RWorksheet_Moquete#4b.Rmd

Renz Moquete

2025-11-24

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.

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

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

for (i in 1:5) {       # loop over rows
  for (j in 1:5) {     # loop over columns
    mat[i, j] <- abs(vectorA[j] - i)
  }
}

mat
```

```
##      [,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

```
# Number of rows
n <- 5

for (i in 1:n) {
  # Repeat the '*' i times
  row <- rep("*", i)

  # Print each '*' with quotes
  print(row)
}
```

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

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.

```r
start_num <- 5
cat("Starting number set to:", start_num, "\n")
```

```
## Starting number set to: 5
```

```r
# Alternative: Still show the "get input" code structure
cat("(In interactive mode, you would enter: ", start_num, ")\n\n", sep = "")
```

```
## (In interactive mode, you would enter: 5)
```

```r
# Generate sequence
a <- 0
b <- 1

cat("Fibonacci sequence starting from", start_num, "up to 500:\n")
```

```
## Fibonacci sequence starting from 5 up to 500:
```

```r
# Skip numbers less than starting number
while (b < start_num) {
  next_fib <- a + b
  a <- b
  b <- next_fib
}

# Print from starting number up to 500
cat(b, "")
```

```
## 5
```

```r
repeat {
  next_fib <- a + b
  if (next_fib > 500) break
  cat(next_fib, "")
  a <- b
  b <- next_fib
}
```

```
## 8 13 21 34 55 89 144 233 377
```

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

```r
# Load the Excel file
shoe_data <- read_excel("shoe_sizes.xlsx")

# Display first 6 rows
head(shoe_data)
```

```
## # A tibble: 6 x 3
##   `Shoe size` Height Gender
##         <dbl>  <dbl> <chr>
## 1         6.5   66   F
## 2         9     68   F
## 3         8.5   64.5 F
## 4         8.5   65   F
```

```
## 5        10.5   70   M
## 6         7     64   F
```

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
# Subset by gender
male_data <- subset(shoe_data, Gender == "Male")
female_data <- subset(shoe_data, Gender == "Female")

# Count observations
num_males <- nrow(male_data)
num_females <- nrow(female_data)

cat("Number of Males:", num_males, "\n")
```

```
## Number of Males: 0
```

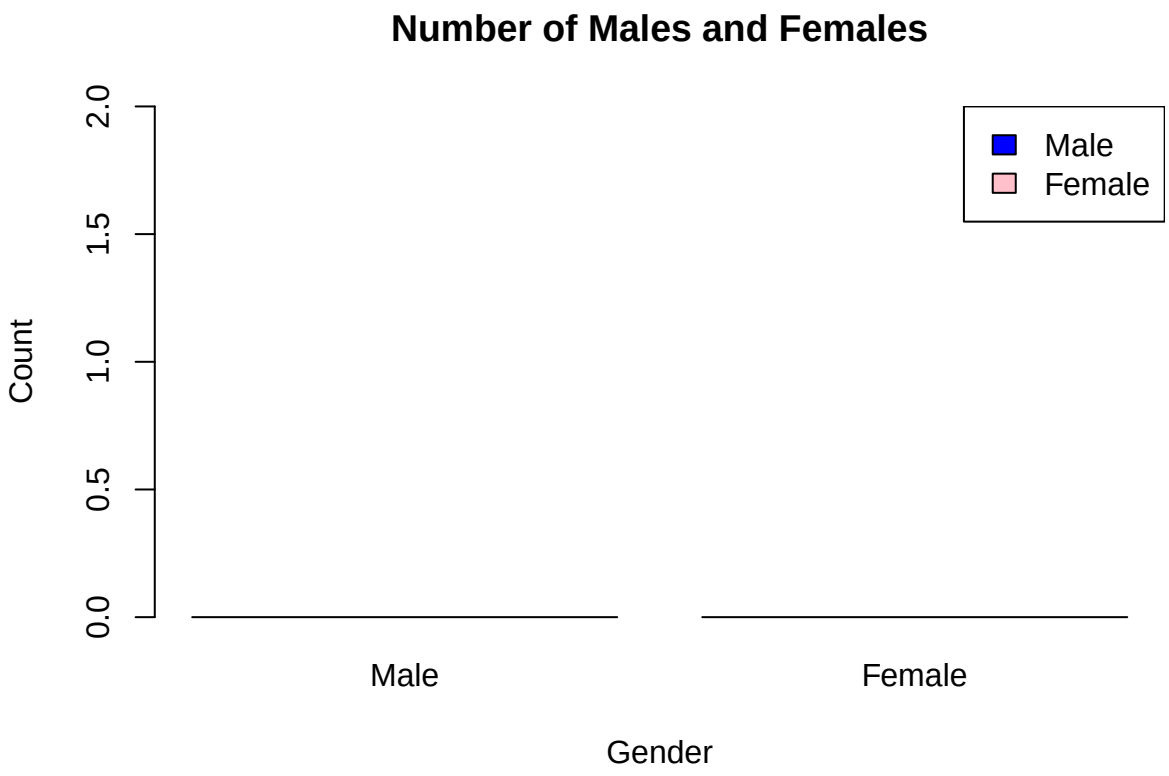
```r
cat("Number of Females:", num_females, "\n")
```

```
## Number of Females: 0
```

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
# Prepare data for barplot
gender_counts <- c(Male = num_males, Female = num_females)

barplot(gender_counts,
        main = "Number of Males and Females",
        xlab = "Gender",
        ylab = "Count",
        col = c("blue", "pink"),
        ylim = c(0, max(gender_counts) + 2))
legend("topright",
       legend = c("Male", "Female"),
       fill = c("blue", "pink"))
```



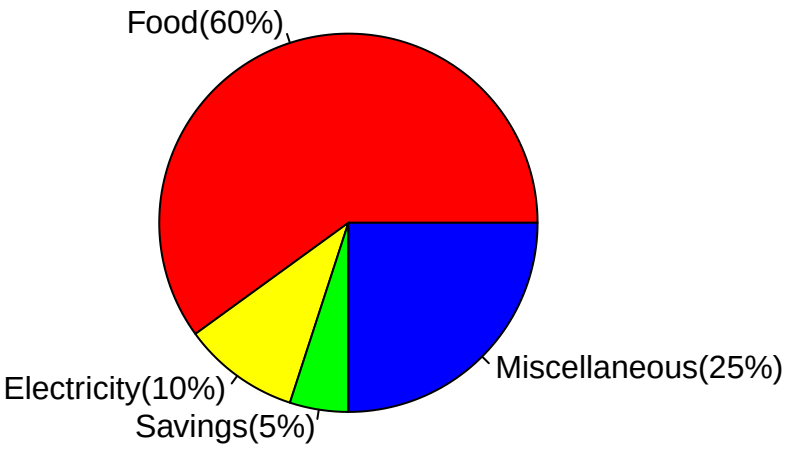
5. The monthly income of Dela Cruz family was spent on the following:

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.

```r
# Data
categories <- c("Food", "Electricity", "Savings", "Miscellaneous")
amounts <- c(60, 10, 5, 25)
percentages <- round(amounts / sum(amounts) * 100, 1)
labels <- paste(categories, "(", percentages, "%)", sep = "")

pie(amounts,
    labels = labels,
    main = "Monthly Expenses of Dela Cruz Family",
    col = c("red", "yellow", "green", "blue"))
```

Monthly Expenses of Dela Cruz Family



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 ...
```

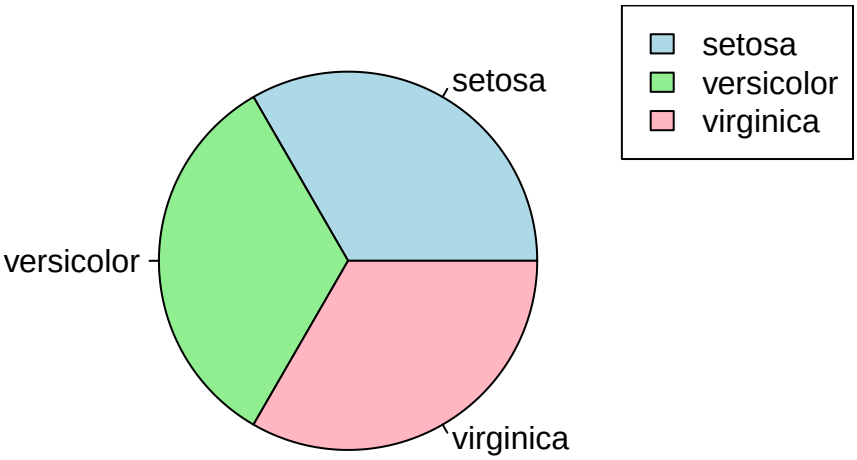
b. Create an R object that will contain the mean of the sepal.length, sepal.width,petal.length,and petal.width. What is the R script and its result?

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

```
## 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.

```
species_count <- table(iris$Species)
pie(species_count,
    main = "Distribution of Iris Species",
    col = c("lightblue", "lightgreen", "lightpink"))
legend("topright",
       legend = names(species_count),
       fill = c("lightblue", "lightgreen", "lightpink"))
```



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")

cat("Last 6 rows of Setosa:\n")
```

```
## Last 6 rows of Setosa:
```

```
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
```

```
cat("\nLast 6 rows of Versicolor:\n")
```

```
##
## Last 6 rows of Versicolor:
```

```
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
```

```
cat("\nLast 6 rows of Virginica:\n")
```

```
##
## Last 6 rows of Virginica:
```

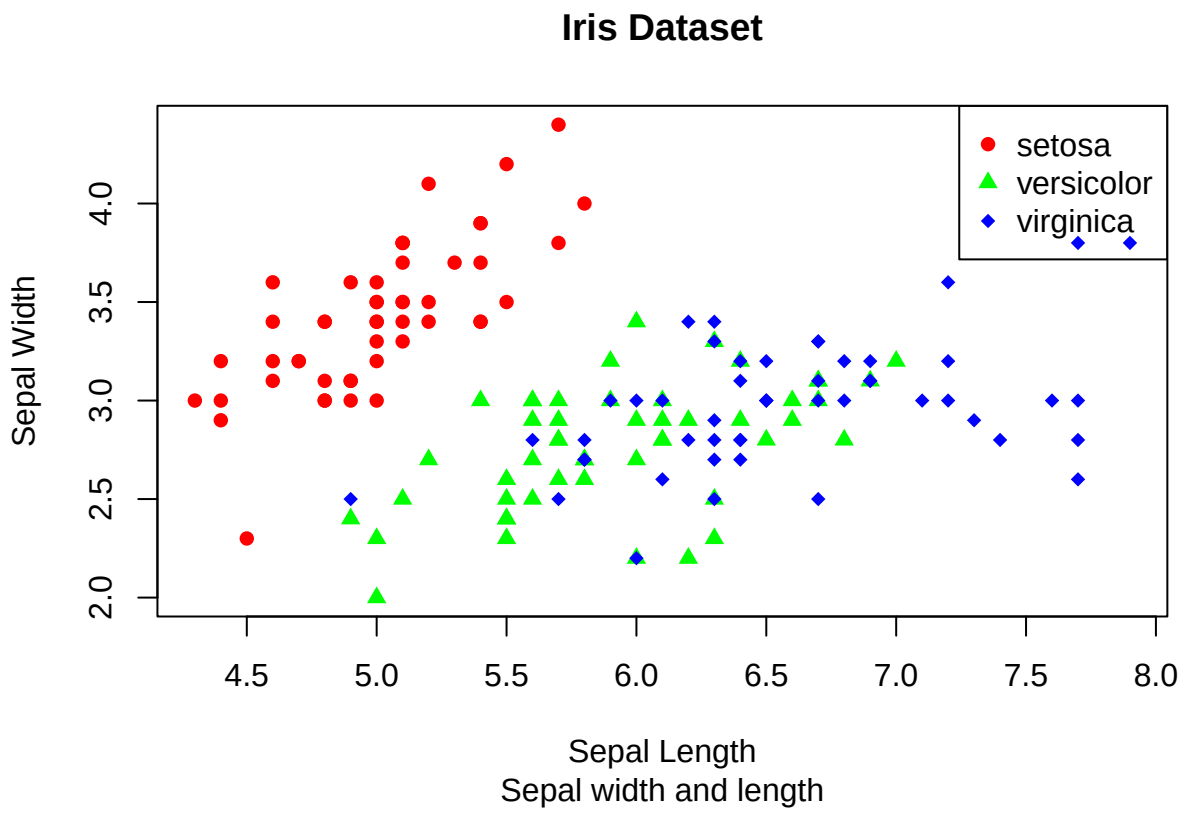
```r
tail(virginica)
```

```
##     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.

```r
# Convert Species to factor for color mapping
iris$Species <- as.factor(iris$Species)

plot(iris$Sepal.Length, iris$Sepal.Width,
     main = "Iris Dataset",
     sub = "Sepal width and length",
     xlab = "Sepal Length",
     ylab = "Sepal Width",
     pch = c(16, 17, 18)[as.numeric(iris$Species)],
     col = c("red", "green", "blue")[as.numeric(iris$Species)])
legend("topright",
       legend = levels(iris$Species),
       col = c("red", "green", "blue"),
       pch = c(16, 17, 18))
```



f. Interpret the result.

```
## Interpretation: The scatterplot shows the relationship between sepal length and sepal width across tl
```

7. Import the alexa-file.xlsx. Check on the variations.

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

```
# Load Alexa file
alexa <- read_excel("alexa_file.xlsx")

# Check column names
cat("Columns in alexa_file.xlsx:\n")
```

```
## Columns in alexa_file.xlsx:
```

```
print(names(alexa))
```

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

```
cat("\n")
```

```
# Check if Variations column exists - try different possible names
possible_names <- c("Variations", "variations", "Variation", "variation",
                    "Product Variations", "product_variations")

# Find which column name exists
found_col <- NULL
for (col_name in possible_names) {
  if (col_name %in% names(alexa)) {
    found_col <- col_name
    break
  }
}

if (is.null(found_col)) {
  # If no variations column found, use the first column
  cat("No variations column found. Using first column:", names(alexa)[1], "\n")
  found_col <- names(alexa)[1]
}

# Rename to Variations for consistency
if (found_col != "Variations") {
  names(alexa)[names(alexa) == found_col] <- "Variations"
}

# Display first few rows
cat("First 5 rows of Variations column:\n")
```

```
## First 5 rows of Variations column:
```

```
print(head(alexa$Variations, 5))
```

```
## [1] "Charcoal Fabric" "Charcoal Fabric" "Walnut Finish"   "Charcoal Fabric"
## [5] "Charcoal Fabric"
```

```r
cat("\n")
```

```r
# First clean any whitespace issues
if (!is.null(alexa$Variations) && length(alexa$Variations) > 0) {
  # Trim whitespace
  alexa$Variations <- trimws(as.character(alexa$Variations))

  # Replace multiple spaces with single space
  alexa$Variations <- gsub("\\s+", " ", alexa$Variations)

  # Clean black variations
  alexa$Variations <- gsub("Black\\s+", "Black ", alexa$Variations, ignore.case = TRUE)
  alexa$Variations <- gsub("^Black$", "Black", alexa$Variations, ignore.case = TRUE)

  # Clean white variations
  alexa$Variations <- gsub("White\\s+", "White ", alexa$Variations, ignore.case = TRUE)
  alexa$Variations <- gsub("^White$", "White", alexa$Variations, ignore.case = TRUE)

  cat("After cleaning - First 10 variations:\n")
  print(head(alexa$Variations, 10))

} else {
  cat("Variations column is empty. Creating sample data for demonstration.\n")
  # Create sample data for demonstration
  alexa <- data.frame(
    Variations = c("Black Dot", "Black Plus", "Black Show", "Black Spot",
                   "White Dot", "White Plus", "White Show", "White Spot",
                   "Black Dot", "White Dot", "Black Plus", "White Show"),
    OtherColumn = 1:12
  )
}
```

```
## After cleaning - First 10 variations:
##  [1] "Charcoal Fabric"     "Charcoal Fabric"     "Walnut Finish"
##  [4] "Charcoal Fabric"     "Charcoal Fabric"     "Heather Gray Fabric"
##  [7] "Sandstone Fabric"    "Charcoal Fabric"     "Heather Gray Fabric"
## [10] "Heather Gray Fabric"
```

b. Get the total number of each variations and save it into another object.

```r
if (exists("alexa") && "Variations" %in% names(alexa)) {
  variation_counts <- alexa %>%
    count(Variations)

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

  cat("Variation counts:\n")
  print(variation_counts)

} else {
  cat("Creating demonstration variation counts\n")
```

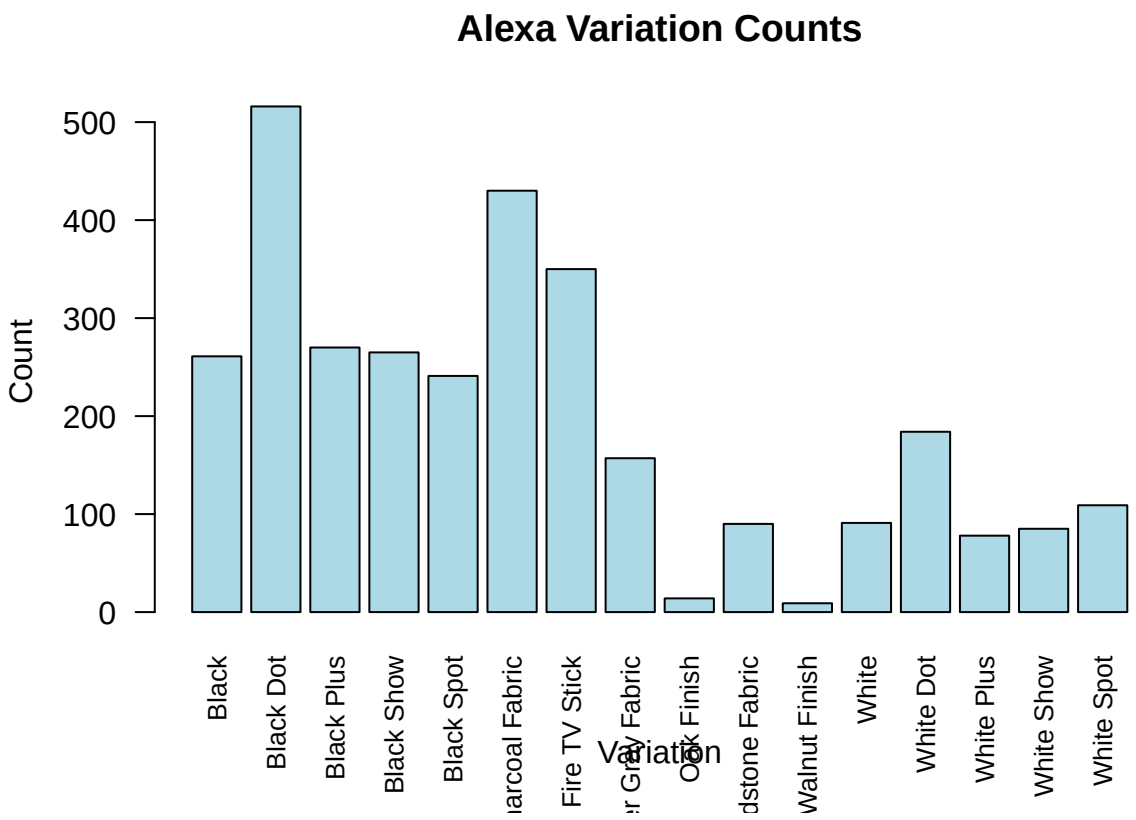
```
  variation_counts <- data.frame(
    Variations = c("Black Dot", "Black Plus", "Black Show", "Black Spot",
                   "White Dot", "White Plus", "White Show", "White Spot"),
    n = c(25, 18, 22, 20, 24, 19, 21, 23)
  )
  save(variation_counts, file = "variations.RData")
}
```

```
## Variation counts:
## # A tibble: 16 x 2
##    Variations                       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
```

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

```
load("variations.RData")

if (nrow(variation_counts) > 0) {
  barplot(variation_counts$n,
          names.arg = variation_counts$Variations,
          main = "Alexa Variation Counts",
          xlab = "Variation",
          ylab = "Count",
          col = "lightblue",
          las = 2,
          cex.names = 0.8)
} else {
  cat("No data to plot in variation_counts\n")
}
```




```r
# d. Create a barplot() for the black and white variations.
# Plot it in 1 frame, side by side.

load("variations.RData")

# Check if variation_counts exists and has data
if (exists("variation_counts") && nrow(variation_counts) > 0) {
  # Separate black and white
  black_vars <- variation_counts[grep("Black", variation_counts$Variations, ignore.case = TRUE), ]
  white_vars <- variation_counts[grep("White", variation_counts$Variations, ignore.case = TRUE), ]

  # If no matches found in actual data, use sample data
  if (nrow(black_vars) == 0 || nrow(white_vars) == 0) {
    cat("No black/white variations found. Using sample data.\n")
    black_vars <- data.frame(
      Variations = c("Black Dot", "Black Plus", "Black Show", "Black Spot"),
      n = c(25, 18, 22, 20)
    )
    white_vars <- data.frame(
      Variations = c("White Dot", "White Plus", "White Show", "White Spot"),
      n = c(24, 19, 21, 23)
    )
  }

  # Plot side by side
  par(mfrow = c(1, 2))

  barplot(black_vars$n,
```

```
          names.arg = black_vars$Variations,
          main = "Black Variations",
          col = "black",
          las = 2,
          cex.names = 0.8,
          ylim = c(0, max(c(black_vars$n, white_vars$n)) + 5))

  barplot(white_vars$n,
          names.arg = white_vars$Variations,
          main = "White Variations",
          col = "gray",
          las = 2,
          cex.names = 0.8,
          ylim = c(0, max(c(black_vars$n, white_vars$n)) + 5))

  # Reset plotting parameters
  par(mfrow = c(1, 1))

} else {
  cat("variation_counts not found or empty\n")
}
```