RWorksheet_BIBIT#4b

John Benedict Bibit

2024-10-28

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. Hint Use abs() function to get the absolute value
2. Print the string "*" using for() function. The output should be the same as shown in Figure

```
# Create the vectorA
vectorA <- c(1, 2, 3, 4, 5)

# Create the 5x5 zero matrix
matrix <- matrix(0, nrow = 5, ncol = 5)

# Display the matrix
for (i in 1:5) {
  for (j in 1:5) {
    matrix[i, j] <- abs(i - j)
    cat(matrix[i, j], " ")
  }
  cat("\n")
}
```

```
## 0  1  2  3  4
## 1  0  1  2  3
## 2  1  0  1  2
## 3  2  1  0  1
## 4  3  2  1  0
```

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

```
# Get the starting number from the user
start_number <- readline(prompt = "Enter the starting number: ")
```

```
## Enter the starting number:
```

```
# Convert the input to a number
start_number <- as.numeric(start_number)

# Initialize variables for the Fibonacci sequence
fib_1 <- 0
fib_2 <- 1

# Print the first number in the sequence
print(fib_1)
```

```
## [1] 0
```

```
# Loop to generate and print the Fibonacci sequence
repeat {
  # Calculate the next Fibonacci number
  fib_next <- fib_1 + fib_2

  # Print the next Fibonacci number
  print(fib_next)

  # Update the values for the next iteration
  fib_1 <- fib_2
  fib_2 <- fib_next

  # Break out of the loop if the next Fibonacci number exceeds 500
  if (fib_next > 500) {
    break
  }
}
```

```
## [1] 1
## [1] 2
## [1] 3
## [1] 5
## [1] 8
## [1] 13
## [1] 21
## [1] 34
## [1] 55
## [1] 89
## [1] 144
## [1] 233
## [1] 377
## [1] 610
```

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

```
# Import the Excel file (replace 'your_file.xlsx' with your actual file path)
shoes <- read.csv("shoes_data.csv")
shoes
```

```
##    Gender Shoe_Size Height
## 1       F       6.5   66.0
## 2       F       9.0   68.0
## 3       F       8.5   64.5
## 4       F       8.5   65.0
## 5       M      10.5   70.0
## 6       F       7.0   64.0
## 7       F       9.5   70.0
## 8       F       9.0   71.0
## 9       M      13.0   72.0
## 10      F       7.5   64.0
## 11      M      10.5   74.5
## 12      F       8.5   67.0
## 13      M      12.0   71.0
## 14      M      10.5   71.0
## 15      M      13.0   77.0
## 16      M      11.5   72.0
## 17      F       8.5   59.0
## 18      F       5.0   62.0
## 19      M      10.0   72.0
## 20      F       6.5   66.0
## 21      F       7.5   64.0
## 22      M       8.5   67.0
## 23      M      10.5   73.0
## 24      F       8.5   69.0
## 25      M      10.5   72.0
## 26      M      11.0   70.0
## 27      M       9.0   69.0
## 28      M      13.0   70.0
```

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.

```
femalesubset <- subset(shoes, Gender == "F")
malesubset <- subset(shoes, Gender == "M")
femalecount <- nrow(femalesubset)
malecount <- nrow(malesubset)
cat("Female:", femalecount, "\n")
```

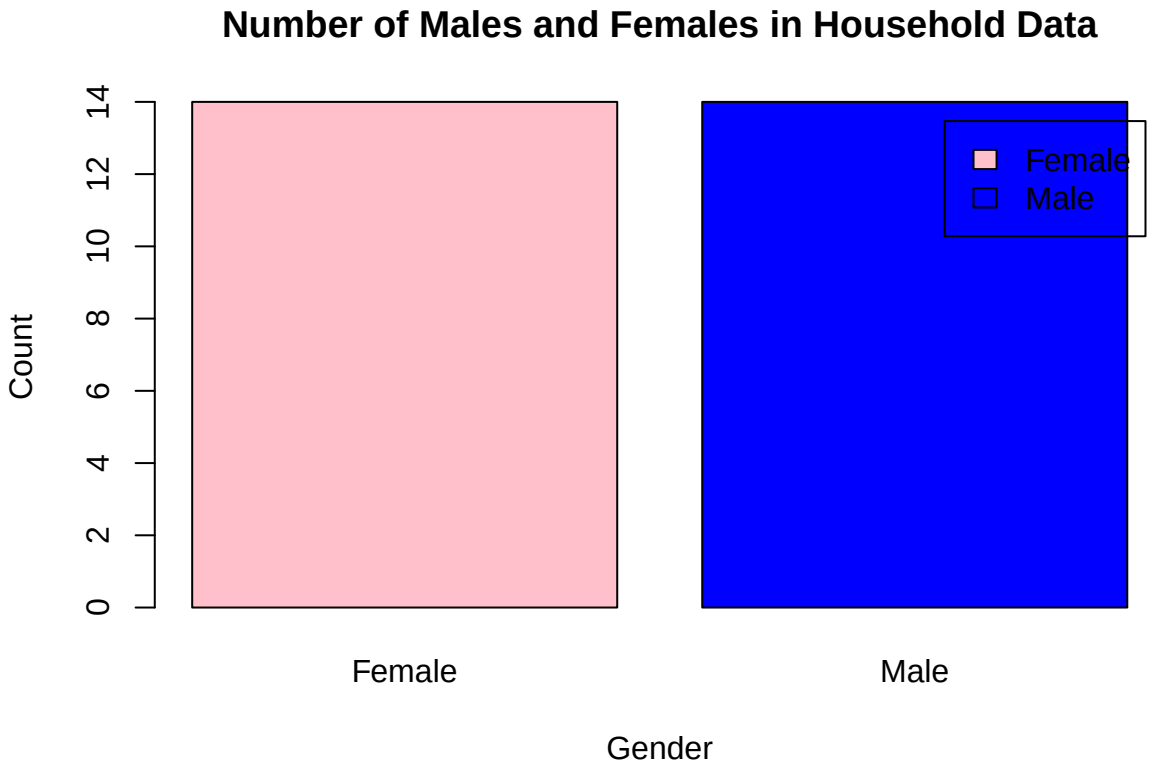
```
## Female: 14
```

```
cat("Male:", malecount, "\n")
```

```
## 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.

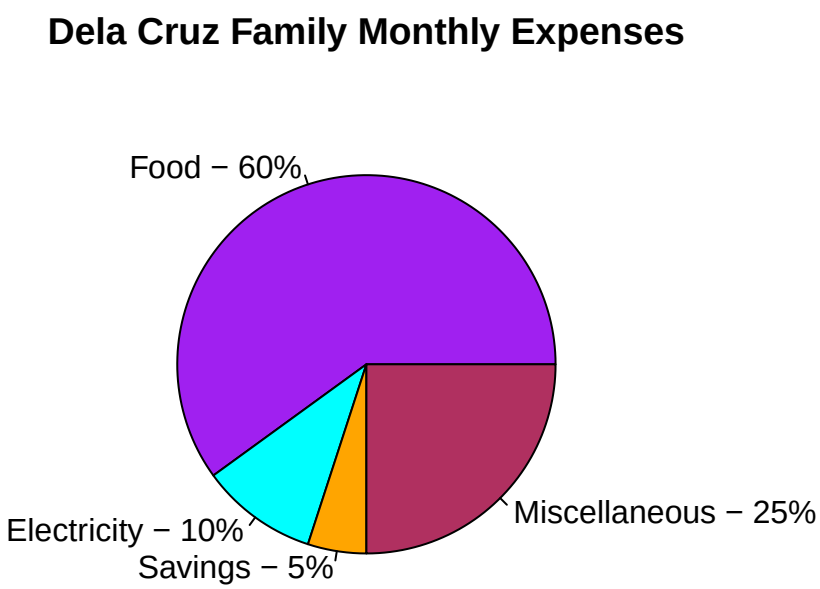
```
gendercounts <- table(shoes$Gender)
barplot(gendercounts,
        main = "Number of Males and Females in Household Data",
        xlab = "Gender",
        ylab = "Count",
        col = c("pink", "blue"),
        legend = c("Female", "Male"),
        names.arg = c("Female", "Male"))
```



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.

```
expenses <- c(Food = 60, Electricity = 10, Savings = 5, Miscellaneous = 25)
percent_labels <- paste0(names(expenses), " - ", round(expenses / sum(expenses) * 100), "%")
pie(expenses,
    labels = percent_labels,
    col = c("purple", "cyan", "orange", "maroon"),
    main = "Dela Cruz Family Monthly Expenses")
```



6. Use the iris dataset.

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

```
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 ...
```

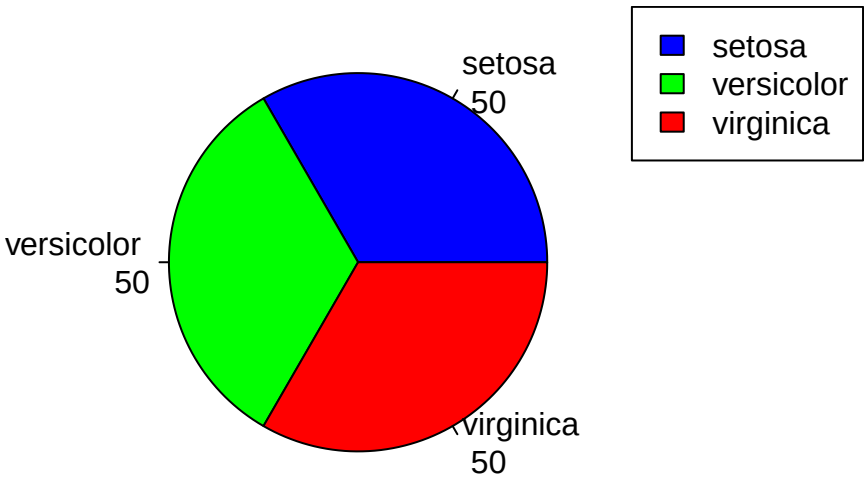
b. Create an R object that will contain the mean of the sepal.length, sepal.width,petal.length,and petal.width. What is the R script and its result?

```
colMeans(iris[, 1:4])
```

```
## 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. Writethe R script and its result.

```
species_counts <- table(iris$Species)
pie(species_counts,
    main = "Species Distribution in Iris Dataset",
    col = c("blue", "green", "red"),
    labels = paste(names(species_counts), "\n", species_counts))
legend("topright",
       legend = names(species_counts),
       fill = c("blue", "green", "red"))
```



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, Species == "setosa")
versicolor <- subset(iris, Species == "versicolor")
virginica <- subset(iris, Species == "virginica")
tail(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
```

```
tail(versicolor)
```

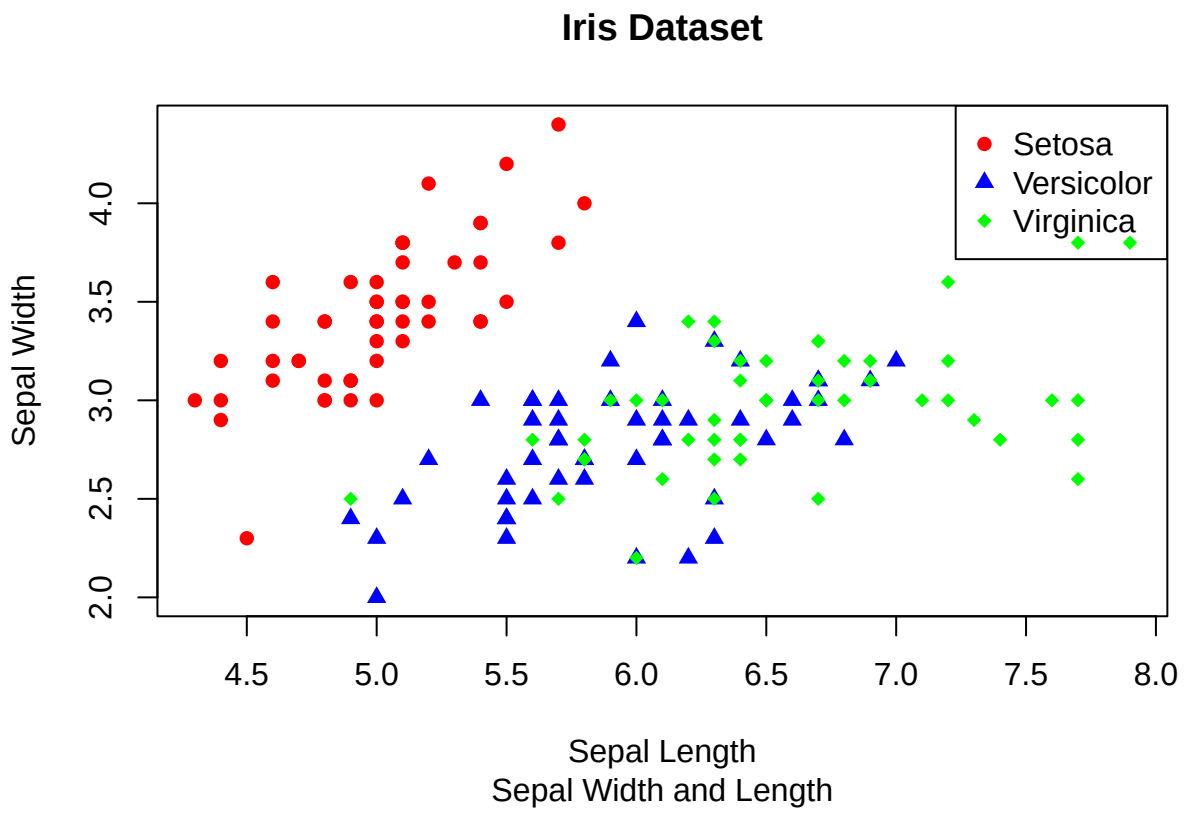
```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 95           5.6         2.7          4.2         1.3 versicolor
## 96           5.7         3.0          4.2         1.2 versicolor
## 97           5.7         2.9          4.2         1.3 versicolor
## 98           6.2     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

```
tail(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.

```
colors <- c("setosa" = "red", "versicolor" = "blue", "virginica" = "green")
symbols <- c("setosa" = 16, "versicolor" = 17, "virginica" = 18)
plot(iris$Sepal.Length, iris$Sepal.Width,
     col = colors[iris$Species],
     pch = symbols[iris$Species],
     main = "Iris Dataset",
     sub = "Sepal Width and Length",
     xlab = "Sepal Length",
     ylab = "Sepal Width")
legend("topright", legend = c("Setosa", "Versicolor", "Virginica"),
       col = c("red", "blue", "green"),
       pch = c(16, 17, 18))
```



f. Interpret the result. The data structure supported both basic analysis and detailed modeling, with mean values offering a quick look at iris flower traits. Visualizations like the pie chart and scatterplot illustrated species distribution and trait correlations, enabling focused, species-specific analysis.

7. Import the alexa-file.xlsx. Check on the variations. Notice that there are extra 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.

```
library(readxl)
alexa_data <- read_excel("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"
```

```
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?

```
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
```

```
variation_counts <- alexa_data %>%
  count(variation)
print(variation_counts)
```

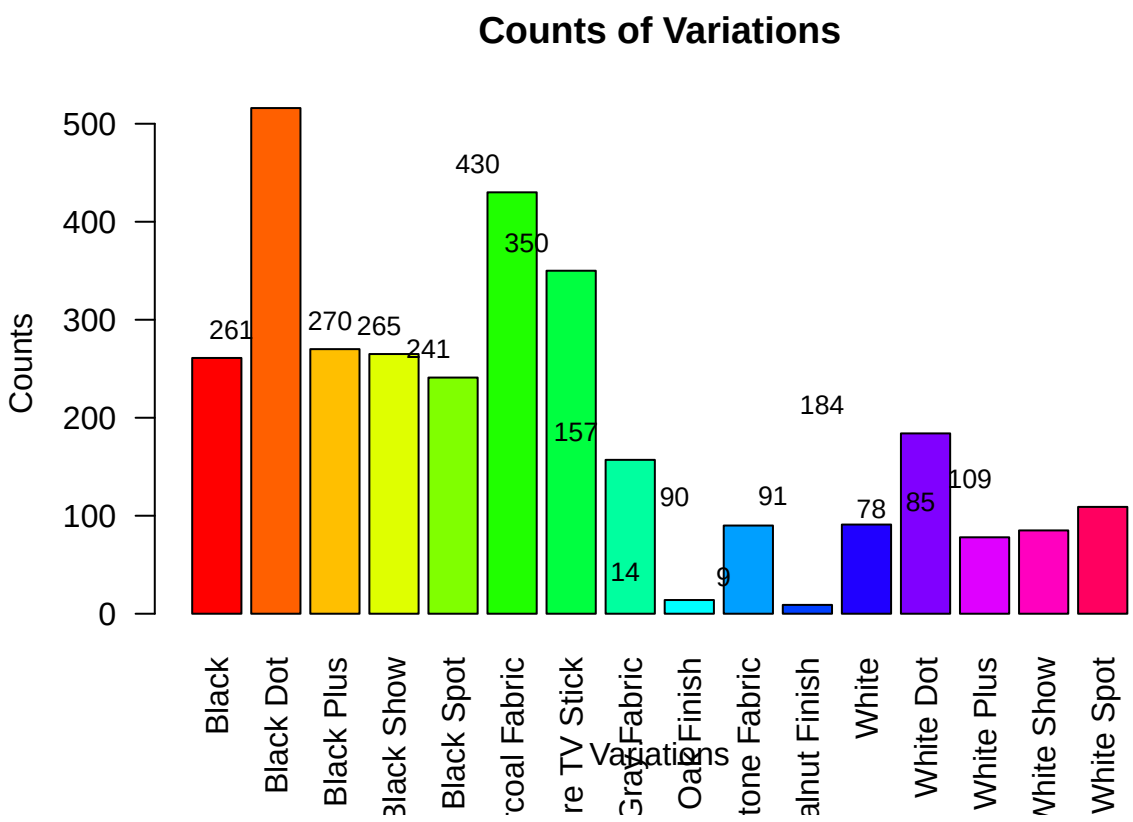
```
## # A tibble: 16 x 2
##    variation                       n
##    <chr>                       <int>
##  1 Black                         261
##  2 Black  Dot                    516
##  3 Black  Plus                   270
##  4 Black  Show                   265
##  5 Black  Spot                 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

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

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

```
library(dplyr)
load("variations.RData")
variation_counts$variation <- gsub(" +", " ", variation_counts$variation)
variation_counts$variation <- trimws(variation_counts$variation)
bar_data <- variation_counts$n
bar_names <- variation_counts$variation
barplot(
  bar_data,
  main = "Counts of Variations",
  col = rainbow(length(bar_data)),
  names.arg = bar_names,
  xlab = "Variations",
  ylab = "Counts",
```

```
  las = 2,
  border = "black"
)
text(x = seq_along(bar_data), y = bar_data, labels = bar_data, pos = 3, cex = 0.8, col = "black")
```



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")
variation_counts$variation <- gsub(" +", " ", variation_counts$variation)
variation_counts$variation <- trimws(variation_counts$variation)
bw_variations <- variation_counts %>%
  filter(grepl("Black|White", variation))
bar_data <- as.matrix(bw_variations$n)
bar_names <- bw_variations$variation
barplot(
  bar_data,
  beside = TRUE,
  main = "Counts of Black and White Variations",
  col = c("black", "gray", "lightgray", "white"),
  names.arg = bar_names,
  xlab = "Variations",
  ylab = "Counts",
  las = 2,
  border = "black"
)
text(x = seq_along(bar_data), y = bar_data, labels = bar_data, pos = 3, cex = 0.8, col = "black")
```