

RWorkSheet#4b

ANGELO A. GARRIDO

2025-11-13

#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.

```
numbers <- matrix(0, nrow = 5, ncol = 5)
```

```
# fill the matrix using for loops
```

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

```
##      [,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*

```
for (i in 1:5) {  
  for (j in 1:i) {  
    cat(' "*" ' )  
  }  
  cat("\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.

```
num <- 10
```

```
firstnum <- 0  
secnum <- 1
```

```

fib_seq <- c()

repeat {
  fib <- firstnum + secnum
  firstnum <- secnum
  secnum <- fib

  if (fib >= num) {
    fib_seq <- c(fib_seq, fib)
  }

  if (fib > 500) {
    break
  }
}

print("Fibonacci sequence from 10 to 500:")

```

```
## [1] "Fibonacci sequence from 10 to 500:"
```

```
print(fib_seq)
```

```
## [1] 13 21 34 55 89 144 233 377 610
```

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

```
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
            ,68.0,70.0)
```

```
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"
            )
```

```
shoesize_Data <- data.frame(size,height,gender)
names(shoesize_Data) <- c("Size","Height","Gender")
shoesize_Data
```

```
##      Size Height Gender
## 1    6.5   66.0      F
## 2    9.0   68.0      F
## 3    8.5   64.5      F
## 4    8.5   65.0      F
## 5   10.5   70.0      M
## 6    7.0   64.0      F
## 7    9.5   70.0      F
## 8    9.0   71.0      F
## 9   13.0   72.0      M
## 10   7.5   64.0      F
## 11  10.5   74.5      M
## 12   8.5   67.0      F
## 13  12.0   71.0      M
```

```
## 14 10.5 71.0 M
## 15 13.0 77.0 M
## 16 11.5 72.0 M
## 17 8.5 59.0 F
## 18 5.0 62.0 F
## 19 10.0 72.0 M
## 20 6.5 66.0 F
## 21 7.5 64.0 F
## 22 8.5 67.0 M
## 23 10.5 73.0 M
## 24 8.5 69.0 F
## 25 10.5 72.0 M
## 26 11.0 70.0 M
## 27 9.0 68.0 M
## 28 13.0 70.0 M
```

#a. Import CSV or Excel and display first 6 rows

```
data <- read.csv("import_march (1).csv")
head(data)
```

```
## Students Students.1 Students.2 Students.3
## 1 Male 8 10 8
## 2 4 8 6
## 3 0 6 4
## 4 Female 14 4 15
## 5 10 2 12
## 6 6 0 9
```

#b. Subset for gender and count observations

```
sub_male <- shoesize_Data[shoesize_Data$Gender == "M", c("Size", "Height")]
sub_female <- shoesize_Data[shoesize_Data$Gender == "F", c("Size", "Height")]
```

```
nrow(sub_male)
```

```
## [1] 14
```

```
nrow(sub_female)
```

```
## [1] 14
```

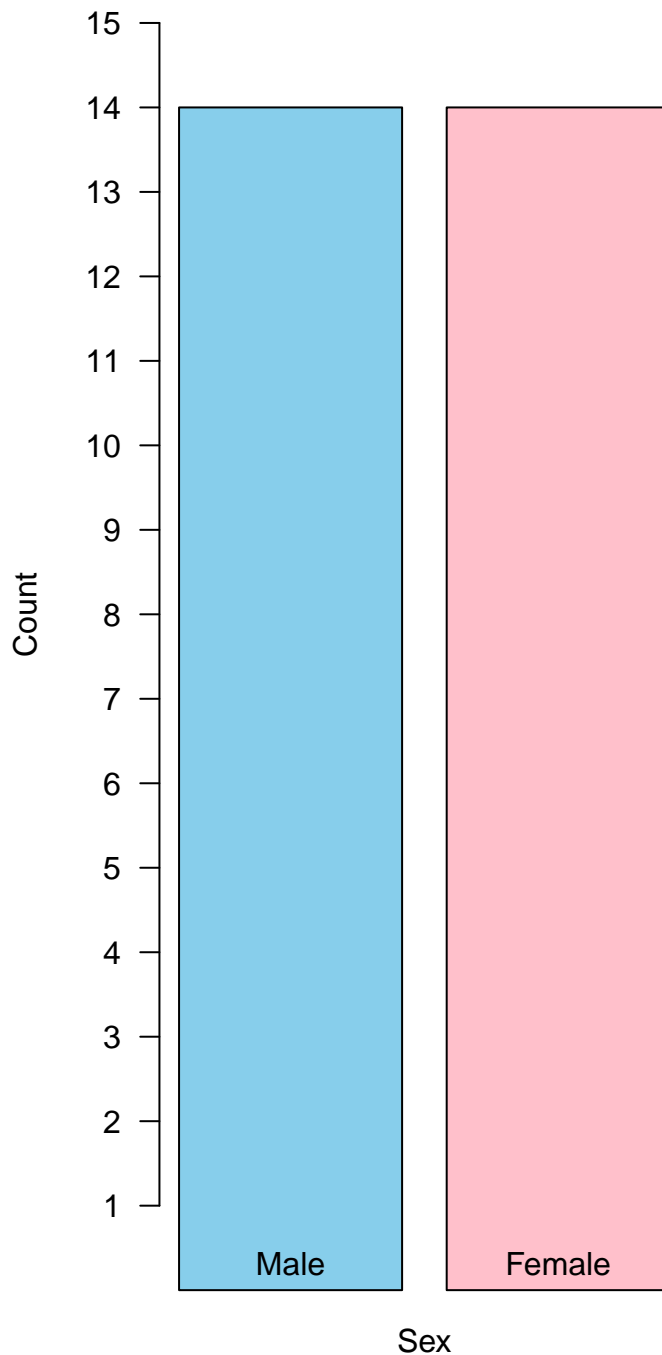
#c. Barplot of males and females

```
gender_counts <- c(Male = nrow(sub_male), Female = nrow(sub_female))
```

```
barplot(gender_counts,
  main = "Number of Males and Females in a Shoe Data Set",
  xlab = "Sex",
  ylab = "Count",
  col = c("skyblue", "pink"),
  width = 0.09,
  ylim = c(1,15),
  yaxt = "n")
```

```
axis(2, at = seq(1, 15, by = 1), las = 1)
```

Number of Males and Females in a Shoe Dat



#Output: Barplot with blue for Male, red for Female, shows genders

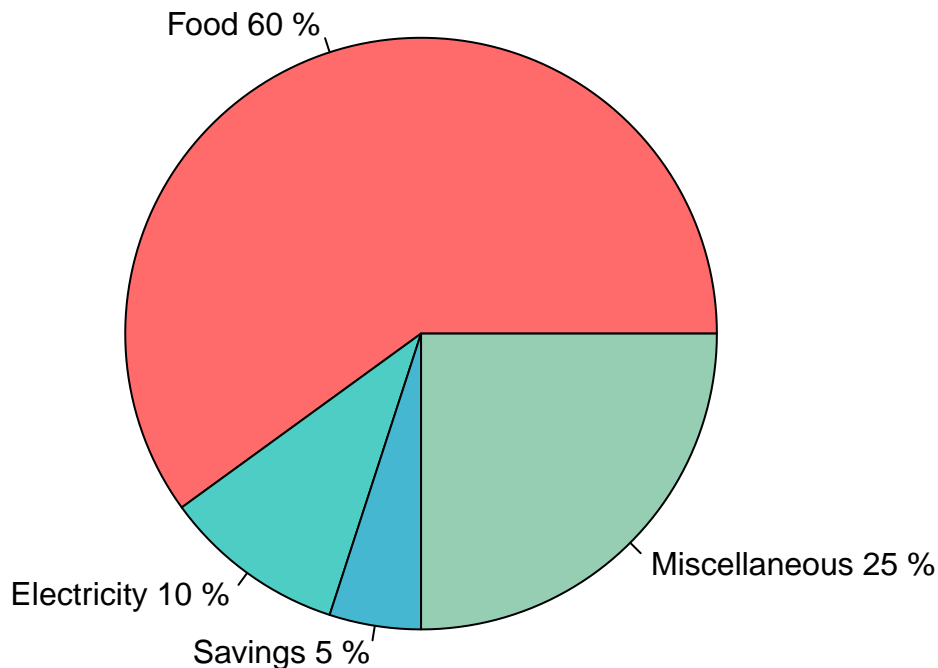
#5. The monthly income of Dela Cruz family was spent on the following:

#a. Create a piechart that will include labels in percentage

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

pie(expenses,
  labels=paste(names(expenses), round(expenses/sum(expenses)*100,1), "%"),
  col = c("#FF6B6B", "#4ECDC4", "#45B7D1", "#96CEB4"),
  main="Monthly Expenses of Dela Cruz Family")
```

Monthly Expenses of Dela Cruz Family



#Output: Pie chart with percentages of Food, Electricity, Savings, Miscellaneous

#6. Use the iris dataset.

```
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 ...
```

#Output:

```
## 'data.frame': 150 obs. of 5 variables
## $ Sepal.Length: num
## $ Sepal.Width : num
## $ Petal.Length: num
## $ Petal.Width : num
```

```
## Species      : Factor w/ 3 levels "setosa","versicolor","virginica"
```

```
#b. Calculate mean of numeric columns
```

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

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

```
# Output:
```

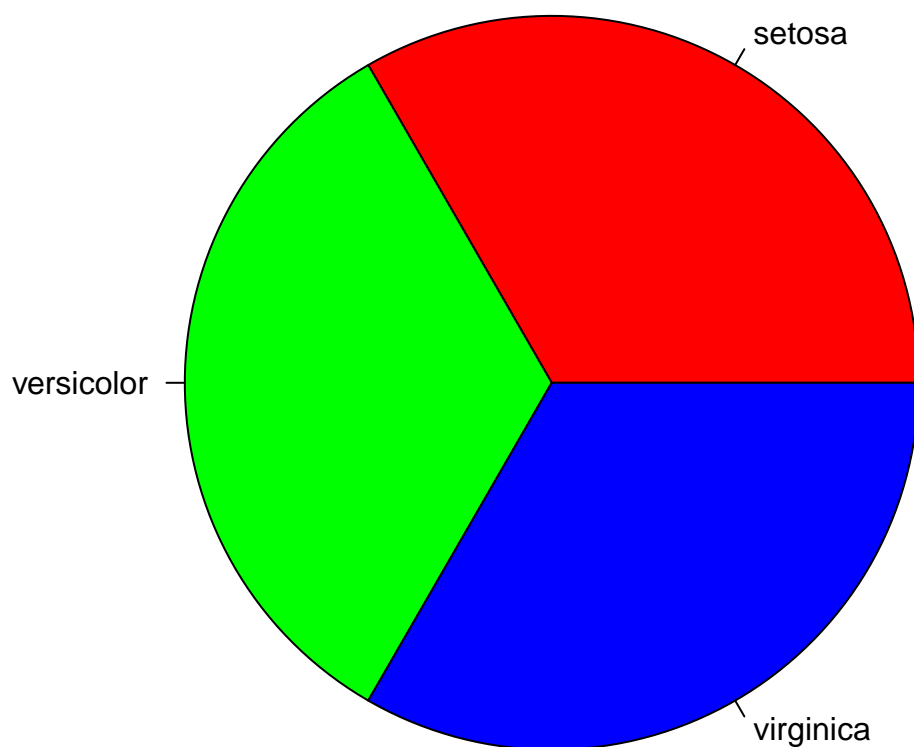
```
# Sepal.Length Sepal.Width Petal.Length Petal.Width  
#      5.843      3.057      3.758      1.199
```

```
#c. Pie chart of Species distribution
```

```
species_count <- table(iris$Species)  
pie(species_count,  
     main="Species Distribution in Iris Dataset",  
     col=c("red","green","blue"),  
     legend.text=TRUE)
```

```
## Warning in text.default(1.1 * P$x, 1.1 * P$y, labels[i], xpd = TRUE, adj =  
## ifelse(P$x < : "legend.text" is not a graphical parameter  
## Warning in text.default(1.1 * P$x, 1.1 * P$y, labels[i], xpd = TRUE, adj =  
## ifelse(P$x < : "legend.text" is not a graphical parameter  
## Warning in text.default(1.1 * P$x, 1.1 * P$y, labels[i], xpd = TRUE, adj =  
## ifelse(P$x < : "legend.text" is not a graphical parameter  
## Warning in title(main = main, ...): "legend.text" is not a graphical parameter
```

Species Distribution in Iris Dataset



#Output: Pie chart showing number of setosa, versicolor, virginica

#d. Subset species and show last 6 rows

```
setosa_iris <- iris[iris$Species=="setosa", ]  
versicolor_iris <- iris[iris$Species=="versicolor", ]  
virginica_iris <- iris[iris$Species=="virginica", ]
```

```
tail(setosa_iris)
```

```
##      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
```

Output: last 6 rows of setosa

```
tail(versicolor_iris)
```

```
##      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
```

Output: last 6 rows of versicolor

```
tail(virginica_iris)
```

```
##      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
```

Output: last 6 rows of virginica

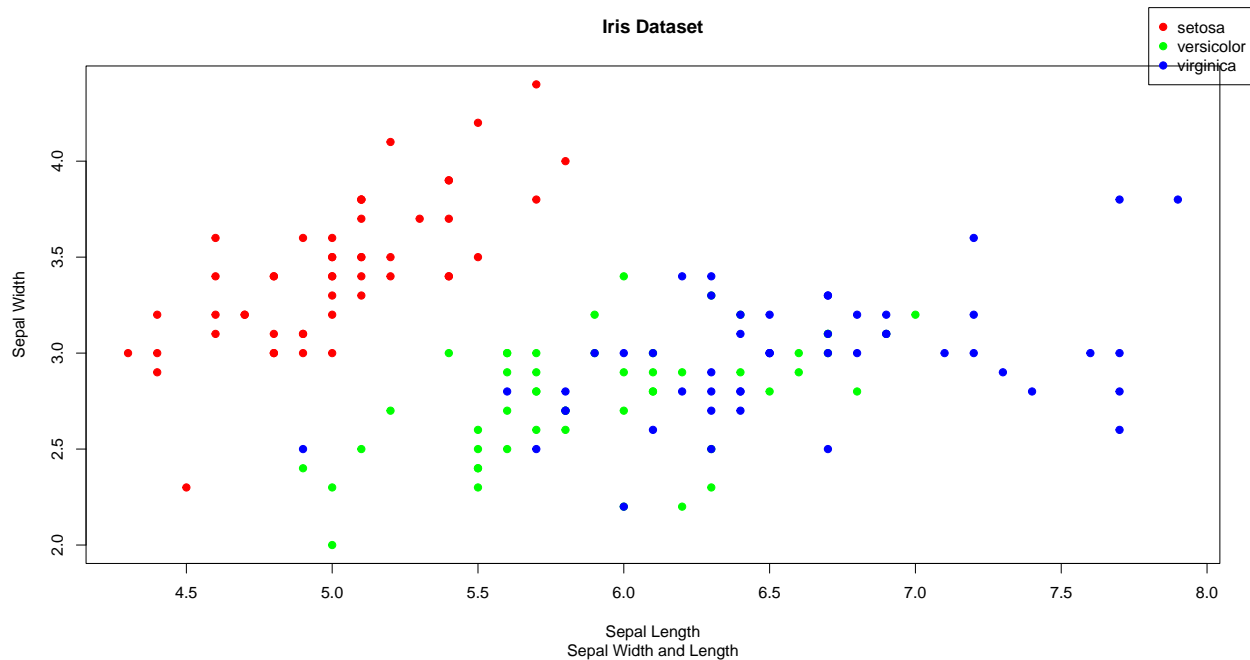
#e. Scatterplot of Sepal.Length vs Sepal.Width by Species

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

```
colors <- c("red","green","blue")
```

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

```
legend(x = 7.8, y = 4.8,      # Upper right coordinates
      legend = levels(iris$Species),
      col = colors,
      pch = 19,
      xpd = TRUE)
```



```
# Output: Scatterplot with red=setosa, green=versicolor, blue=virginica
```

```
#f. Interpret the result.
```

```
#Interpretation: Setosa is compact, Virginica is long/lean, and Versicolor
```

```
#comfortably sits in between.
```

```
#7. Import the alexa-file.xlsx
```

```
library(readxl)
```

```
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
```

```
# Read the Excel file
```

```
alexa_xlsx <- read_excel("alexa-file.xlsx")
```

```
# Check the structure and column names
```

```
str(alexa_xlsx)
```

```
## tibble [26 x 5] (S3: tbl_df/tbl/data.frame)
```

```
## $ rating          : chr [1:26] "5" "5" "5" "5" ...
```

```
## $ date            : chr [1:26] "2018-08-03" "2018-08-01" "2024-11-14" "2018-07-30" ...
```

```
## $ variation       : chr [1:26] "White Show" "White Dot" "black" "White Spot" ...
```

```
## $ verified_reviews: chr [1:26] "N/A" "Perfect for my bedside table." "Quick setup" "Love it!" ...
```

```
## $ feedback        : chr [1:26] "1" "1" "1" "1" ...
```

```

colnames(alexa_xlsx)

## [1] "rating"          "date"            "variation"        "verified_reviews"
## [5] "feedback"

#a. Rename the white and black variants by using gsub() function.
# Remove extra whitespaces in black and white variants
alexa_xlsx$variation <- gsub("Black\\s+", "Black ", alexa_xlsx$variation)
alexa_xlsx$variation <- gsub("White\\s+", "White ", alexa_xlsx$variation)

# Snippet of the cleaned variation column
head(alexa_xlsx$variation)

## [1] "White Show" "White Dot"  "black"      "White Spot" "Black Show"
## [6] "Black Spot"

# Show snippet of cleaned variations (you would replace this with your actual image path)
# knitr::include_graphics("path/to/your/snippet/image.png")

#b. Get the total number of each variations and save it into another object.
variations <- alexa_xlsx %>% count(variation)
variations

## # A tibble: 10 x 2
##   variation      n
##   <chr>      <int>
## 1 Black Dot      4
## 2 Black Plus     1
## 3 Black Show     1
## 4 Black Spot     1
## 5 White Dot      3
## 6 White Plus     3
## 7 White Show     2
## 8 White Spot     3
## 9 black          4
## 10 white         4

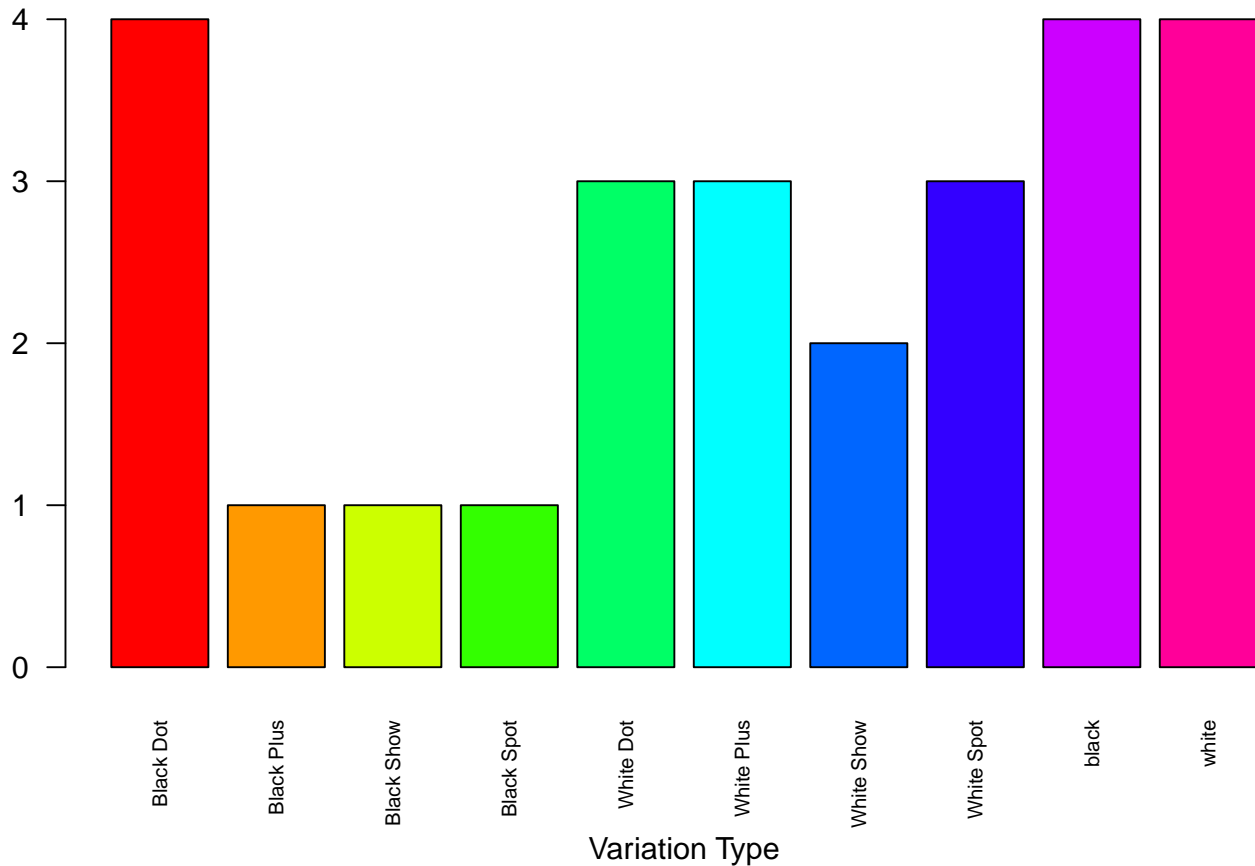
# Save the object as variations.RData
save(variations, file = "variations.RData")

#c. From the variations.RData, create a barplot()
load("variations.RData")

# Create barplot with all variations
barplot(variations$n,
        names.arg = variations$variation,
        col = rainbow(length(variations$n)),
        main = "Total Number of Alexa Variations",
        xlab = "Variation Type",
        ylab = "Total Count",
        las = 2,
        cex.names = 0.7,
        ylim = c(-0.1, max(variations$n) * 1.15),
        mgp = c(4, 1, 0))

```

Total Number of Alexa Variations



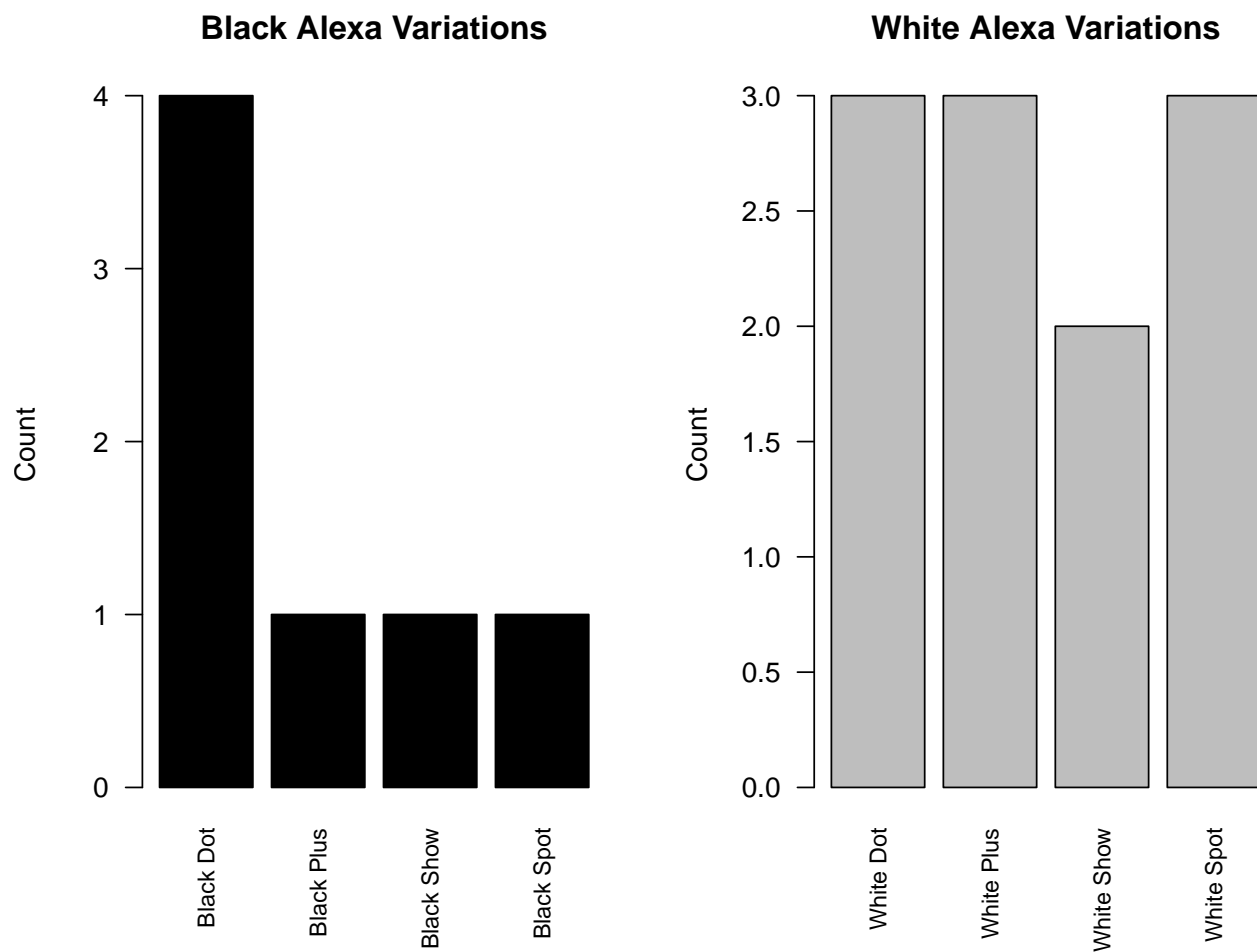
```
#d. Create barplot() for black and white variations side by side
# Filter black and white variations
black_variations <- variations[grep("Black", variations$variation), ]
white_variations <- variations[grep("White", variations$variation), ]

# Set up side-by-side plot
par(mfrow = c(1, 2))

# Black variations barplot
barplot(black_variations$n,
        names.arg = black_variations$variation,
        col = "black",
        main = "Black Alexa Variations",
        ylab = "Count",
        las = 2,
        cex.names = 0.8)

# White variations barplot
barplot(white_variations$n,
        names.arg = white_variations$variation,
        col = "gray",
```

```
main = "White Alexa Variations",
ylab = "Count",
las = 2,
cex.names = 0.8)
```



```
# Reset plot parameters
par(mfrow = c(1, 1))
```