

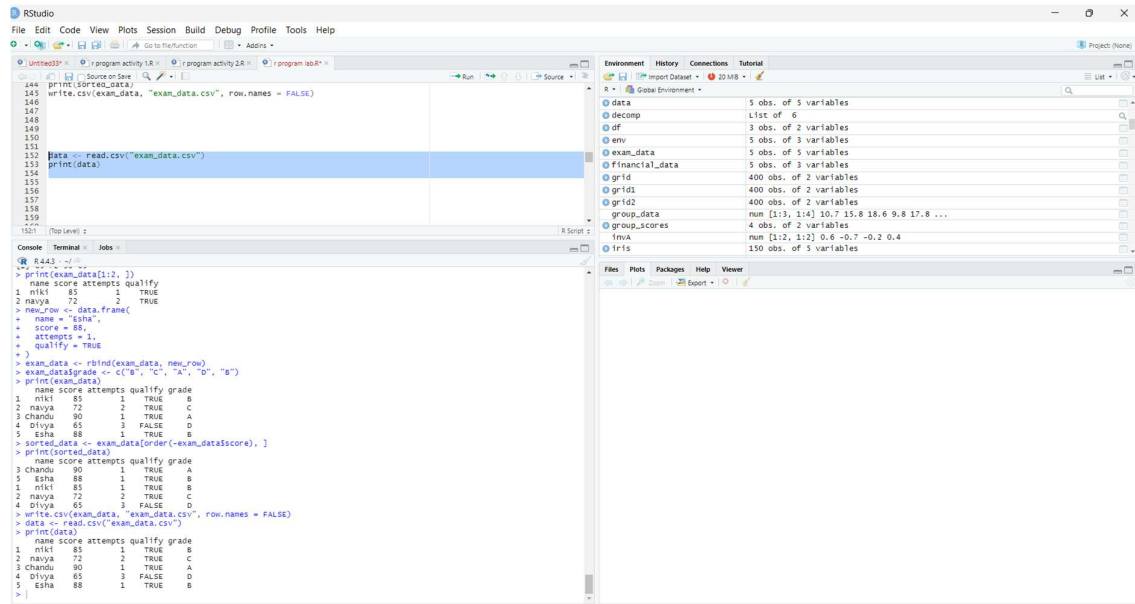
11. Write an R program to read a .csv file and display contents.

CODE:

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

```
print(data)
```

OUTPUT:



12. Perform data reshaping on airquality dataset: melt, cast, compute monthly averages for Ozone, Solar.R, Wind, and Temperature.

CODE:

```
data(airquality)
```

```
monthly_avg <- aggregate(
```

```
  cbind(Ozone, Solar.R, Wind, Temp) ~ Month,
```

```
  data = airquality,
```

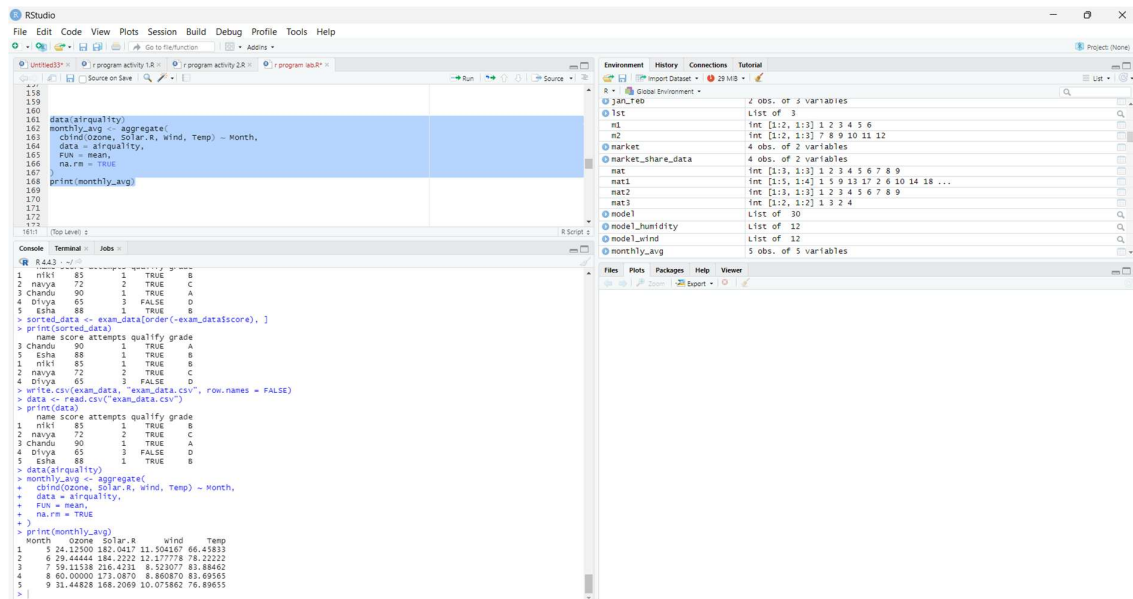
```
  FUN = mean,
```

```
  na.rm = TRUE
```

```
)
```

```
print(monthly_avg)
```

OUTPUT:



13. Combine multiple arrays row-wise.

CODE:

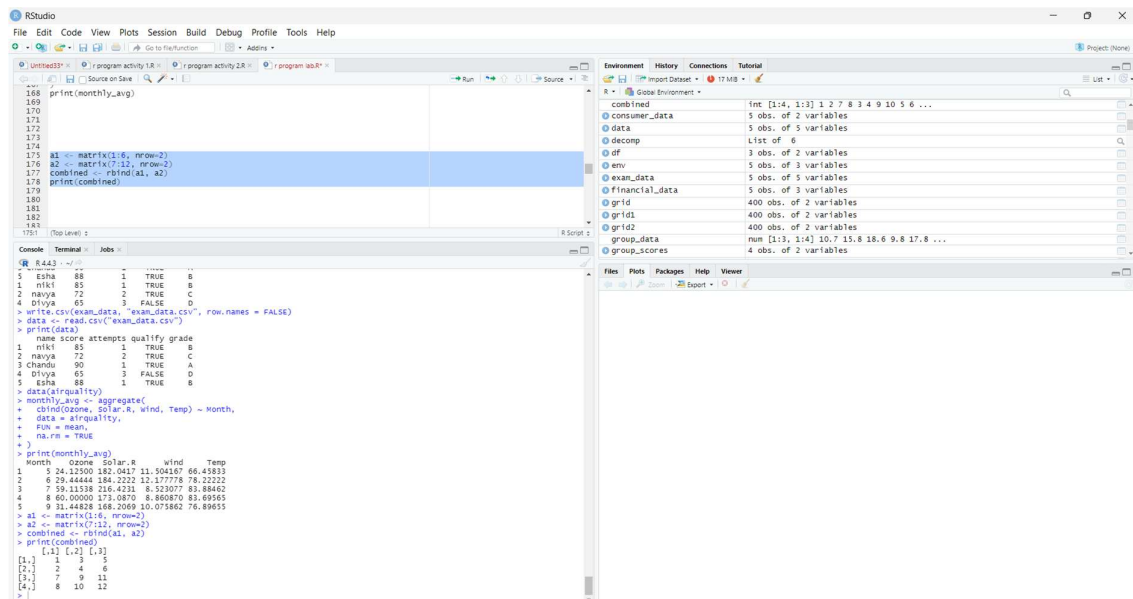
```
a1 <- matrix(1:6, nrow=2)
```

```
a2 <- matrix(7:12, nrow=2)
```

```
combined <- rbind(a1, a2)
```

```
print(combined)
```

OUTPUT:



14. Explore and manipulate ChickWeight dataset (sorting, melting, casting by Diet).

CODE:

```
data(ChickWeight)

head(ChickWeight)

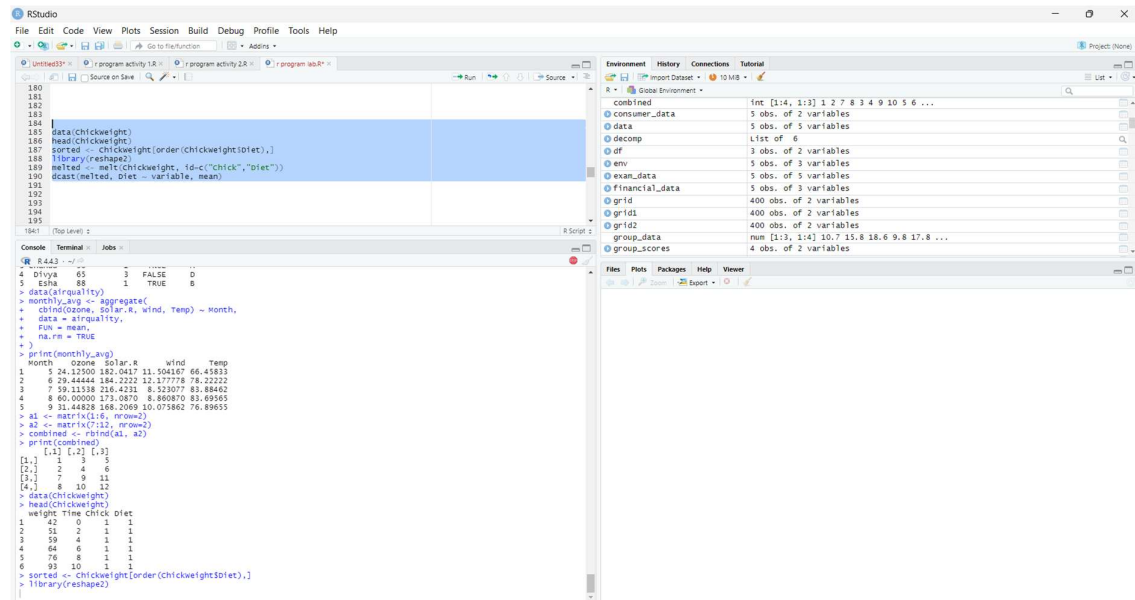
sorted <- ChickWeight[order(ChickWeight$Diet),]

library(reshape2)

melted <- melt(ChickWeight, id=c("Chick", "Diet"))

dcast(melted, Diet ~ variable, mean)
```

OUTPUT:



```
R 4.4.3 ~ /~
4 Olyya 65 3 FALSE 0
5 Esna 88 1 TRUE 8
> data(airquality)
> monthly_avg <- aggregate(
+   cbind(ozone, solar.r, wind, Temp) ~ Month,
+   data = airquality,
+   FUN = mean,
+   na.rm = TRUE
+ )
> print(monthly_avg)
      Month      Ozone      solar.r      wind      Temp
1      12      12.900      182.0417      11.504167      66.45833
2       1       29.44444      186.2222      12.177778      78.22222
3       2       59.11538      216.4231      8.523077      83.88462
4       3       60.00000      173.0870      8.860870      81.69565
5       4       31.44828      146.2069      10.075862      76.89955
> a1 <- matrix(1:6, nrow=2)
> a2 <- matrix(7:12, nrow=2)
> combined <- rbind(a1, a2)
> print(combined)
      [,1] [,2] [,3]
[1,] 1  2  3
[2,] 4  5  6
[3,] 7  8  9
[4,] 10 11 12
> data(ChickWeight)
> head(ChickWeight)
  weight time chick diet
1    42.0   0     1    1
2    51.2   2     1    1
3    59.4   1     1    1
4    64.6   1     1    1
5    76.8   1     1    1
6    93.10  1     1    1
> sorted <- ChickWeight[order(ChickWeight$Diet),]
> library(reshape2)
```

15. Perform EDA on iris dataset: dimensions, summary, standard deviation, quantiles, grouping by Species, pivot table, categorical grouping with Sepal.Length categories.

CODE:

```
data(iris)

dim(iris)

summary(iris)

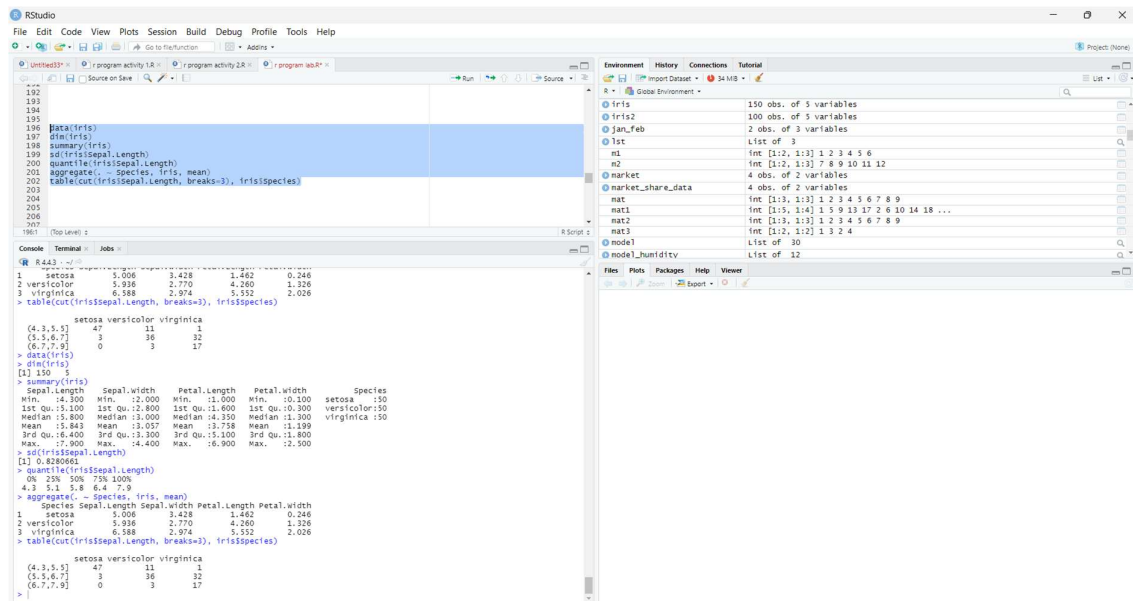
sd(iris$Sepal.Length)

quantile(iris$Sepal.Length)

aggregate(. ~ Species, iris, mean)

table(cut(iris$Sepal.Length, breaks=3), iris$Species)
```

OUTPUT:



16. Explore USArrests dataset: summary statistics, state with largest rape arrests, max & min murder rates, correlation among features, states above median assault arrests and bottom 25% for murder, visualization with histogram, density, scatterplots, bar graphs.

CODE:

```
data(USArrests)

summary(USArrests)

rownames(USArrests)[which.max(USArrests$Rape)]

max(USArrests$Murder); min(USArrests$Murder)

cor(USArrests)

USArrests[USArrests$Assault > median(USArrests$Assault),]

USArrests[USArrests$Murder < quantile(USArrests$Murder,0.25),]

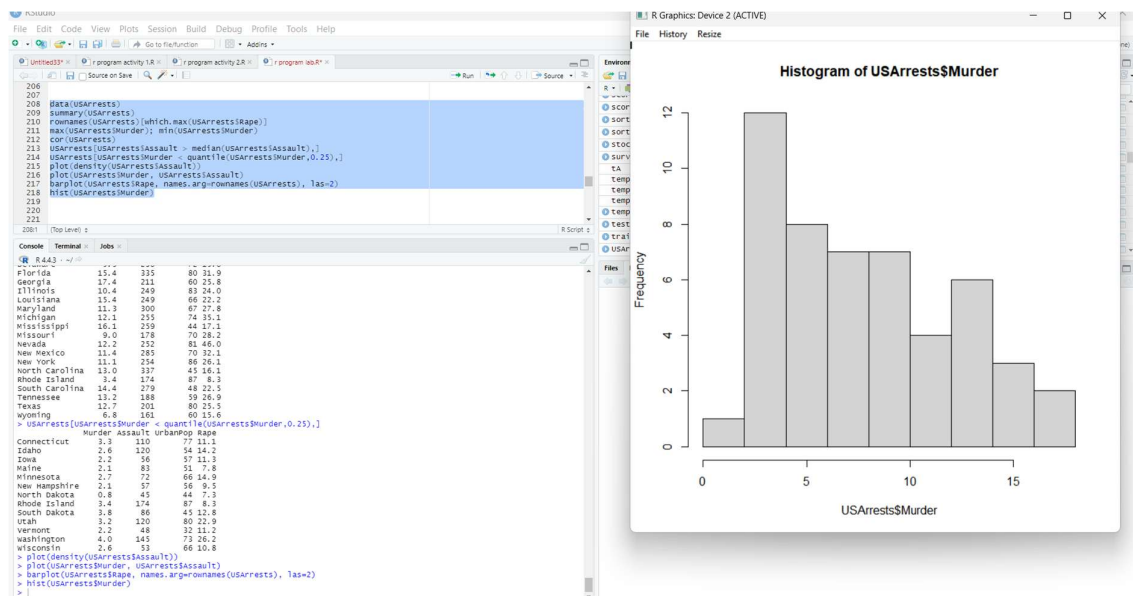
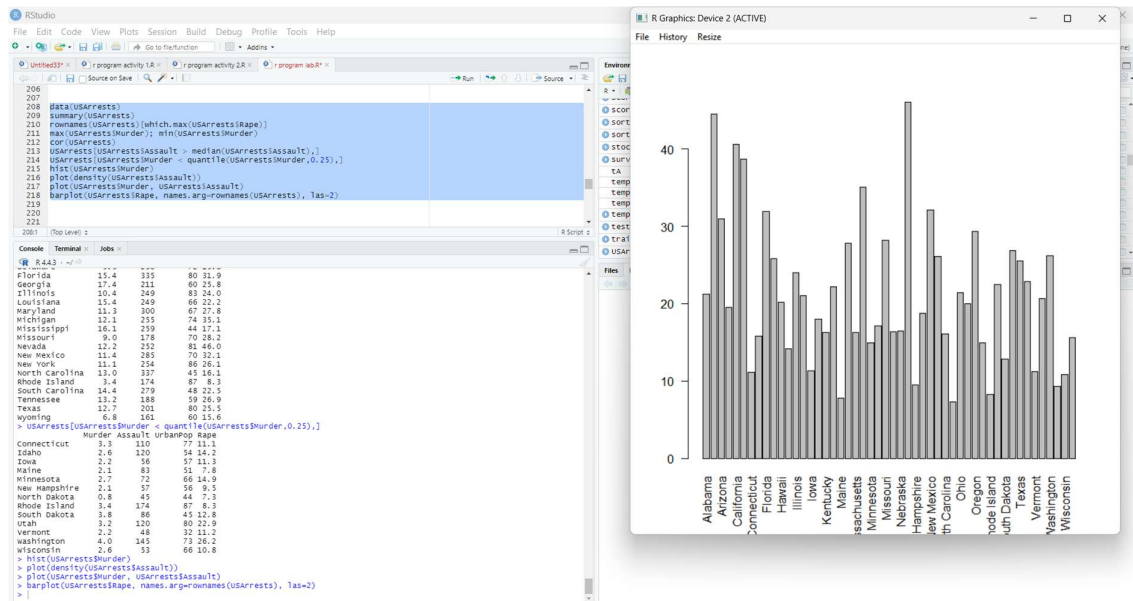
barplot(USArrests$Rape, names.arg=rownames(USArrests), las=2)

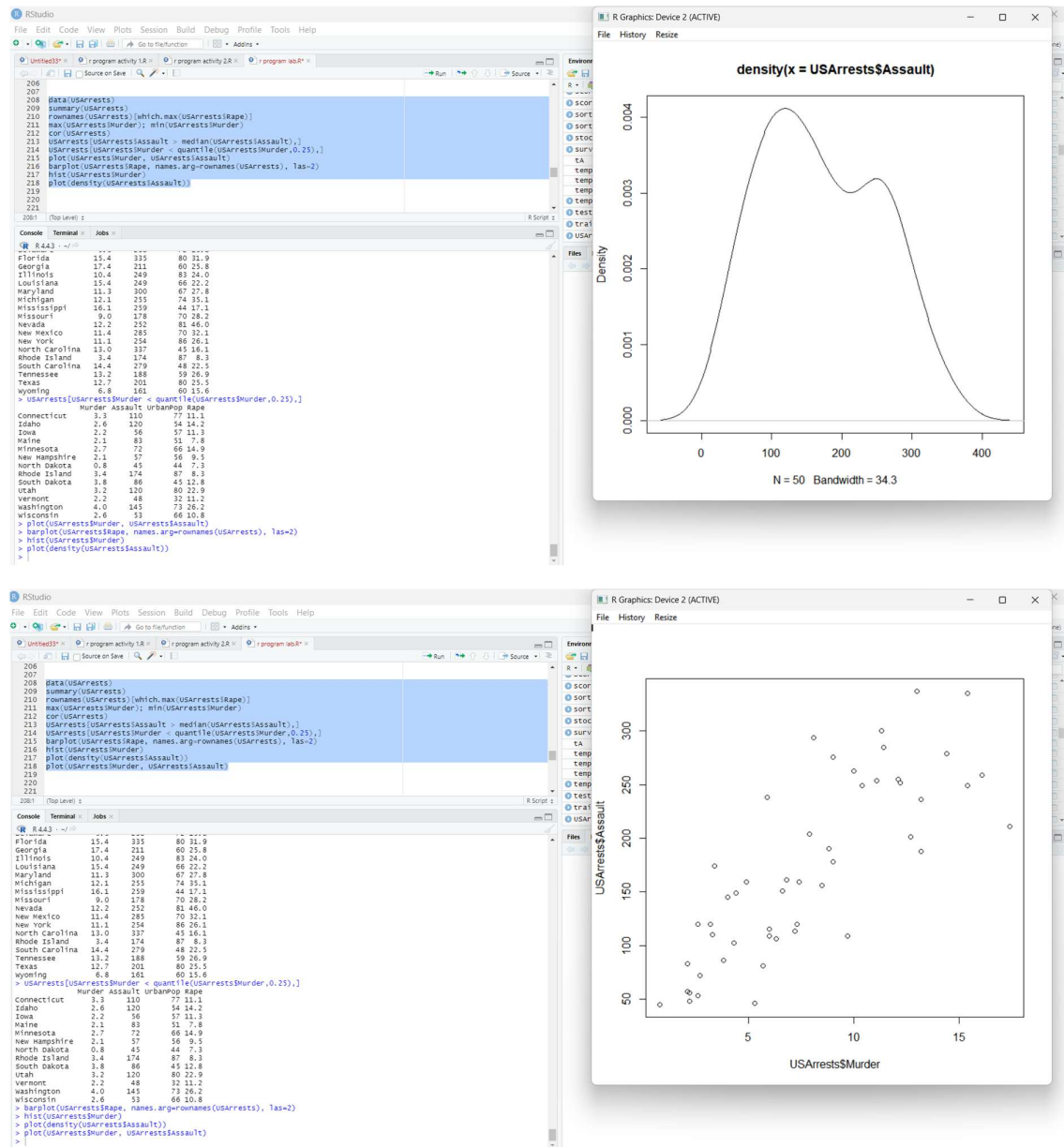
hist(USArrests$Murder)

plot(density(USArrests$Assault))

plot(USArrests$Murder, USArrests$Assault)
```

OUTPUT:





17. Explore Titanic dataset: bar chart of survival vs class, modify plot by gender, histogram of Age.

CODE:

```
data(Titanic)
```

```
titanic_df <- as.data.frame(Titanic)
```

```
barplot(table(titanic_df$Survived, titanic_df$Class),
```

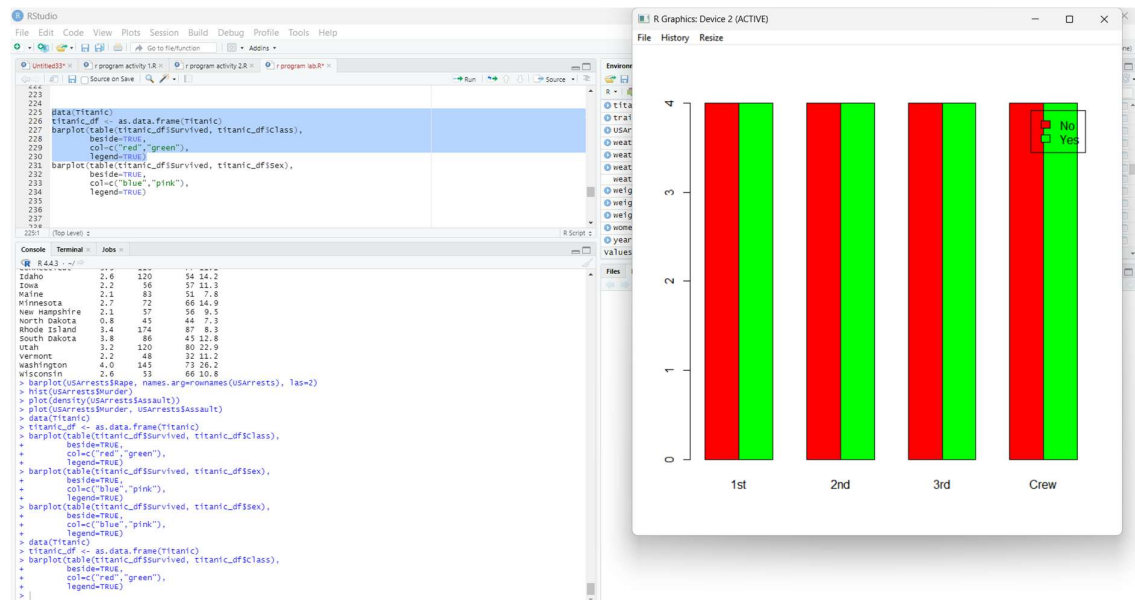
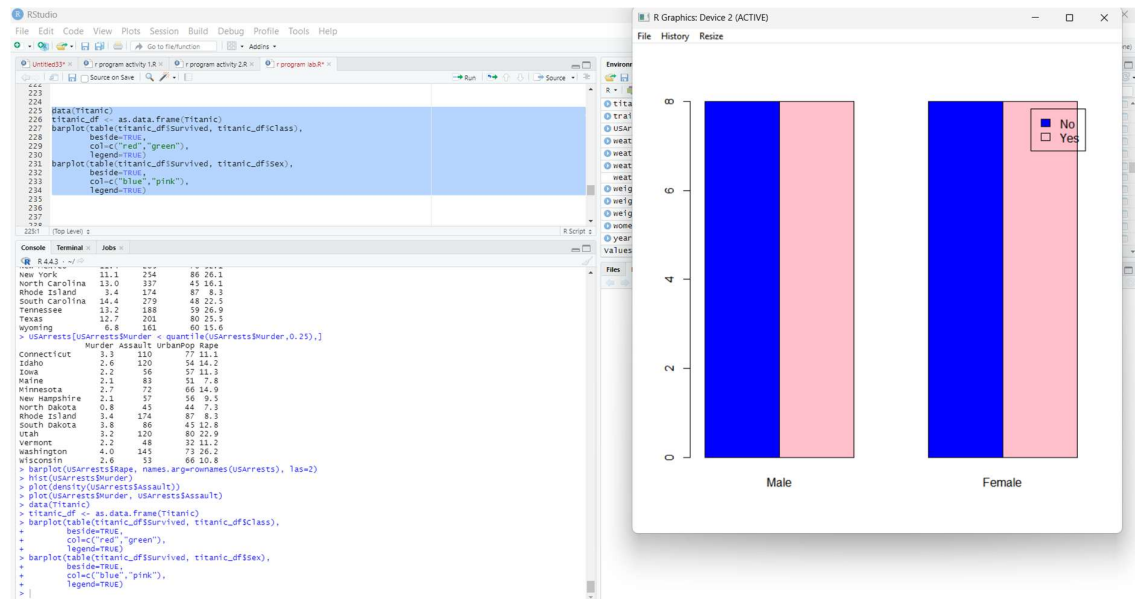
```
beside=TRUE,
```

```
col=c("red", "green"),
```

```
legend=TRUE)
```

```
barplot(table(titanic_df$Survived, titanic_df$Sex),
        beside=TRUE,
        col=c("blue", "pink"),
        legend=TRUE)
```

OUTPUT:



18. Create graphs in R: boxplot, histogram, bar plot, line chart, scatter plot.

CODE:

```
boxplot(iris$Sepal.Length)
```

```
hist(iris$Petal.Length)
```

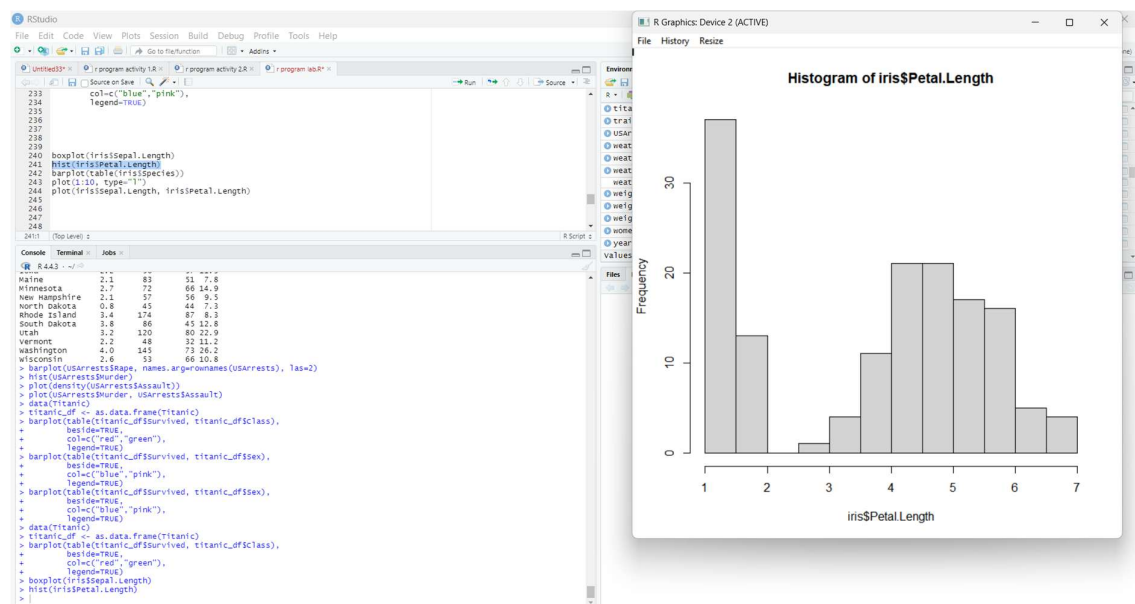
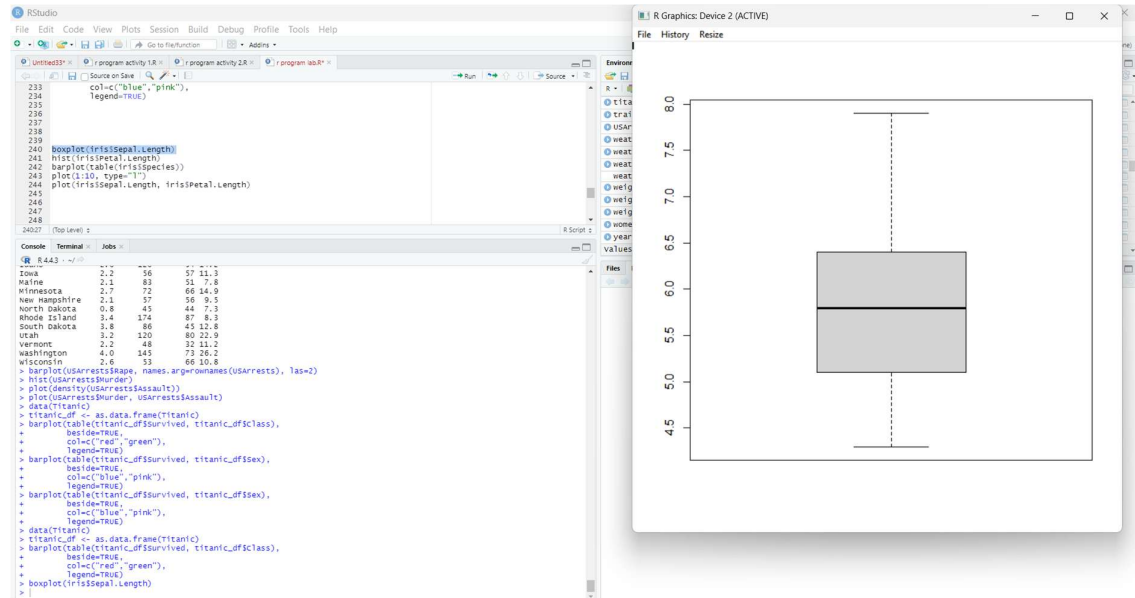


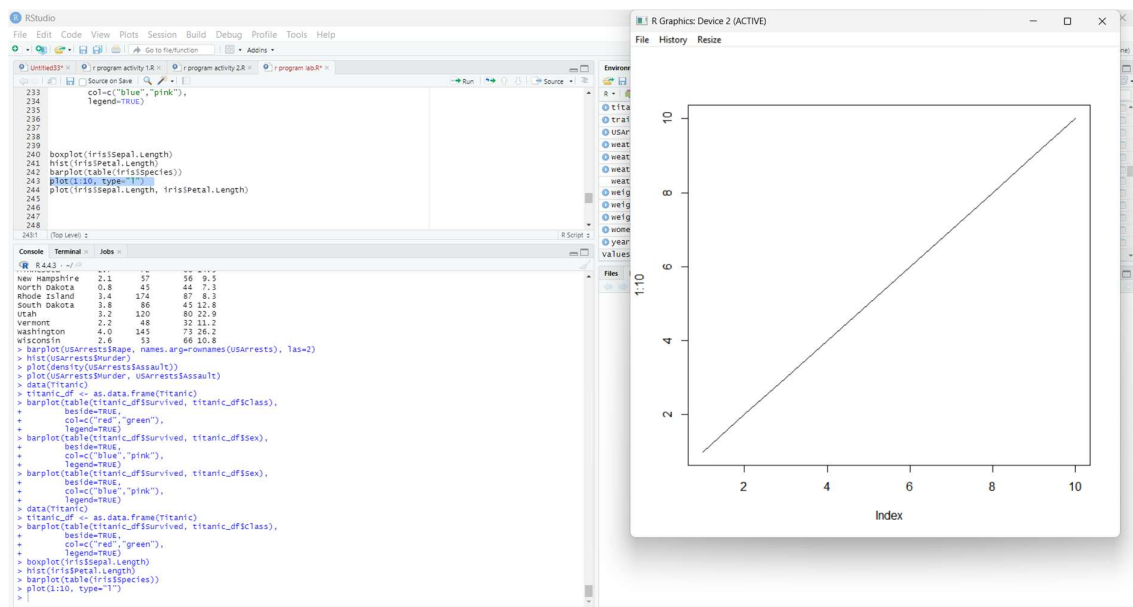
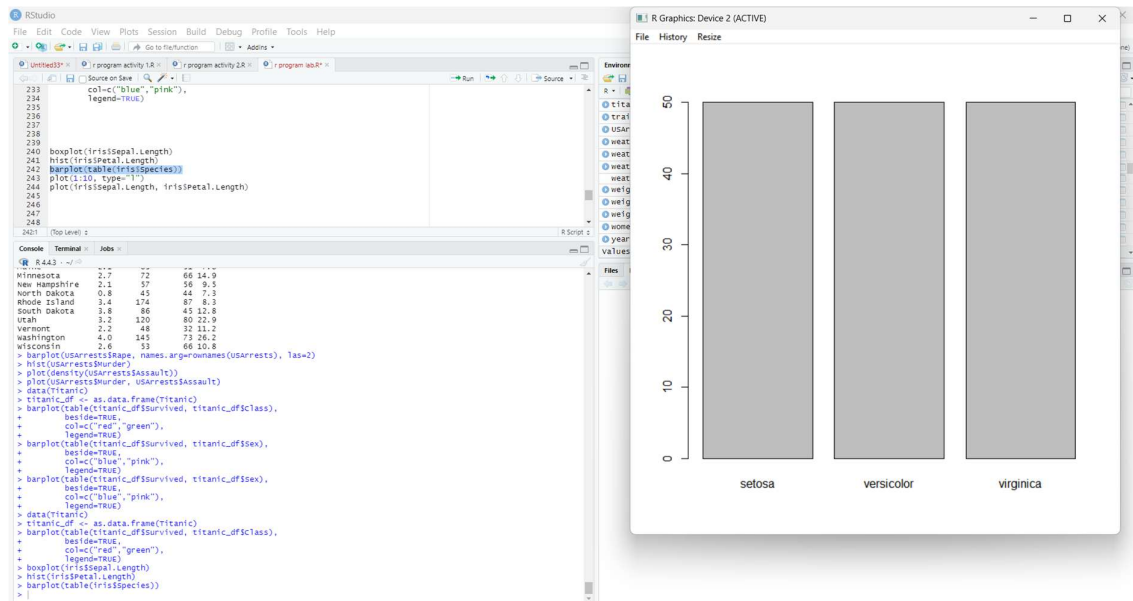
```
barplot(table(iris$Species))
```

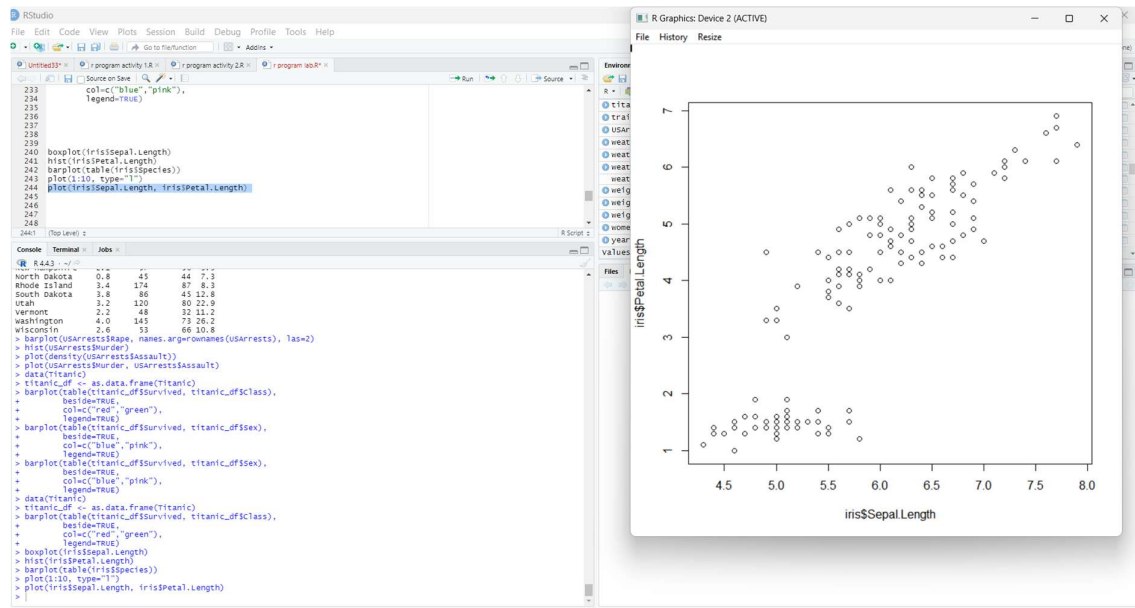
```
plot(1:10, type="l")
```

```
plot(iris$Sepal.Length, iris$Petal.Length)
```

OUTPUT:







19. Build a regression model on advertising dataset (Sales ~ Spend) and predict Sales.

CODE:

```
advertising <- data.frame(
  Spend = c(100,150,200,250,300),
  Sales = c(10,15,22,28,35)
)

model <- lm(Sales ~ Spend, data=advertising)

summary(model)

predict(model, data.frame(Spend=200))
```

OUTPUT:

Petal.Width), predict, and evaluate with confusion matrix

CODE:

```
set.seed(1)

data(iris)

iris2 <- subset(iris, Species != "setosa")

iris2$Species <- factor(iris2$Species)

train_index <- sample(1:nrow(iris2), 0.8*nrow(iris2))

train <- iris2[train_index,]

test <- iris2[-train_index,]

model <- glm(Species ~ Petal.Length + Petal.Width,
             data=train, family=binomial)

prob <- predict(model, test, type="response")

pred <- ifelse(prob > 0.5,
              levels(train$Species)[2],
              levels(train$Species)[1])

table(Predicted=pred, Actual=test$Species)
```

OUTPUT:

The screenshot shows the RStudio interface with the following components:

- Source Editor:** Contains the R code from the previous block, with lines 271-286 highlighted in blue.
- Console:** Displays the output of the code execution, including the glm model summary and the confusion matrix.
- Environment:** Lists the objects in the environment, including 'selection', 'selection_sorted', 'skew_price', 'skew_rating', 'students', 'sub', 'sun_data', 'sun_a', 'tab', 'temp_seq', 'tempA', 'tempB', 'Titanic', and 'train_index'.

Console Output:

```
## R4.3 ##
> set.seed(1)
> data(iris)
> iris2 <- subset(iris, species != "setosa")
> iris2$Species <- factor(iris2$Species)
> train_index <- sample(1:nrow(iris2), 0.8*nrow(iris2))
> train <- iris2[train_index,]
> test <- iris2[-train_index,]
> model <- glm(Species ~ Petal.Length + Petal.Width,
+             data=train, family=binomial)
> prob <- predict(model, test, type="response")
> pred <- ifelse(prob > 0.5,
+               levels(train$Species)[2],
+               levels(train$Species)[1])
> table(Predicted=pred, Actual=test$Species)

      Actual
Predicted versicolor virginica
versicolor    11         0
virginica      0         9
```