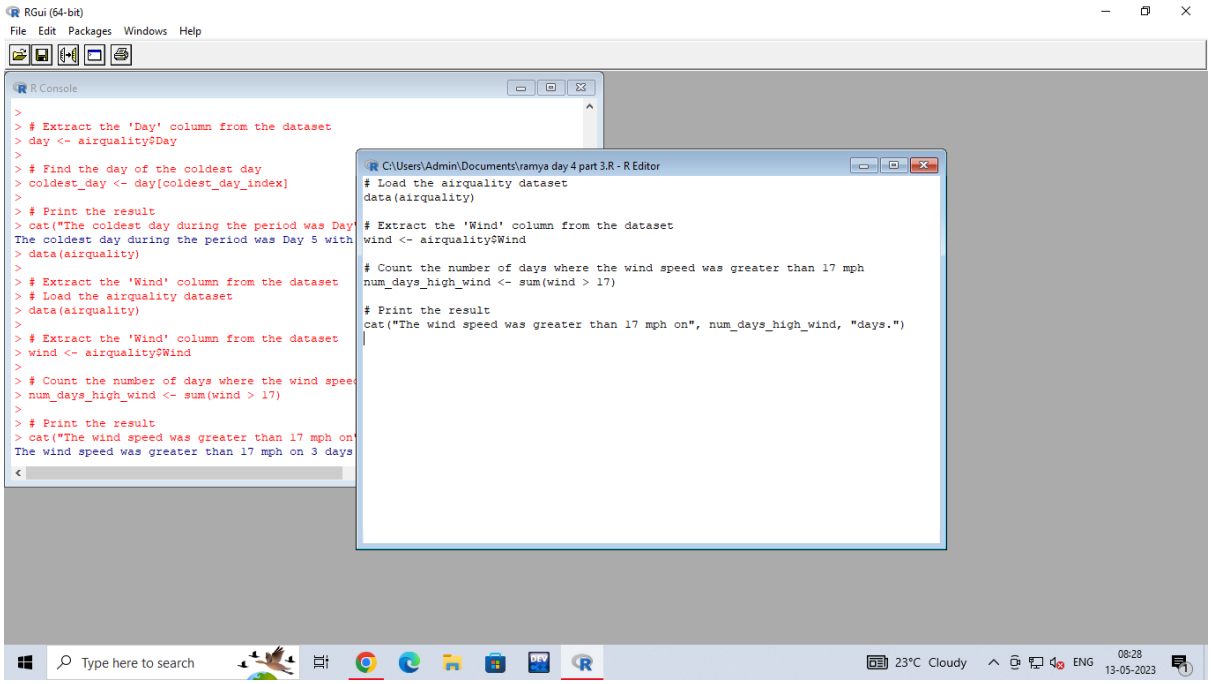


v.How many days was the wind speed greater than 17 mph?
program:
# Load the airquality dataset
data(airquality)

# Extract the 'Wind' column from the dataset
wind <- airquality$Wind

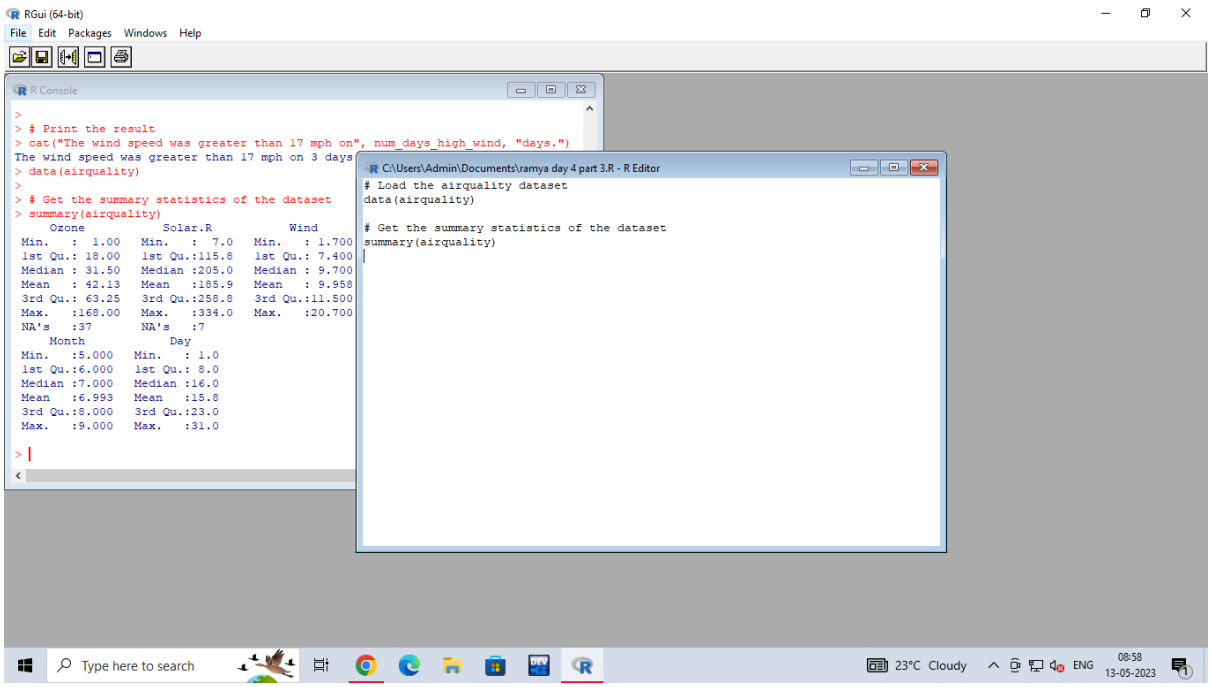
# Count the number of days where the wind speed was greater than 17 mph
num_days_high_wind <- sum(wind > 17)

# Print the result
cat("The wind speed was greater than 17 mph on", num_days_high_wind, "days.")



4. (i)Get the Summary Statistics of air quality dataset

program:



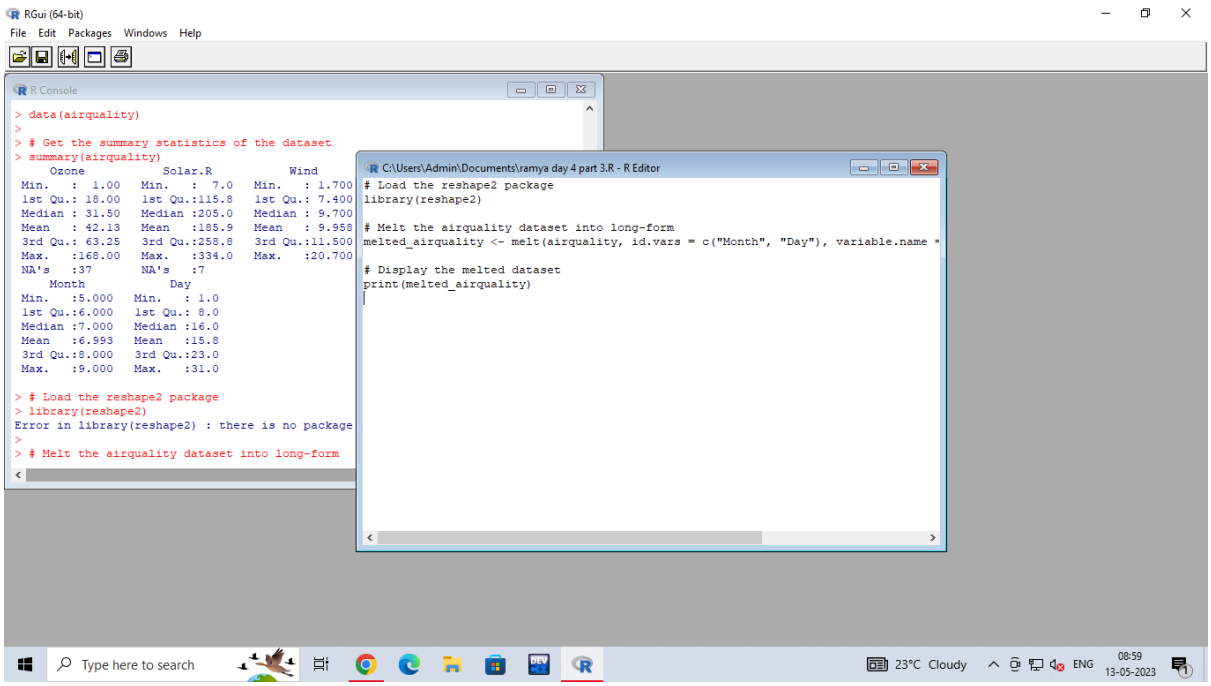
# Load the airquality dataset
data(airquality)

```
# Get the summary statistics of the dataset
summary(airquality)
```

(ii)Melt airquality data set and display as a long – format data?

program:

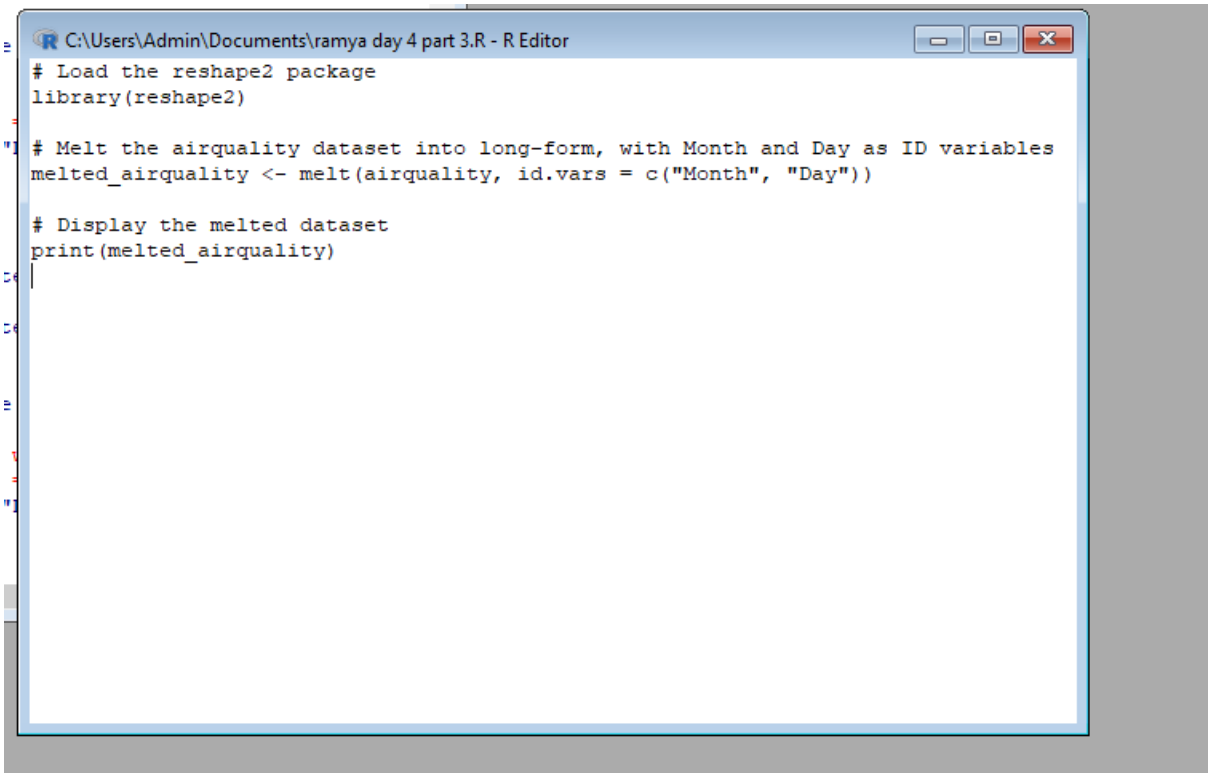
```
# Load the reshape2 package
library(reshape2)

# Melt the airquality dataset into long-form
melted_airquality <- melt(airquality, id.vars = c("Month", "Day"), variable.name =
"Variable", value.name = "Value")

# Display the melted dataset
print(melted_airquality)
```



(iii)Melt airquality data and specify month and day to be “ID variables”?

program:

```
# Load the reshape2 package
library(reshape2)

# Melt the airquality dataset into long-form, with Month and Day as ID variables
melted_airquality <- melt(airquality, id.vars = c("Month", "Day"))

# Display the melted dataset
print(melted_airquality)
```

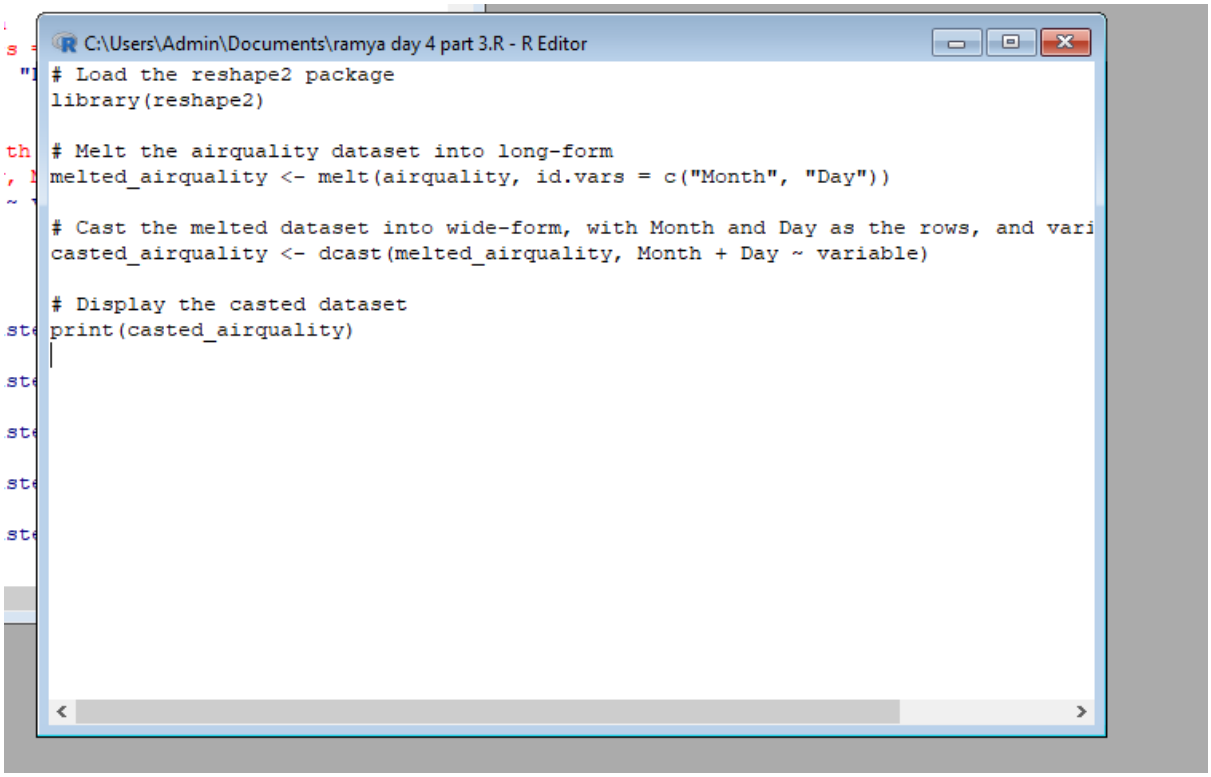


(iv)Cast the molten airquality data set with respect to month and date features

program:

Load the reshape2 package

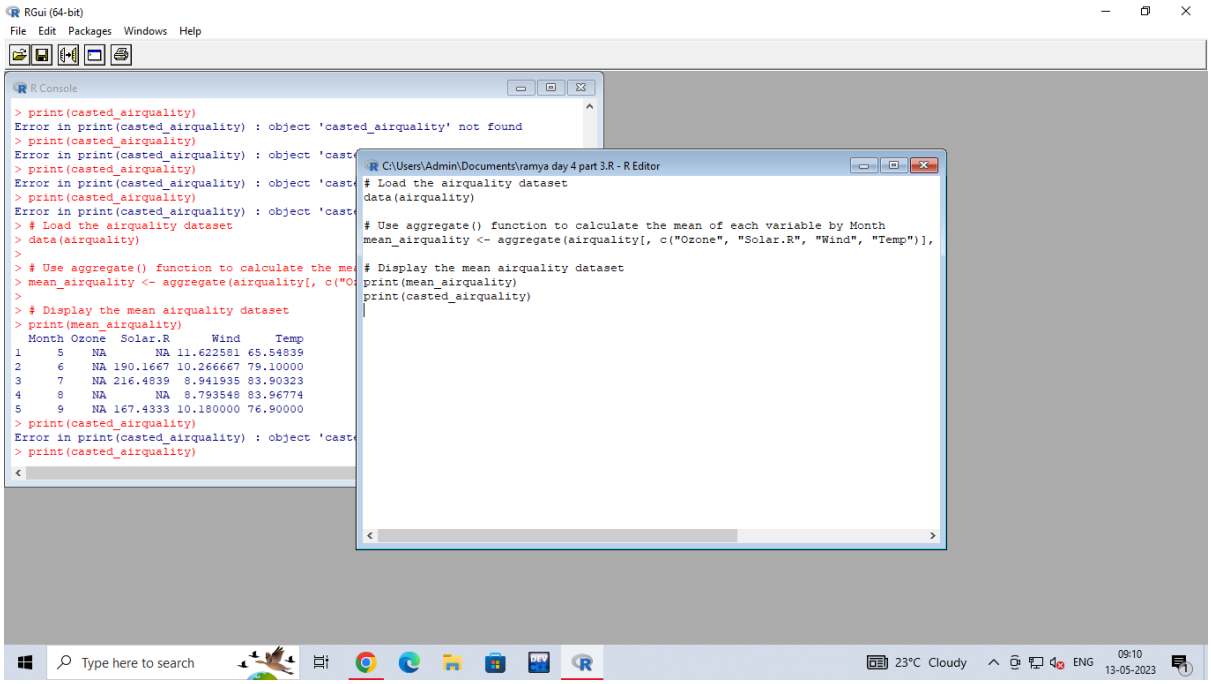
```
library(reshape2)

# Melt the airquality dataset into long-form
melted_airquality <- melt(airquality, id.vars = c("Month", "Day"))

# Cast the melted dataset into wide-form, with Month and Day as the rows, and variables as columns
casted_airquality <- dcast(melted_airquality, Month + Day ~ variable)

# Display the casted dataset
print(casted_airquality)
```



(v) Use cast function appropriately and compute the average of Ozone, Solar.R , Wind and temperature per month?

prgram:

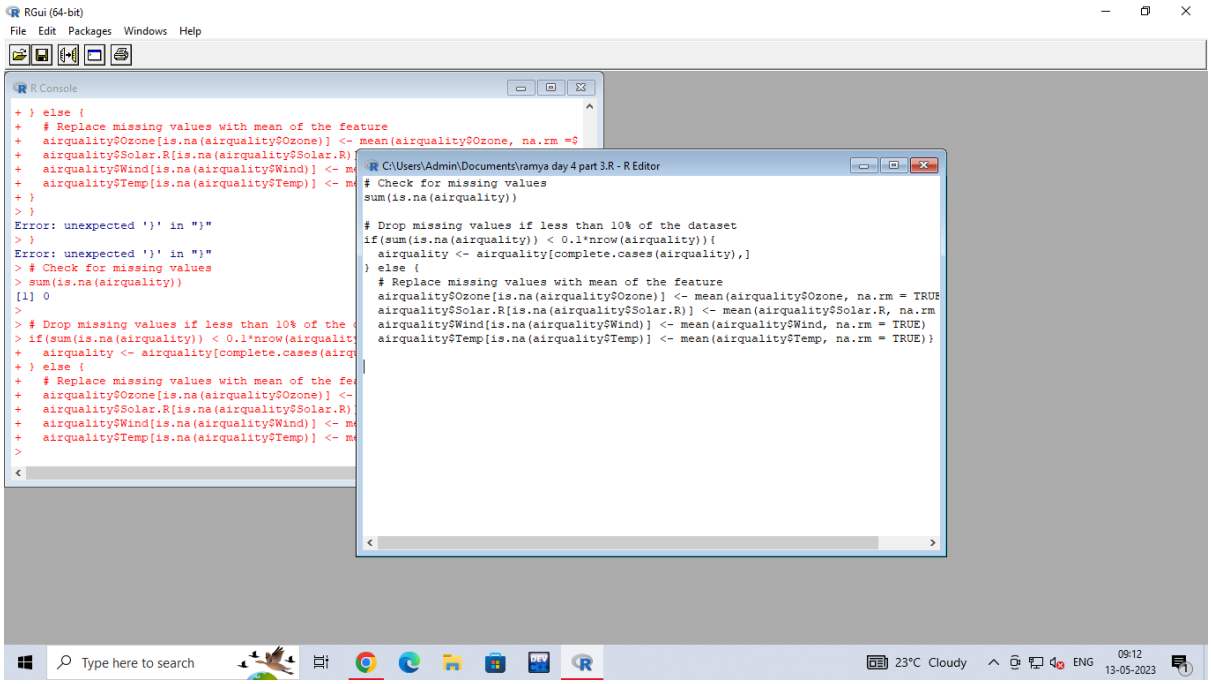
```
# Load the airquality dataset
data(airquality)

# Use aggregate() function to calculate the mean of each variable by Month
mean_airquality <- aggregate(airquality[, c("Ozone", "Solar.R", "Wind", "Temp")], by = list(Month = airquality$Month), mean)

# Display the mean airquality dataset
print(mean_airquality)
print(casted_airquality)
```



5.(i) Find any missing values(na) in features and drop the missing values if its less than 10% else replace that with mean of that feature.

program:

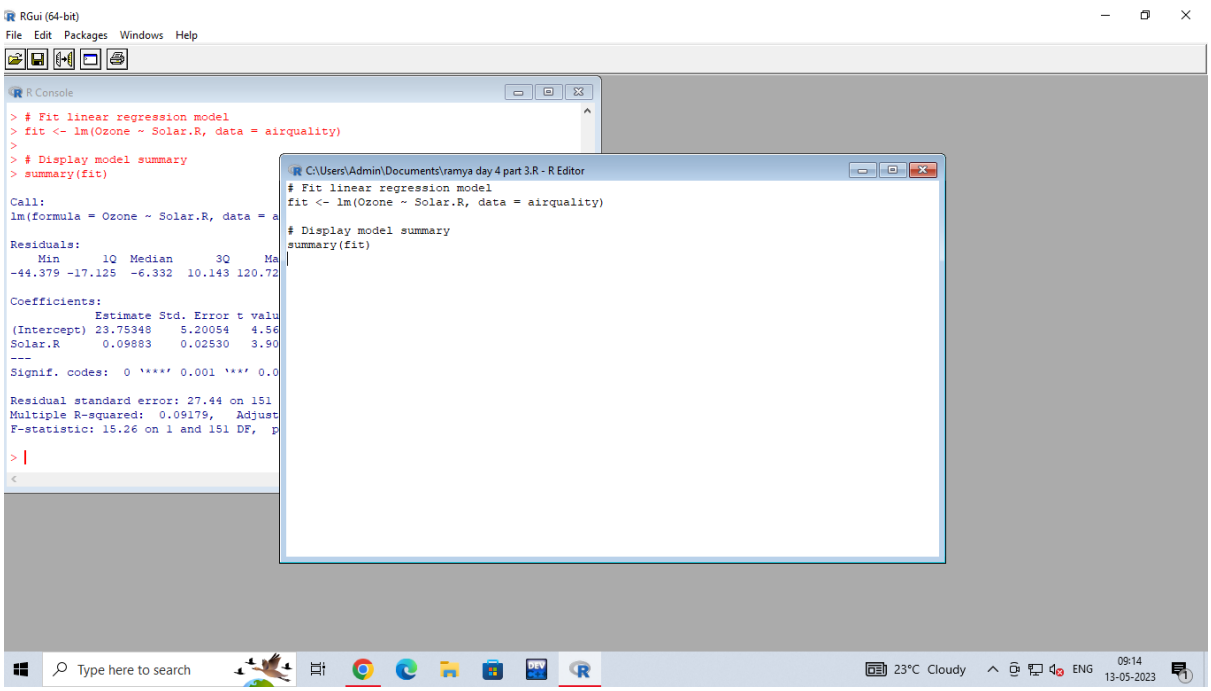
```
# Check for missing values
sum(is.na(airquality))

# Drop missing values if less than 10% of the dataset
if(sum(is.na(airquality)) < 0.1*nrow(airquality)){
  airquality <- airquality[complete.cases(airquality),]
} else {
  # Replace missing values with mean of the feature
  airquality$Ozone[is.na(airquality$Ozone)] <- mean(airquality$Ozone, na.rm = TRUE)
  airquality$Solar.R[is.na(airquality$Solar.R)] <- mean(airquality$Solar.R, na.rm = TRUE)
  airquality$Wind[is.na(airquality$Wind)] <- mean(airquality$Wind, na.rm = TRUE)
  airquality$Temp[is.na(airquality$Temp)] <- mean(airquality$Temp, na.rm = TRUE)}
```



(ii) Apply a linear regression algorithm using Least Squares Method on “Ozone” and “Solar.R”

program:

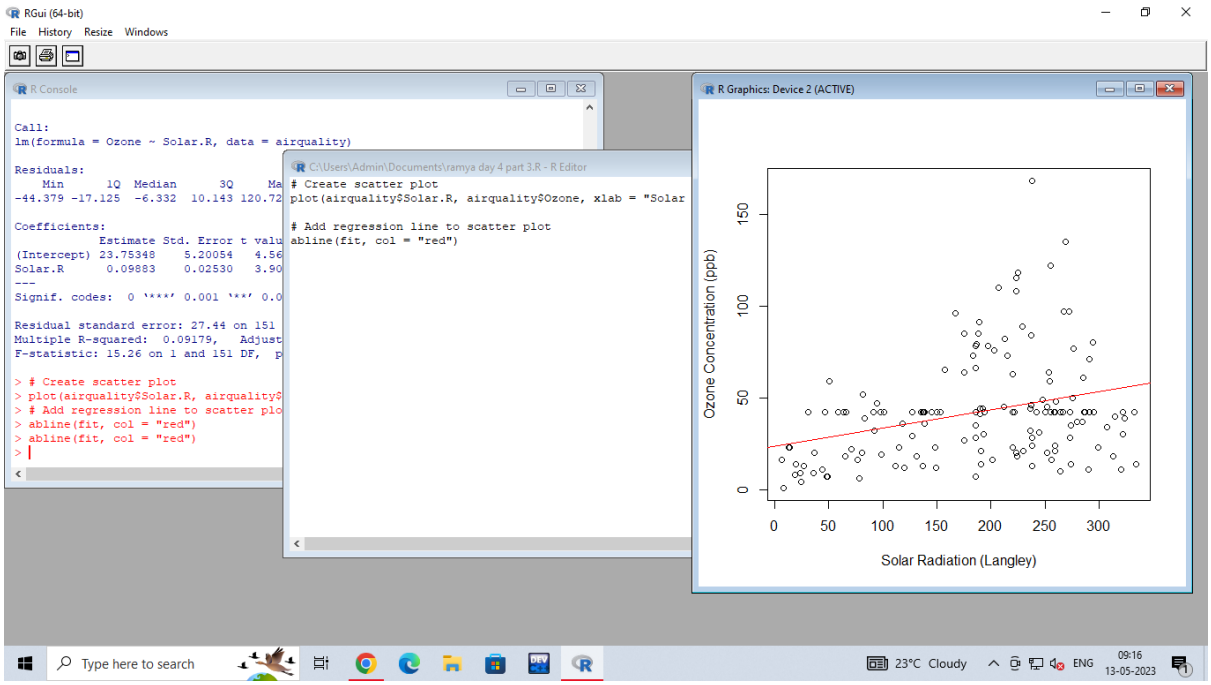
```
# Fit linear regression model
fit <- lm(Ozone ~ Solar.R, data = airquality)

# Display model summary
summary(fit)
```



(iii)Plot Scatter plot between Ozone and Solar and add regression line created by above model

program:
# Create scatter plot
plot(airquality$Solar.R, airquality$Ozone, xlab = "Solar Radiation (Langley)", ylab = "Ozone Concentration (ppb)")

# Add regression line to scatter plot
abline(fit, col = "red")



6. Load dataset named ChickWeight,

(i).Order the data frame, in ascending order by feature name “weight” grouped by feature

“diet” and Extract the last 6 records from order data frame.

(ii).a Perform melting function based on “Chick", "Time", "Diet" features as ID variables

b. Perform cast function to display the mean value of weight grouped by Diet

c. Perform cast function to display the mode of weight grouped by Diet

program(i,ii):
load the ChickWeight dataset

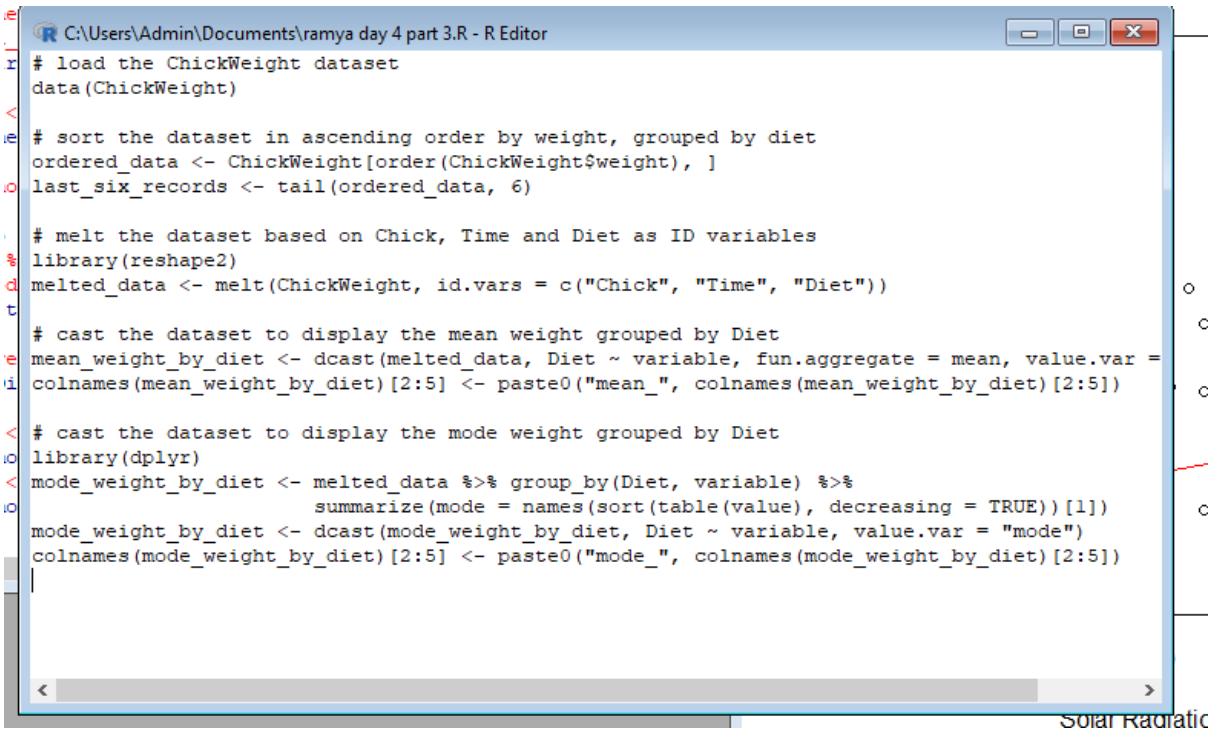
data(ChickWeight)

sort the dataset in ascending order by weight, grouped by diet

ordered_data <- ChickWeight[order(ChickWeight$weight),]

last_six_records <- tail(ordered_data, 6)

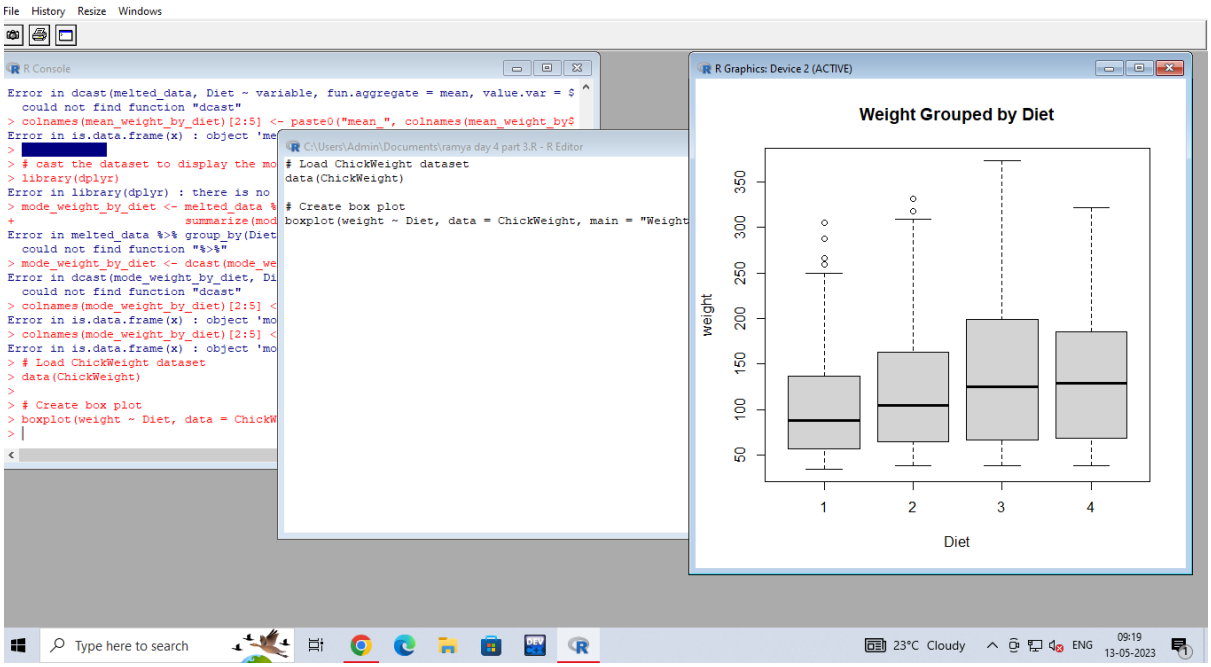
```
# melt the dataset based on Chick, Time and Diet as ID variables

library(reshape2)

melted_data <- melt(ChickWeight, id.vars = c("Chick", "Time", "Diet"))


# cast the dataset to display the mean weight grouped by Diet

mean_weight_by_diet <- dcast(melted_data, Diet ~ variable, fun.aggregate = mean, value.var
= "value")

colnames(mean_weight_by_diet)[2:5] <- paste0("mean_",
colnames(mean_weight_by_diet)[2:5])


# cast the dataset to display the mode weight grouped by Diet

library(dplyr)

mode_weight_by_diet <- melted_data %>% group_by(Diet, variable) %>%

                summarize(mode = names(sort(table(value), decreasing = TRUE))[1])

mode_weight_by_diet <- dcast(mode_weight_by_diet, Diet ~ variable, value.var = "mode")

colnames(mode_weight_by_diet)[2:5] <- paste0("mode_",
colnames(mode_weight_by_diet)[2:5])
```



7. a. Create Box plot for “weight” grouped by “Diet”

program:

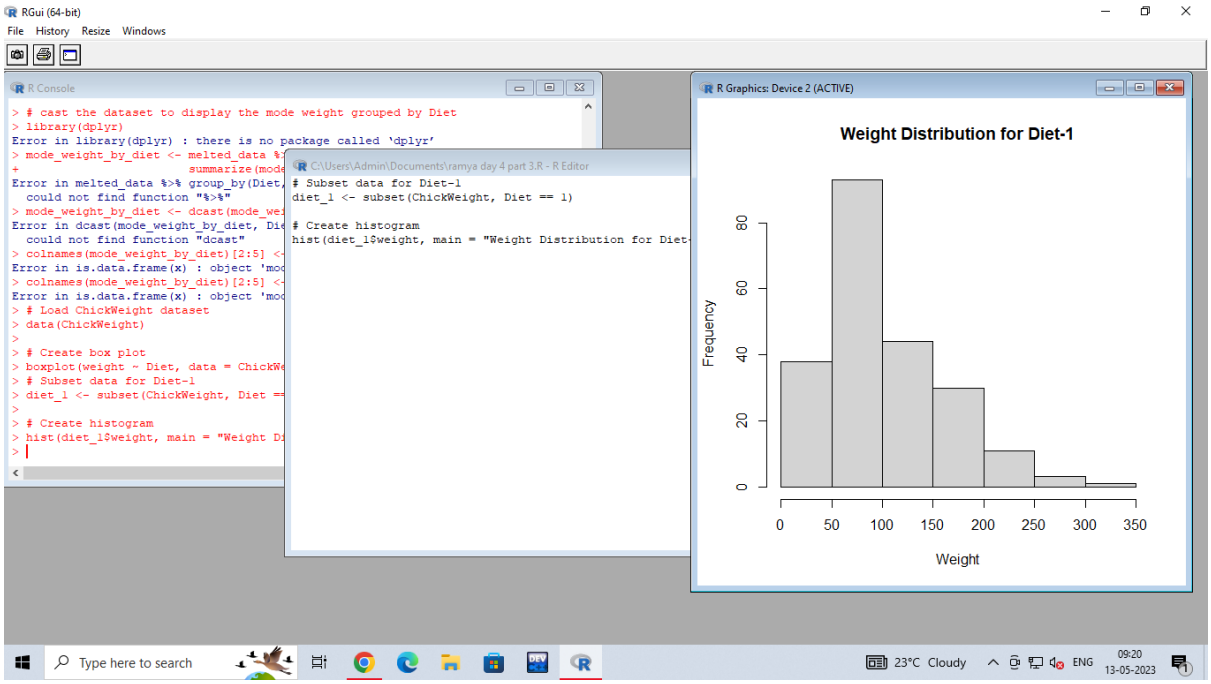
```
# Load ChickWeight dataset
data(ChickWeight)

# Create box plot
boxplot(weight ~ Diet, data = ChickWeight, main = "Weight Grouped by Diet")
```



b. Create a Histogram for “weight” features belong to Diet- 1 category

program:

```
# Subset data for Diet-1
diet_1 <- subset(ChickWeight, Diet == 1)

# Create histogram
hist(diet_1$weight, main = "Weight Distribution for Diet-1", xlab = "Weight")
```



c. Create Scatter plot for " weight" vs "Time" grouped by Diet

program:

Create scatter plot

plot(weight ~ Time, data = ChickWeight, col = Diet, main = "Weight vs Time Grouped by Diet",

xlab = "Time", ylab = "Weight")

legend("topright", legend = levels(as.factor(ChickWeight$Diet)), col = 1:4, pch = 1)