RWork-sheet_jalando-on#4b

Ralyn Queen Jalando-on

2024-10-30

USING FOR LOOP FUNCTION

1. Using the for loop, create an R script that will display a 5x5 matrix as shown in Figure 1. It must contain vectorA = [1,2,3,4,5] and a 5 x 5 zero matrix.

```
matrixVA <- matrix(0, nrow = 5, ncol = 5)
vectorA <- c(1, 2, 3, 4, 5)

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

print(matrixVA)
```

```
##      [,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. The output should be the same as shown in Figure

```
rows <- 5

for (i in 1:rows) {
  cat(rep("''*''", i), "\n")
}
```

```
## ''*''
## ''*'' ''*''
## ''*'' ''*'' ''*''
## ''*'' ''*'' ''*'' ''*''
## ''*'' ''*'' ''*'' ''*'' ''*''
```

3. Get an input from the user to print the Fibonacci sequence starting from the 1st input up to 500. Use repeat and break statements. Write the R Scripts and its output.

```
start <- 10
a <- 0
b <- 1
repeat {
  newfibonacci <- a + b
  if (newfibonacci > 500) {
    break
  }
  if (newfibonacci >= start) {
    cat(newfibonacci, "\n")
  }
  a <- b
  b <- newfibonacci
}
```

```
## 13
## 21
## 34
## 55
## 89
## 144
## 233
## 377
```

USING BASIC GRAPHICS (plot(),barplot(),pie(),hist())

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

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

```
library(readr)
data <- read_csv("/cloud/project/shoesizes.csv")
```

```
## New names:
## Rows: 14 Columns: 6
## -- Column specification
## -------------------------------------------------------- Delimiter: "," chr
## (2): Gender...3, Gender...6 dbl (4): Shoe size...1, Height...2, Shoe size...4,
## Height...5
## i Use `spec()` to retrieve the full column specification for this data. i
## Specify the column types or set `show_col_types = FALSE` to quiet this message.
## * `Shoe size` -> `Shoe size...1`
## * `Height` -> `Height...2`
## * `Gender` -> `Gender...3`
## * `Shoe size` -> `Shoe size...4`
## * `Height` -> `Height...5`
## * `Gender` -> `Gender...6`
```

```
data
```

```
## # A tibble: 14 x 6
##    `Shoe size...1` Height...2 Gender...3 `Shoe size...4` Height...5 Gender...6
##              <dbl>      <dbl> <chr>                <dbl>      <dbl> <chr>
##  1             6.5         66 F                     13           77 M
##  2             9           68 F                     11.5         72 M
##  3             8.5         64 F                      8.5         59 F
##  4             8.5         65 F                      5           62 F
##  5            10.5         70 M                     10           72 M
##  6             7           64 F                      6.5         66 F
##  7             9.5         70 F                      7.5         64 F
##  8             9           71 F                      8.5         67 M
##  9            13           72 M                     10.5         73 M
## 10          7.5       64 F                 8.5       69 F
## 11         10.5       74 M                10.5       72 M
## 12          8.5       67 F                11         70 M
## 13         12         71 M                 9         69 M
## 14         10.5       71 M                13         70 M
```

b. Create a subset for gender(female and male). How many observations are there in Male? How about in Female? Write the R scripts and its output.

```r
df <- data.frame(
  Shoe_size = c(6.5, 9.0, 8.5, 8.5, 10.5, 7.0, 9.5, 9.0, 13.0, 7.5, 10.5, 8.5, 12.0, 10.5,
              13.0, 11.5, 8.5, 5.0, 10.0, 6.5, 7.5, 8.5, 10.5, 8.5, 10.5, 11.0, 9.0, 13.0),
  Height = c(66.0, 68.0, 64.5, 65.0, 70.0, 64.0, 70.0, 71.0, 72.0, 64.0, 74.5, 67.0, 71.0, 71.0,
             77.0, 72.0, 59.0, 62.0, 72.0, 66.0, 64.0, 67.0, 73.0, 69.0, 72.0, 70.0, 69.0, 70.0),
  Gender = c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M", "F", "M", "M",
             "M", "M", "F", "F", "M", "F", "F", "M", "M", "F", "M", "M", "M", "M")
)

female_subset <- subset(df, Gender == "F")
male_subset <- subset(df, Gender == "M")

num_females <- nrow(female_subset)
num_males <- nrow(male_subset)

cat("Number of observations in Female:", num_females, "\n")
```

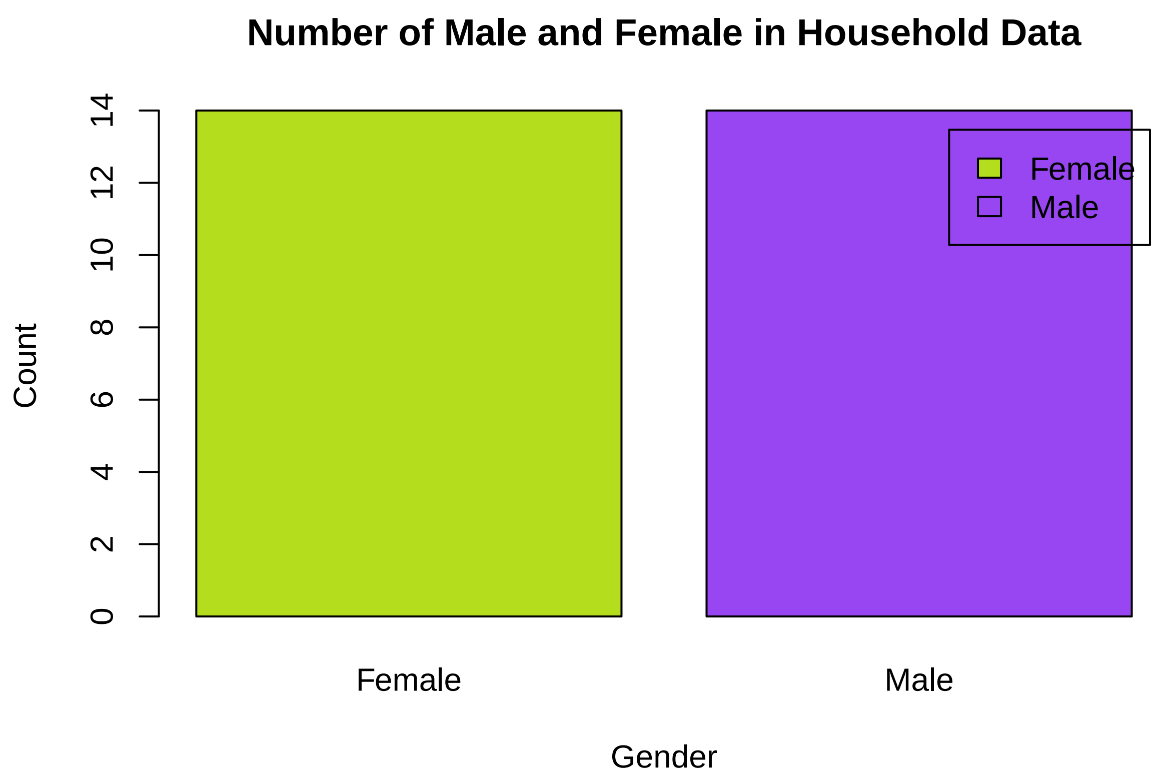
```
## Number of observations in Female: 14
```

```r
cat("Number of observations in Male:", num_males, "\n")
```

```
## Number of observations in Male: 14
```

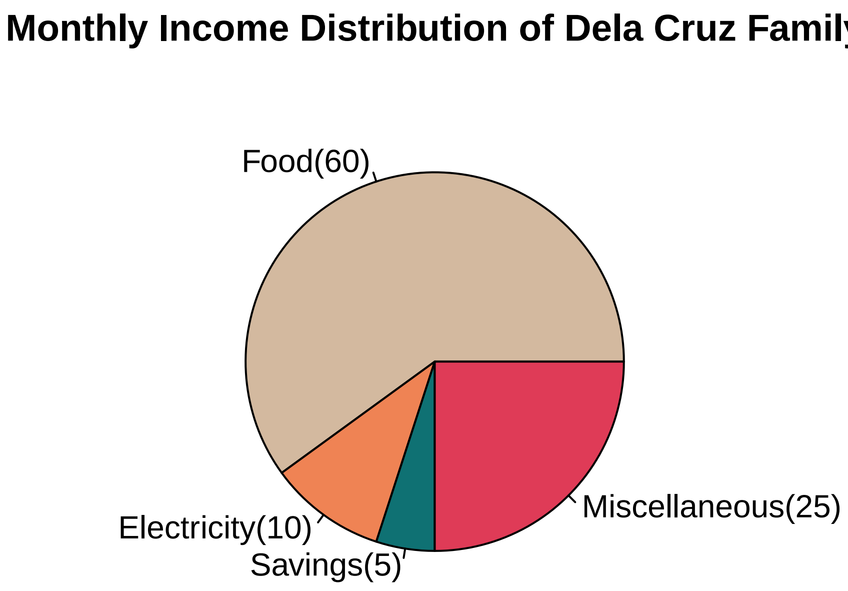
c. Create a graph for the number of males and females for Household Data. Use plot(), chart type = barplot. Make sure to place title, legends, and colors. Write the R scripts and its result.

```r
gender_counts <- c(Female = 14, Male = 14)

barplot(
  gender_counts,
  main = "Number of Male and Female in Household Data",
  xlab = "Gender",
  ylab = "Count",
  col = c("#b4dd1e", "#9746f2"),
  legend.text = c("Female", "Male"),
  beside = TRUE
)
```



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

```
categories <- c("Food", "Electricity", "Savings", "Miscellaneous")
values <- c(60, 10, 5, 25)

pie_chart <- pie(values,
                  labels = paste(categories, "(", values, ")", sep = ""),
                  col = c("#D3B99F", "#EF8354", "#0F7173", "#DF3B57"),
                  main = "Monthly Income Distribution of Dela Cruz Family")

percentages <- round(values / sum(values) * 100, 1)
text(0, 5, paste(percentages, "%"), cex = 1.2, pos = 3)
```



6. Use the iris dataset. data(iris)

a. Check for the structure of the dataset using the str() function. Describe what you have seen in the output.

```r
data(iris)
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 ...
```

The iris dataset is a data frame with 150 obs and 5 columns. Four of these columns contain numeric measurements: Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width, which describe the physical features of iris flowers. The fifth column, Species, is a category that includes three types of iris: "setosa," "versicolor," and "virginica."

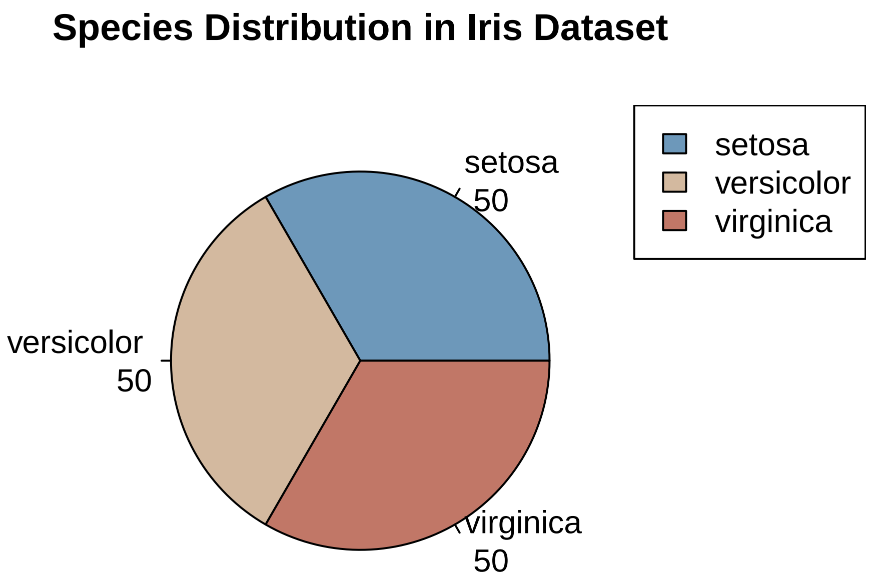
b. Create an R object that will contain the mean of the sepal.length, sepal.width,petal.length,and petal.width. What is the R script and its result?

```r
mean_values <- colMeans(iris[, 1:4])
mean_values
```

```
## Sepal.Length  Sepal.Width Petal.Length  Petal.Width 
##     5.843333     3.057333     3.758000     1.199333
```

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

```r
data(iris)

species_counts <- table(iris$Species)

pie(species_counts,
    main = "Species Distribution in Iris Dataset",
    col = c("#6d98ba", "#D3B99F", "#C17767"),
    labels = paste(names(species_counts), "\n", species_counts))
legend("topright", legend = names(species_counts), fill = c("#6d98ba", "#D3B99F", "#C17767"))
```



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

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

last_six_setosa <- tail(setosa_subset, 6)
last_six_versicolor <- tail(versicolor_subset, 6)
last_six_virginica <- tail(virginica_subset, 6)

last_six_setosa
```

```
##    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
```

```
last_six_versicolor
```

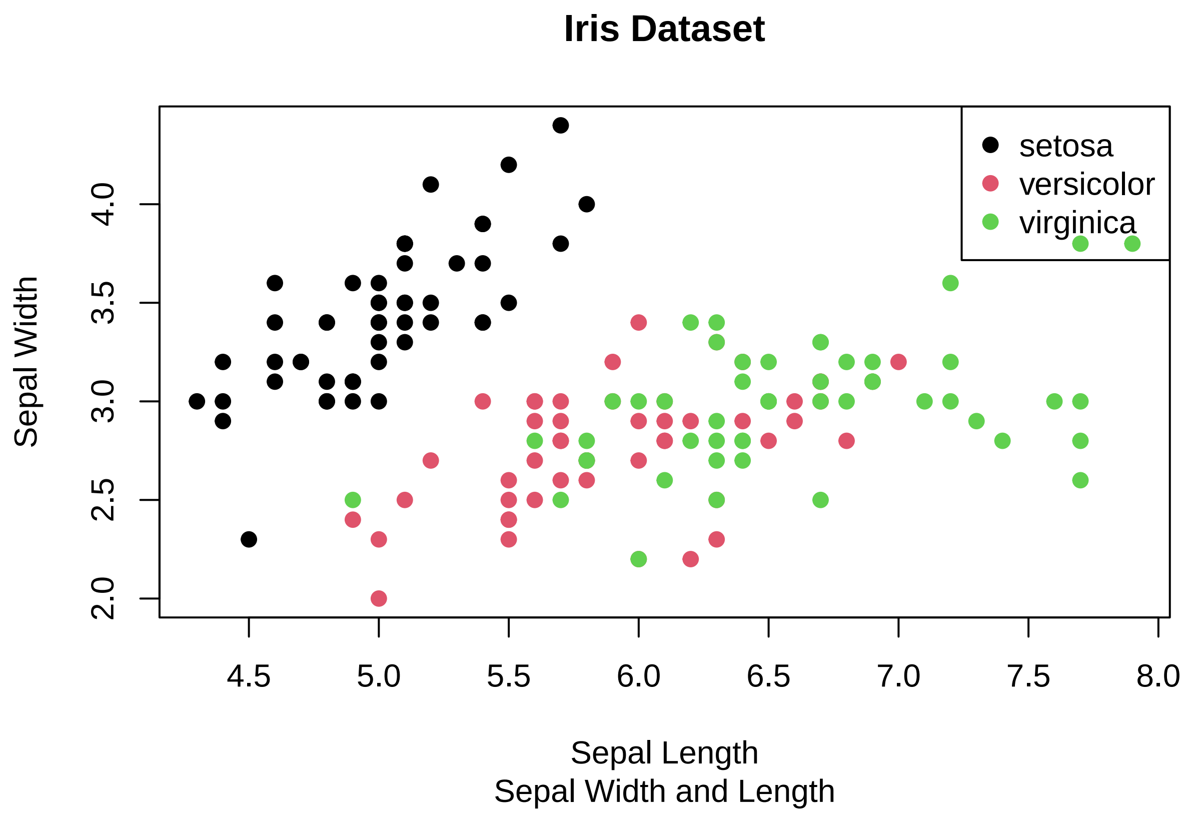
```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 95           5.6         2.7          4.2         1.3 versicolor
## 96           5.7         3.0          4.2         1.2 versicolor
## 97           5.7         2.9          4.2         1.3 versicolor
## 98           6.2         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
```

```
last_six_virginica
```

```
##     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
```

e. Create a scatterplot of the sepal.length and sepal.width using the different species(setosa,versicolor,virginica). 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.

```r
plot(iris$Sepal.Length, iris$Sepal.Width,
     main = "Iris Dataset",
     sub = "Sepal Width and Length",
     xlab = "Sepal Length",
     ylab = "Sepal Width",
     pch = 19,
     col = iris$Species)

legend("topright", legend = levels(iris$Species),
       col = 1:3, pch = 19)
```



f. Interpret the result. The pie chart shows how many of each iris species are in the dataset. Each slice represents the size of each group that determine which species is the most common. The subsets display the last six entries for each species, allowing us to look at their specific measurements, like sepal and petal sizes, and understand how they differ. The scatterplot illustrates the relationship between sepal length and width. If setosa clusters at lower measurements, it means this species tends to have smaller flowers, while if versicolor and virginica overlap, it suggests their sizes are similar, making it harder to tell them apart.

BASIC CLEANING AND TRANSFORMATION OF OBJECTS

7. Import the alexa-file.xlsx. Check on the variations. Notice that there are ex- tra whitespaces among black variants (Black Dot, Black Plus, Black Show, Black

Spot). Also on the white variants (White Dot, White Plus, White Show, White Spot).

a. Rename the white and black variants by using gsub() function.

```r
library(readxl)
alexa_data <- read_excel("/cloud/project/alexa_file.xlsx")
unique(alexa_data$variation)
```

```
##  [1] "Charcoal Fabric"              "Walnut Finish"
##  [3] "Heather Gray Fabric"          "Sandstone Fabric"
##  [5] "Oak Finish"                   "Black"
##  [7] "White"                        "Black  Spot"
##  [9] "White  Spot"                  "Black  Show"
## [11] "White  Show"                  "Black  Plus"
## [13] "White  Plus"                  "Configuration: Fire TV Stick"
## [15] "Black  Dot"                   "White  Dot"
```

```r
alexa_data$variation <- gsub("Black Dot", "BlackDot", alexa_data$variation)
alexa_data$variation <- gsub("Black Plus", "BlackPlus", alexa_data$variation)
alexa_data$variation <- gsub("Black Show", "BlackShow", alexa_data$variation)
alexa_data$variation <- gsub("Black Spot", "BlackSpot", alexa_data$variation)
alexa_data$variation <- gsub("White Dot", "WhiteDot", alexa_data$variation)
alexa_data$variation <- gsub("White Plus", "WhitePlus", alexa_data$variation)
alexa_data$variation <- gsub("White Show", "WhiteShow", alexa_data$variation)
alexa_data$variation <- gsub("White Spot", "WhiteSpot", alexa_data$variation)
unique(alexa_data$variation)
```

```
##  [1] "Charcoal Fabric"              "Walnut Finish"
##  [3] "Heather Gray Fabric"          "Sandstone Fabric"
##  [5] "Oak Finish"                   "Black"
##  [7] "White"                        "Black  Spot"
##  [9] "White  Spot"                  "Black  Show"
## [11] "White  Show"                  "Black  Plus"
## [13] "White  Plus"                  "Configuration: Fire TV Stick"
## [15] "Black  Dot"                   "White  Dot"
```

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? Hint: Use the dplyr package. Make sure to install it before loading the package.

```r
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
variations <- alexa_data %>%
  count(variation)
print(variations)
```

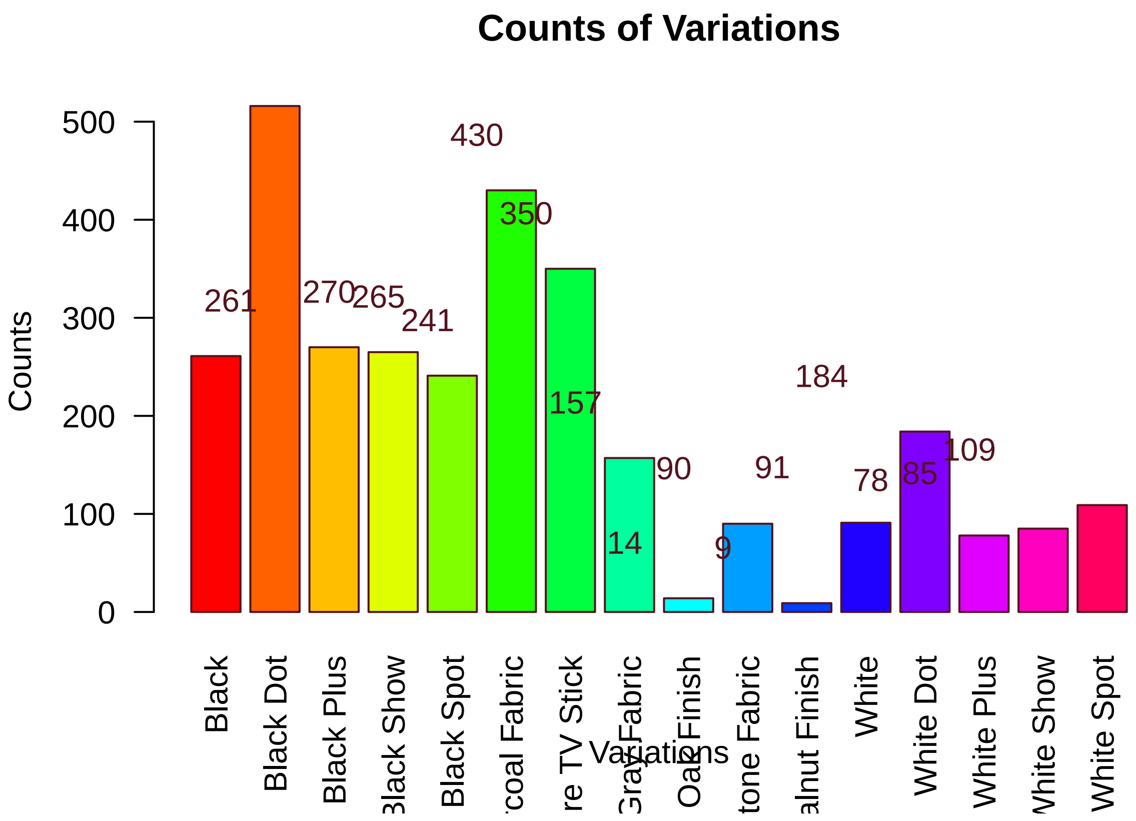
```
## # A tibble: 16 x 2
##    variation                        n
##    <chr>                        <int>
##  1 Black                          261
##  2 Black  Dot                     516
##  3 Black  Plus                    270
##  4 Black  Show                    265
##  5 Black  Spot                    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")
```

SAMPLE OUTPUT

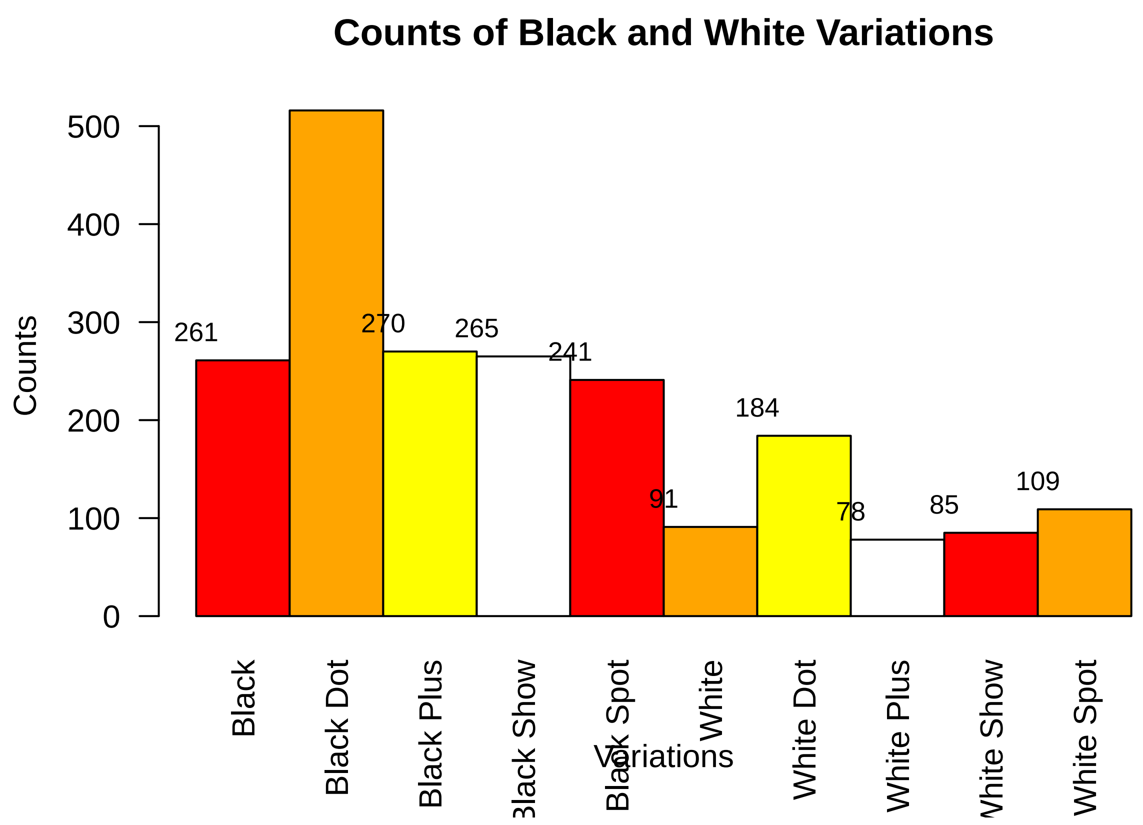
c. From the variations.RData, create a barplot(). Complete the details of the chart which include the title, color, labels of each bar.

```r
library(dplyr)

load("variations.RData")
variations$variation <- gsub(" +", " ", variations$variation)
variations$variation <- trimws(variations$variation)
bardata <- variations$n
barnames <- variations$variation

barplot(
  bardata,
  main = "Counts of Variations",
  col = rainbow(length(bardata)),
  names.arg = barnames,
  xlab = "Variations",
  ylab = "Counts",
  las = 2,
  border = "#53131e"
)

text(
  x = seq_along(bardata),
  y = bardata + max(bardata) * 0.05,
  labels = bardata,
  pos = 3,
  cex = 1,
  col = "#53131e"
)
```



d. Create a barplot() for the black and white variations. Plot it in 1 frame, side by side. Complete the details of the chart.

```
library(ggplot2)
library(dplyr)
load("variations.RData")
variations$variation <- gsub(" +", " ", variations$variation)
variations$variation <- trimws(variations$variation)
bwvariations <- variations %>%
  filter(grepl("Black|White", variation))
bardata <- as.matrix(bwvariations$n)
barnames <- bwvariations$variation
barplot(
  bardata,
  beside = TRUE,
  main = "Counts of Black and White Variations",
  col = c("red", "orange", "yellow", "white"),
  names.arg = barnames,
  xlab = "Variations",
  ylab = "Counts",
  las = 2,
  border = "black"
)
text(x = seq_along(bardata), y = bardata, labels = bardata, pos = 3, cex = 0.8, col = "black")
```


Counts of Black and White Variations
Counts
0
100
200
300
400
500
261
270
265
241
91
184
78
85
109
Black
Black Dot
Black Plus
Black Show
Black Spot
White
White Dot
White Plus
White Show
White Spot
Variations