Untitled

Nicole De Luna

2023-11-08

```
#1.  Using the for loop, create an R script that will display a 5x5 matrix.

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

matrixA <- matrix(c(0, 0, 0, 0, 0), nrow = 5, ncol = 5)

for (i in 1:5) {
  matrixA[i,] <- abs(vectorA - vectorA[i])
}

print(matrixA)
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    0    1    2    3    4
## [2,]    1    0    1    2    3
## [3,]    2    1    0    1    2
## [4,]    3    2    1    0    1
## [5,]    4    3    2    1    0
```

```
#2. Print the string "*" using for() function.

for(i in 1:5) {
  numb <- rep("*", i)
  print(numb)
}
```

```
## [1] "*"
## [1] "*" "*"
## [1] "*" "*" "*"
## [1] "*" "*" "*" "*"
## [1] "*" "*" "*" "*" "*"
```

```
#3. Fibonacci
start_num <- as.numeric(readline("Enter the starting number for the Fibonacci sequence: "))
```

```
## Enter the starting number for the Fibonacci sequence:
```

```
if (is.na(start_num)) {
  cat("Please enter a valid numeric starting number.\n")
} else {
  num1 <- 0
  num2 <- 1

  repeat {
    if (!is.na(start_num) && num2 >= start_num) {
```

```
      cat(num2, " ")
    }

    fib_sum <- num1 + num2
    num1 <- num2
    num2 <- fib_sum

    if (num2 > 500) {
      break
    }
  }

  cat("\n")
}
```

```
## Please enter a valid numeric starting number.
```

```
#4  Import the dataset as shown in Figure 1 you have created previously.

#4. A  What is the R script for importing an excel or a csv file?
#Display the first 6 rows of the dataset?
#Show your codes and its result.

imprt <- read.csv("HouseholdData.csv")
head(imprt)
```

```
##   X Respondents    Sex FatherOccupation PersonAtHome SiblingsAtSchool
## 1 1           1   Male                1            5                2
## 2 2           2 Female                2            7                3
## 3 3           3 Female                3            3                0
## 4 4           4   Male                3            8                5
## 5 5           5   Male                1            6                2
## 6 6           6 Female                2            4                3
##       HouseType
## 1          Wood
## 2      Congrete
## 3      Congrete
## 4          Wood
## 5 Semi-congrete
## 6 Semi-congrete
```

4.

```
Household <- read.csv("HouseholdData.csv")

# Filter the data based on Gender
males <- Household[Household$Sex == "Male",]
females <- Household[Household$Sex == "Female",]

# Display the results
males
```

```
##     X Respondents  Sex FatherOccupation PersonAtHome SiblingsAtSchool
## 1   1           1 Male                1            5                2
## 4   4           4 Male                3            8                5
## 5   5           5 Male                1            6                2
```

```
## 8   8           8 Male                3            2                2
## 10 10          10 Male                3            6                2
##        HouseType
## 1           Wood
## 4           Wood
## 5  Semi-congrete
## 8  Semi-congrete
## 10      Congrete
```

```
females
```

```
##   X Respondents    Sex FatherOccupation PersonAtHome SiblingsAtSchool
## 2 2           2 Female                2            7                3
## 3 3           3 Female                3            3                0
## 6 6           6 Female                2            4                3
## 7 7           7 Female                2            4                1
## 9 9           9 Female                1           11                6
##       HouseType
## 2      Congrete
## 3      Congrete
## 6 Semi-congrete
## 7          Wood
## 9 Semi-congrete
```

```
# Calculate the number of observations for each gender
f <- nrow(females)
m <- nrow(males)

# Display the number of observations
cat("Number of Female Observations:", f, "\n")
```

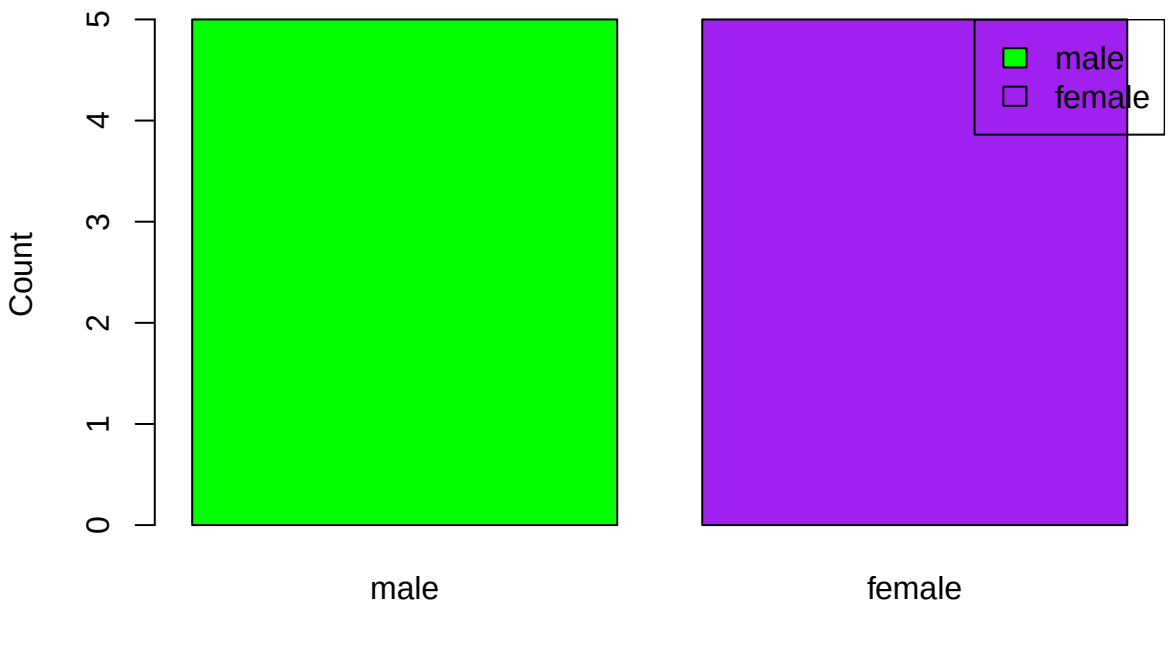
```
## Number of Female Observations: 5
```

```
cat("Number of Male Observations:", m, "\n")
```

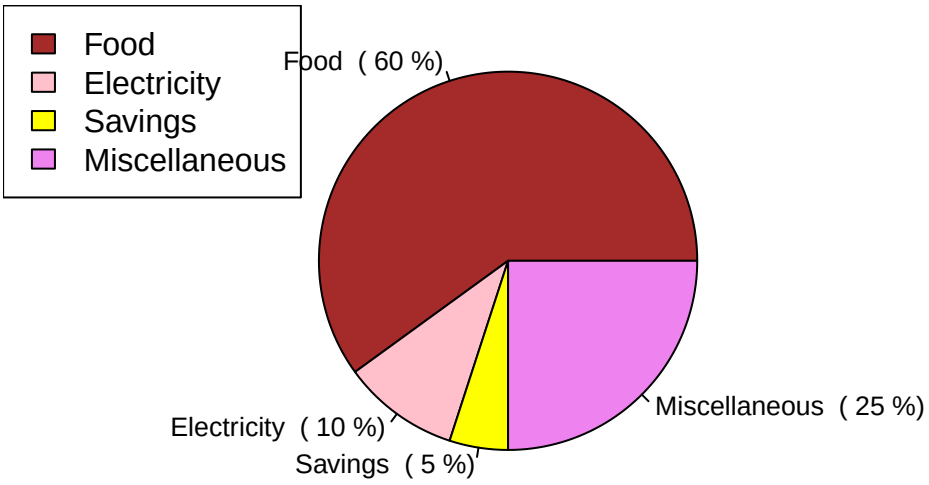
```
## Number of Male Observations: 5
```

#4 c

```
genderCount <- c(male = m, female = f)
#totalFM <- table(Household$Gender)
barplot(genderCount,
main = "Number of Males and Females", xlab = "Gender", ylab = "Count", col = c("green", "purple"))

legend("topright",legend = names(genderCount), fill = c("green", "purple"))
```




```r
#5. The monthly income of Dela Cruz family was spent on the following:

#5. A Create a piechart that will include labels in percentage
#Add some colors and title of the chart. Write the R scripts and show its output.

spend <- data.frame(
  Category = c("Food", "Electricity", "Savings", "Miscellaneous"),
  Value = c(60, 10, 5, 25)
)

spend$Percentage <- spend$Value / sum(spend$Value) * 100
colors <- c("brown", "pink", "yellow", "violet")

# Adjust the font size with the cex parameter
pie(spend$Value,
    labels = paste(spend$Category, " (", spend$Percentage, "%)"),
    col = colors,
    main = "The Monthly Income Spending of Dela Cruz Family",cex = 0.8)

legend("topleft", spend$Category, fill = colors)
```




```
#6. Use the iris dataset.

#6 A. Check for the structure of the dataset using the str() function.
#Describe what youhave seen in the output.

str(iris)
```

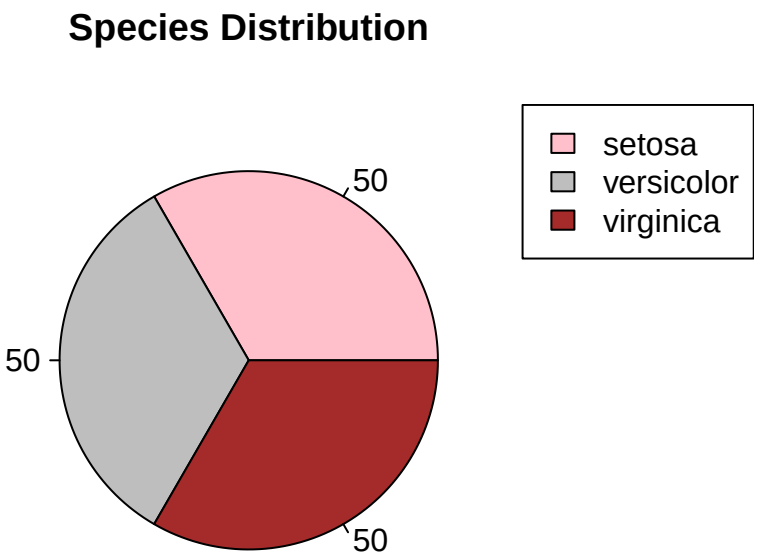
```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
#I was curious and made a research about this.
#The output of the code provides information
#about the iris dataset, which is renowned for its
#use in data analysis and machine learning.
#It reveals the following details:

#1. The dataset comprises 150 observations and 5 variables.
#2. 'Sepal.Length' represents the sepal length of iris flowers.
#3. 'Sepal.Width' represents the sepal width of iris flowers.
#4. 'Petal.Length' signifies the petal length of iris flowers.
#5. 'Petal.Width' signifies the petal width of iris flowers.
#6. 'Species' This is the categorized variables.
```

```
#6 B. Create an R object that will contain the mean of the sepal.length,sepal.width,petal.length,and pe
#What is the R script and its result?

value_of_means <- c(
  Lsepal <- mean(iris$Sepal.Length),
  Wsepal <- mean(iris$Sepal.Width) ,
  Lpetal <- mean(iris$Petal.Length),
  Wpetal <- mean(iris$Petal.Width)
)
value_of_means
```

```
## [1] 5.843333 3.057333 3.758000 1.199333
```

```
#6 C.Create a pie chart for the Species distribution.
#Add title, legends, and colors. Write the R script and its result.

# Assuming that I have a flower called the 'iris' and it  is the name of my dataset
species <- table(iris$Species)
colors <- c("pink", "gray", "brown")

# Create the pie chart
pie(species, col = colors, labels = species )
legend("topright", legend = levels(iris$Species), fill = colors)
title("Species Distribution")
```




```
#6 D. Subset the species into setosa, versicolor, and virginica.
#Write the R scripts and show the last six (6) rows of each species.


#subset

setosa_subset <- iris[iris$Species == "setosa" ,]
versicolor_subset <- iris[iris$Species == "versicolor",]
virginica_subset <- iris[iris$Species == "virginica",]

#last 6 row each

tail(setosa_subset, 6)
```

```
##    Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 45          5.1         3.8          1.9         0.4  setosa
## 46          4.8         3.0          1.4         0.3  setosa
## 47          5.1         3.8          1.6         0.2  setosa
## 48          4.6         3.2          1.4         0.2  setosa
## 49          5.3         3.7          1.5         0.2  setosa
## 50          5.0         3.3          1.4         0.2  setosa
```

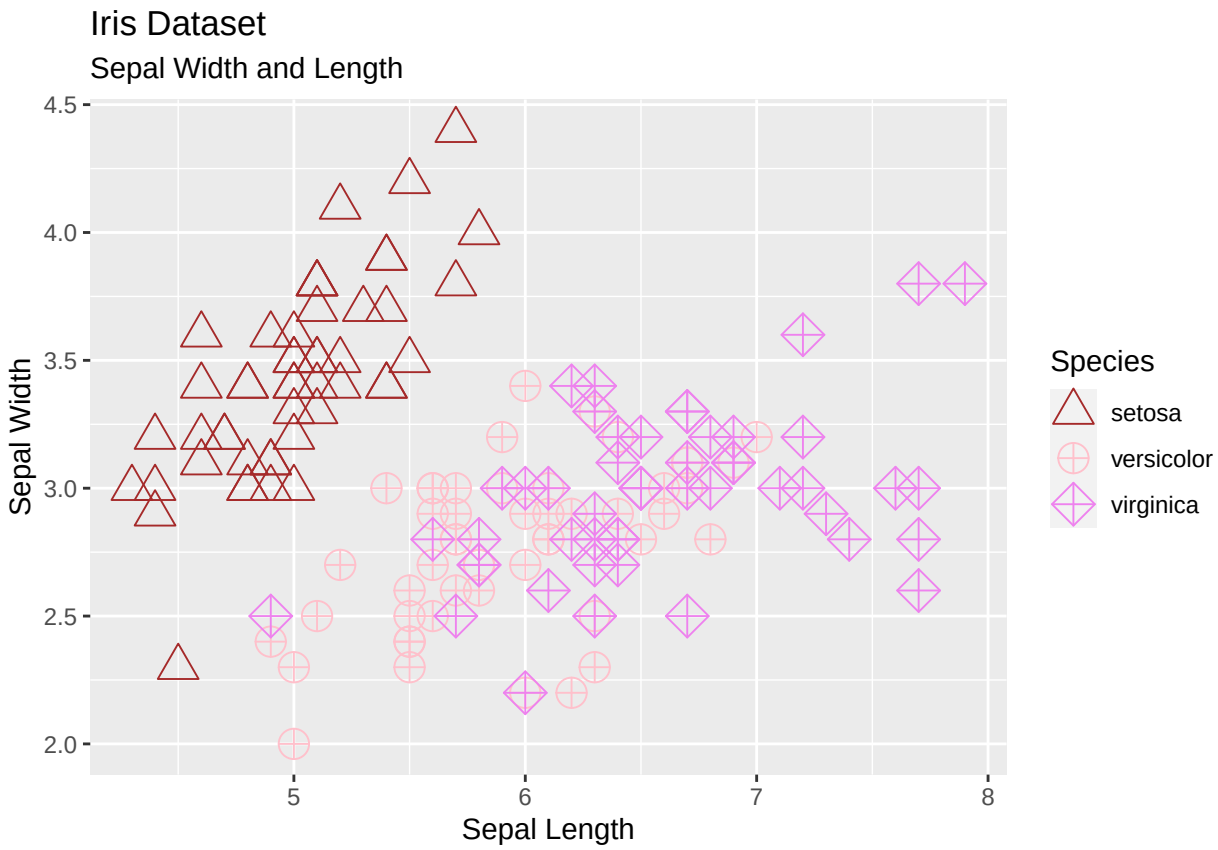
```r
tail(versicolor_subset, 6)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 95           5.6         2.7          4.2         1.3 versicolor
## 96           5.7         3.0          4.2         1.2 versicolor
## 97           5.7         2.9          4.2         1.3 versicolor
## 98           6.2         2.9          4.3         1.3 versicolor
## 99           5.1         2.5          3.0         1.1 versicolor
## 100          5.7         2.8          4.1         1.3 versicolor
```

```r
tail(virginica_subset, 6)
```

```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width   Species
## 145          6.7         3.3          5.7         2.5 virginica
## 146          6.7         3.0          5.2         2.3 virginica
## 147          6.3         2.5          5.0         1.9 virginica
## 148          6.5         3.0          5.2         2.0 virginica
## 149          6.2         3.4          5.4         2.3 virginica
## 150          5.9         3.0          5.1         1.8 virginica
```

```r
#6 E.
#Create a scatterplot of the sepal.length and sepal.width using the differentspecies(setosa,versicolor,
#Add a title = "Iris Dataset",
#subtitle = "Sepal width and length, labels for the x and y axis,
#the pch symbol and colors should be based on the species.

#Hint: Need to convert to factors the species to store categorical variables.


library(ggplot2)

#  factor
iris$Species <- as.factor(iris$Species)

# Create a scatterplot
scatterplot <- ggplot(iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species, shape = Species)) +
  geom_point(size = 5) +
  labs(
    title = "Iris Dataset",
    subtitle = "Sepal Width and Length",
    x = "Sepal Length",
    y = "Sepal Width"
  ) +
  scale_color_manual(values = c("setosa" = "brown", "versicolor" = "pink", "virginica" = "violet")) +
  scale_shape_manual(values = c("setosa" = 2, "versicolor" = 10, "virginica" = 9))

print(scatterplot)
```




```
#6 F. Interpret the result.

#The plot shows a visual representation
#of the Sepal Length and Sepal Width for
#each Iris flower species. Each species is
#represented by a different color and shape.

#Setosa flowers are brown and have a cross-like shape.
#Versicolor flowers are pink and have a circle shape.
#Virginica flowers are violet and have a diamond shape.

#This plot allows you to easily compare and differentiate between
#different species based on their sepal
#length and sepal width, offering a comprehensive
#and visually appealing representation of the data.
```

7

```
library(readxl)
alexa_file <- read_excel("alexa_file.xlsx")
alexa_file
```

```
## # A tibble: 3,150 x 5
##    rating date                variation           verified_reviews      feedback
##     <dbl> <dttm>              <chr>               <chr>                    <dbl>
##  1      5 2018-07-31 00:00:00 Charcoal Fabric     Love my Echo!                1
##  2      5 2018-07-31 00:00:00 Charcoal Fabric     Loved it!                    1
##  3      4 2018-07-31 00:00:00 Walnut Finish       Sometimes while play~        1
```

```
##  4      5 2018-07-31 00:00:00 Charcoal Fabric     I have had a lot of ~        1
##  5      5 2018-07-31 00:00:00 Charcoal Fabric     Music                        1
##  6      5 2018-07-31 00:00:00 Heather Gray Fabric I received the echo ~        1
##  7      3 2018-07-31 00:00:00 Sandstone Fabric    Without having a cel~        1
##  8      5 2018-07-31 00:00:00 Charcoal Fabric     I think this is the ~        1
##  9      5 2018-07-30 00:00:00 Heather Gray Fabric looks great                  1
## 10      5 2018-07-30 00:00:00 Heather Gray Fabric Love it! I've listen~        1
## # i 3,140 more rows
```

```
#7 A. Rename the white and black variants by using gsub() function.

#Rename the variants
alexa_file$variation <- gsub("White Dot", "WhiteDot", alexa_file$variation)
alexa_file$variation <- gsub("White Show", "WhiteShow", alexa_file$variation)
alexa_file$variation <- gsub("White Plus", "WhitePlus", alexa_file$variation)
alexa_file$variation <- gsub("White Spot", "WhiteSpot", alexa_file$variation)

alexa_file$variation <- gsub("Black Dot", "BlacDot", alexa_file$variation)
alexa_file$variation <- gsub("Black Show", "BlackShow", alexa_file$variation)
alexa_file$variation <- gsub("Black Plus", "BlackPlus", alexa_file$variation)
alexa_file$variation <- gsub("Black Spot", "BlackSpot", alexa_file$variation)

alexa_file
```

```
## # A tibble: 3,150 x 5
##    rating date                variation           verified_reviews      feedback
##     <dbl> <dttm>              <chr>               <chr>                    <dbl>
##  1      5 2018-07-31 00:00:00 Charcoal Fabric     Love my Echo!                1
##  2      5 2018-07-31 00:00:00 Charcoal Fabric     Loved it!                    1
##  3      4 2018-07-31 00:00:00 Walnut Finish       Sometimes while play~        1
##  4      5 2018-07-31 00:00:00 Charcoal Fabric     I have had a lot of ~        1
##  5      5 2018-07-31 00:00:00 Charcoal Fabric     Music                        1
##  6      5 2018-07-31 00:00:00 Heather Gray Fabric I received the echo ~        1
##  7      3 2018-07-31 00:00:00 Sandstone Fabric    Without having a cel~        1
##  8      5 2018-07-31 00:00:00 Charcoal Fabric     I think this is the ~        1
##  9      5 2018-07-30 00:00:00 Heather Gray Fabric looks great                  1
## 10      5 2018-07-30 00:00:00 Heather Gray Fabric Love it! I've listen~        1
## # i 3,140 more rows
```

```
#7 B. Get the total number of each variations and save it into another object.
#Save the object as variations.RData.
#Write the R scripts. What is its result?


library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

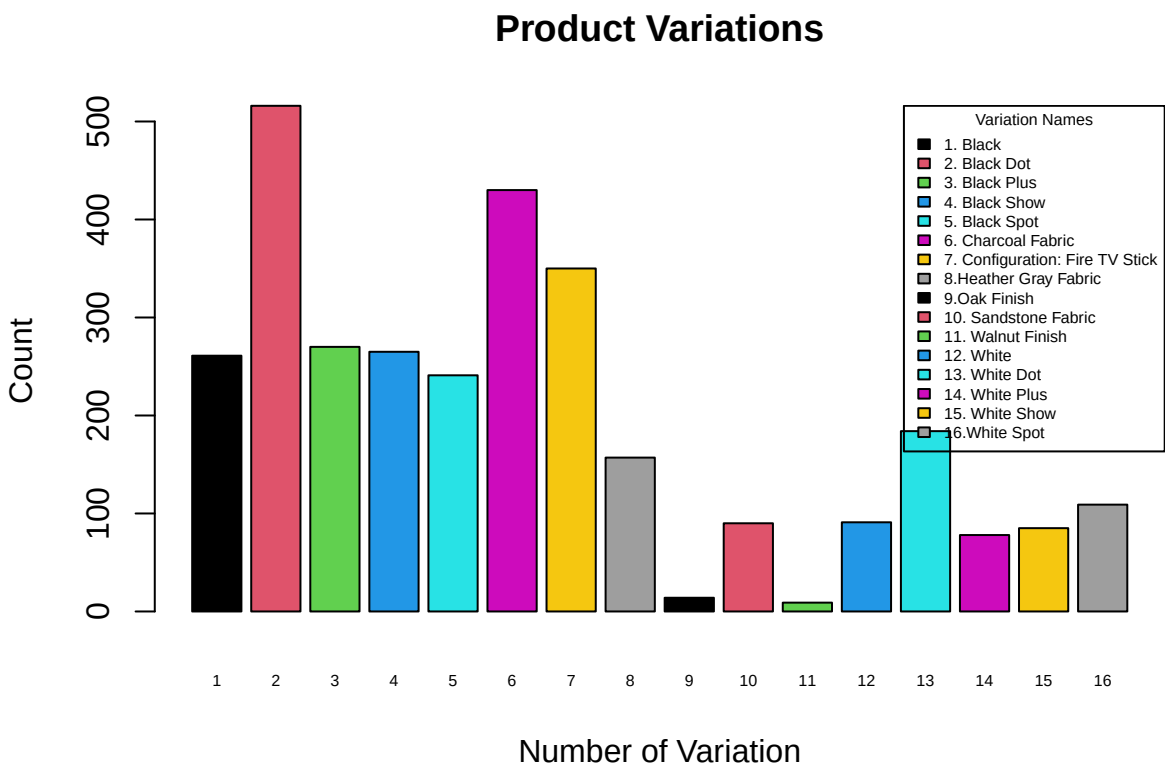
```r
# Perform the desired operations
variations <- alexa_file %>%
  count(alexa_file$variation)


variations
```

```
## # A tibble: 16 x 2
##    `alexa_file$variation`           n
##    <chr>                        <int>
##  1 Black                          261
##  2 Black  Dot                     516
##  3 Black  Plus                    270
##  4 Black  Show                    265
##  5 Black  Spot                    241
##  6 Charcoal Fabric                430
##  7 Configuration: Fire TV Stick   350
##  8 Heather Gray Fabric            157
##  9 Oak Finish                      14
## 10 Sandstone Fabric                90
## 11 Walnut Finish                    9
## 12 White                           91
## 13 White  Dot                     184
## 14 White  Plus                     78
## 15 White  Show                     85
## 16 White  Spot                    109
```

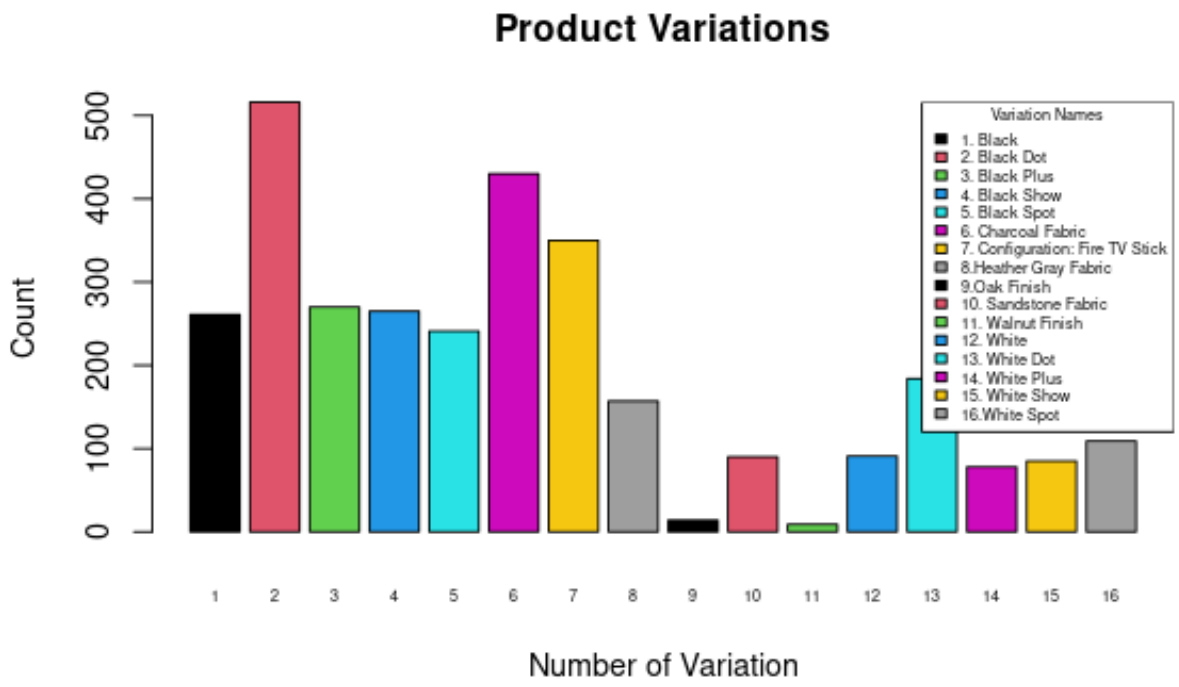
```r
save(variations, file = "variations.RData")
```

```r
#7 C. From the variations.RData, create a barplot().
#Complete the details of the chart
#which include the title, color, labels of each bar.

# Load the variations data
load("variations.RData")

# Extract the variation names
Vnames <- c(
  "1. Black", "2. Black Dot", "3. Black Plus", "4. Black Show",
  "5. Black Spot", "6. Charcoal Fabric", "7. Configuration: Fire TV Stick",
  "8.Heather Gray Fabric", "9.Oak Finish", "10. Sandstone Fabric",
  "11. Walnut Finish", "12. White", "13. White Dot", "14. White Plus", "15. White Show", "16.White Spot
)

# Barplot
CompletePlot <- barplot(variations$n,
  names.arg = 1:16,
  col = 1:16,
  main = "Product Variations",
  xlab = "Number of Variation",
  ylab = "Count",
  las = 0.0,
  cex.names = 0.5,
  space = 0.2
)
```

```
# Add legend
legend("topright", legend = Vnames,
       fill = 1:16,
       title = "Variation Names", cex = 0.5)
```




```
knitr::include_graphics("/cloud/project/worksheet#4.rmd/Worksheet#4b/vari.png")
```



#7 d.

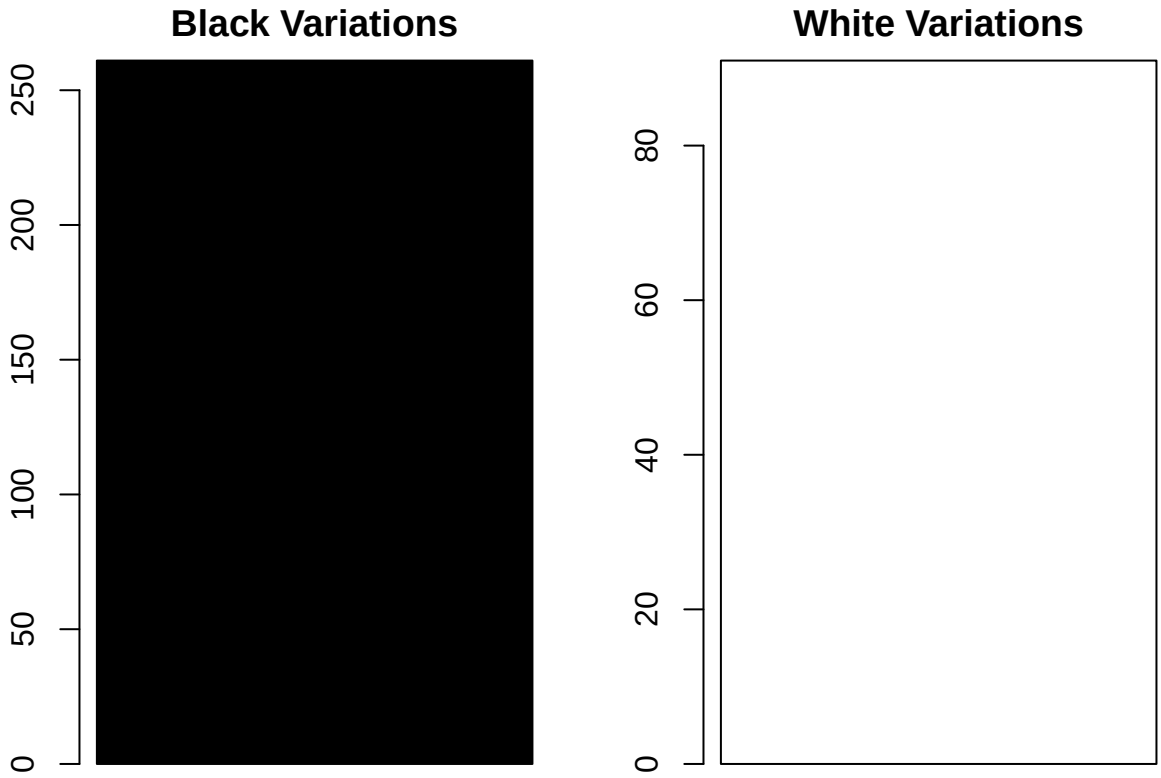
```
 Blackplot <- variations[variations$`alexa_file$variation` %in% c("Black", "BlackDot", "BlackShow", "Bla

  Whiteplot <- variations[variations$`alexa_file$variation` %in% c("White", "WhiteDot", "WhiteShow", "Wh


  par(mfrow = c(1, 2), mar = c(2, 2, 2, 2))

  Black <- barplot(height = Blackplot$n,
                   names.arg = Blackplot$`alexa$variation`,
                   col = "Black",
                   main = "Black Variations",
                   xlab = "Number of Variation",
                   ylab = "Count",
                   las = 0.2,
                   cex.names = 0.4,
                   space = 0.2
)
```

```
## Warning: Unknown or uninitialised column: `alexa$variation`.
```

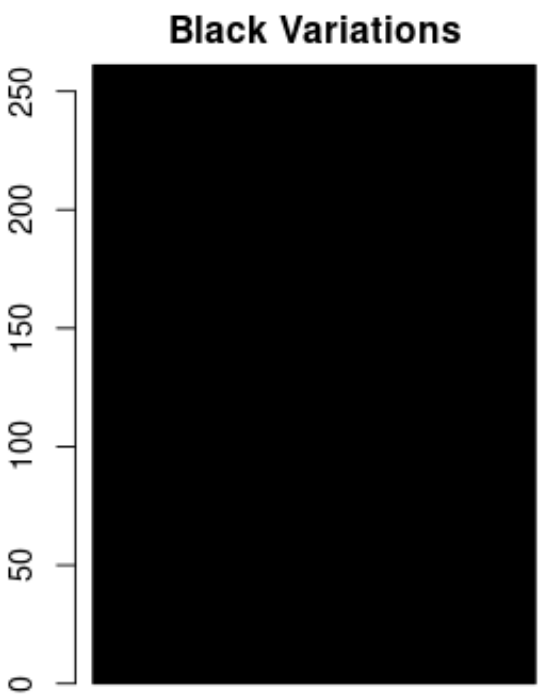
```
  # Barplot for White variations
  White <- barplot(height = Whiteplot$n,
                   names.arg = Whiteplot$`alexa$variation`,
                   col = "White",
                   main = "White Variations",
                   xlab = "Number of Variation",
                   ylab = "Count",
                   las = 0.2,
```

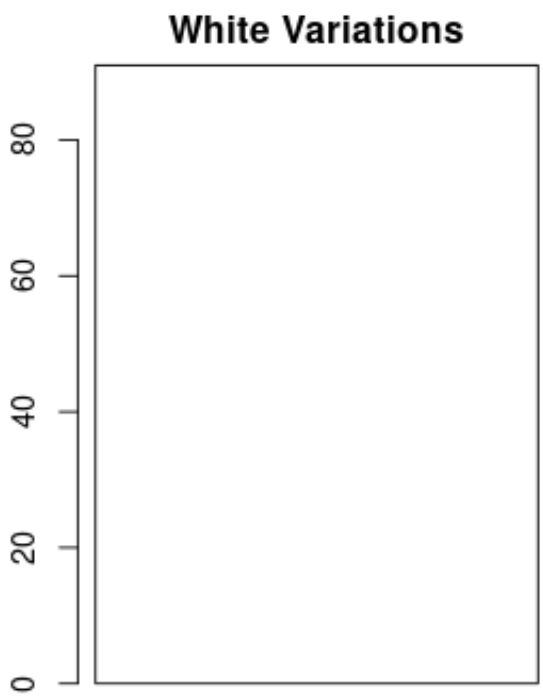
```
                cex.names = 0.4,
                space = 0.2
)
```

```
## Warning: Unknown or uninitialised column: `alexa$variation`.
```




```
knitr::include_graphics("/cloud/project/worksheet#4.rmd/Worksheet#4b/bw.png")
```


Black Variations
0
50
100
150
200
250


White Variations
0
20
40
60
80