RWorksheet_Perez#4b

2024-10-28

Using Loop Function

for() loop

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

```
vectorA <- 1:5
matrixA <- matrix(0, nrow=5, ncol=5)
for (i in 1:5) {
    matrixA[i, ] <- abs(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. The output should be the same as shown in Figure.

```
for (i in 1:5) {
    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.

```
first <- as.integer(readline(prompt="Enter the first number of the Fibonacci sequence: "))
```

```
## Enter the first number of the Fibonacci sequence:
```

```
second <- as.integer(readline(prompt="Enter the second number of the Fibonacci sequence: "))
```

```
## Enter the second number of the Fibonacci sequence:
```

```
#for knitting purposes
if (is.null(first) || is.na(first)) first <- 1
if (is.null(second)|| is.na(second)) second <- 1

fibonacci <- c(first, second)
prev <- first
```

```
current <- second
repeat {
  next_val <- prev + current
  if (next_val > 500) break
      fibonacci <- c(fibonacci, next_val)
      prev <- current
      current <- next_val
}
print(fibonacci)
```

```
##  [1]   1   1   2   3   5   8  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.

```
shoe_data <- read.csv("/cloud/project/ShoeData.csv")
shoe_data
```

```
##    Shoe.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   69.0      M
## 28      13.0   70.0      M
```

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.

```
shoe_data <- read.csv("/cloud/project/ShoeData.csv")
head(shoe_data)
```

```
##   Shoe.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
```

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.

```
female <- subset(shoe_data, Gender == "F")
male <- subset(shoe_data, Gender == "M")
cat("Female count:", nrow(female),"\n")
```

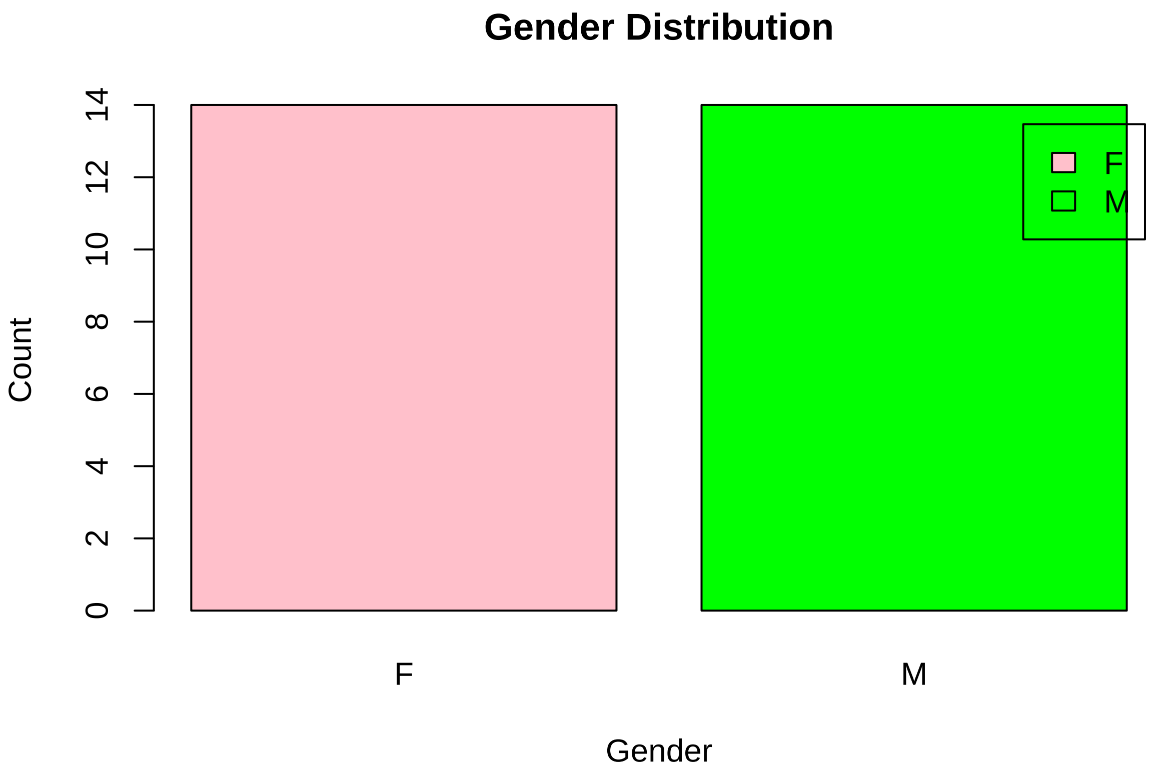
```
## Female count: 14
```

```
cat("Male count:", nrow(male),"\n")
```

```
## Male count: 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.

```
gender_count <- table(shoe_data$Gender)
barplot(gender_count, main= "Gender Distribution", col = c("pink", "green"),
xlab="Gender", ylab="Count", legend=TRUE)
```



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

Food Electricity Savings Miscellaneous

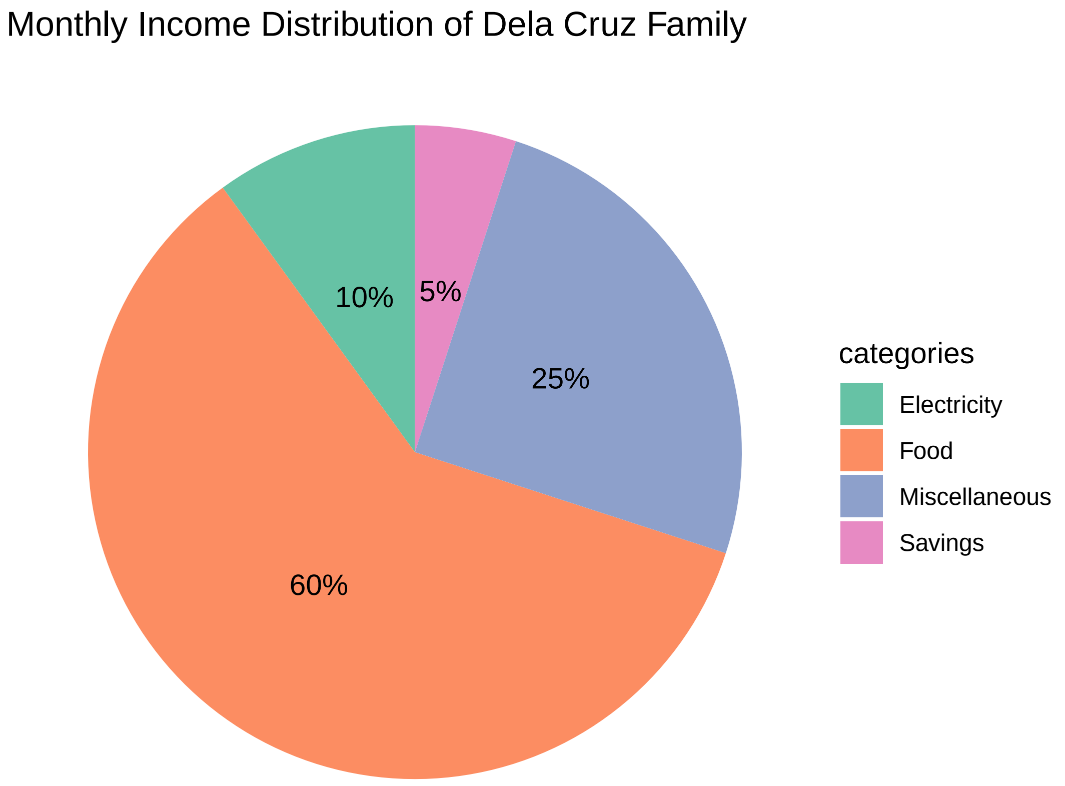
60 10 5 25

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
install.packages("ggplot2")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```r
library(ggplot2)

categories <- c("Food", "Electricity", "Savings", "Miscellaneous")
spent <- c(60, 10, 5, 25)
amount_spent <- data.frame(categories, spent)
amount_spent$percentage <- amount_spent$spent / sum(amount_spent$spent) * 100
ggplot(amount_spent, aes(x = "", y = percentage, fill = categories)) + geom_bar(stat = "identity", widt
geom_text(aes(label = paste0(round(percentage), "%")), position = position_stack(vjust = 0.5)) + labs(t
```



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.

```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 ...
```

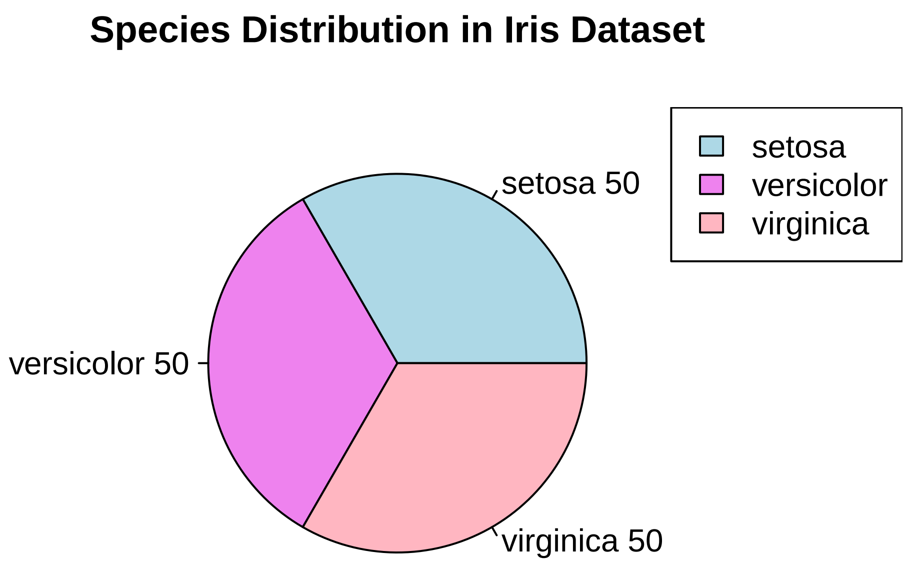
b. Create an R object that will contain the mean of the sepal.length, sepal.width,petal.length,and petal.width. What is the R script and its result?

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

```
## 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
specs <- table(iris$Species)
pie(specs,
main = "Species Distribution in Iris Dataset",
col = c("lightblue", "violet", "lightpink"),
labels = paste(names(specs), specs))
legend("topright", legend = names(specs), fill = c("lightblue", "violet", "lightpink"))
```



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

```r
setosa <- iris[iris$Species == "setosa", ]
versicolor <- iris[iris$Species == "versicolor", ]
virginica <- iris[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
```

```r
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
```

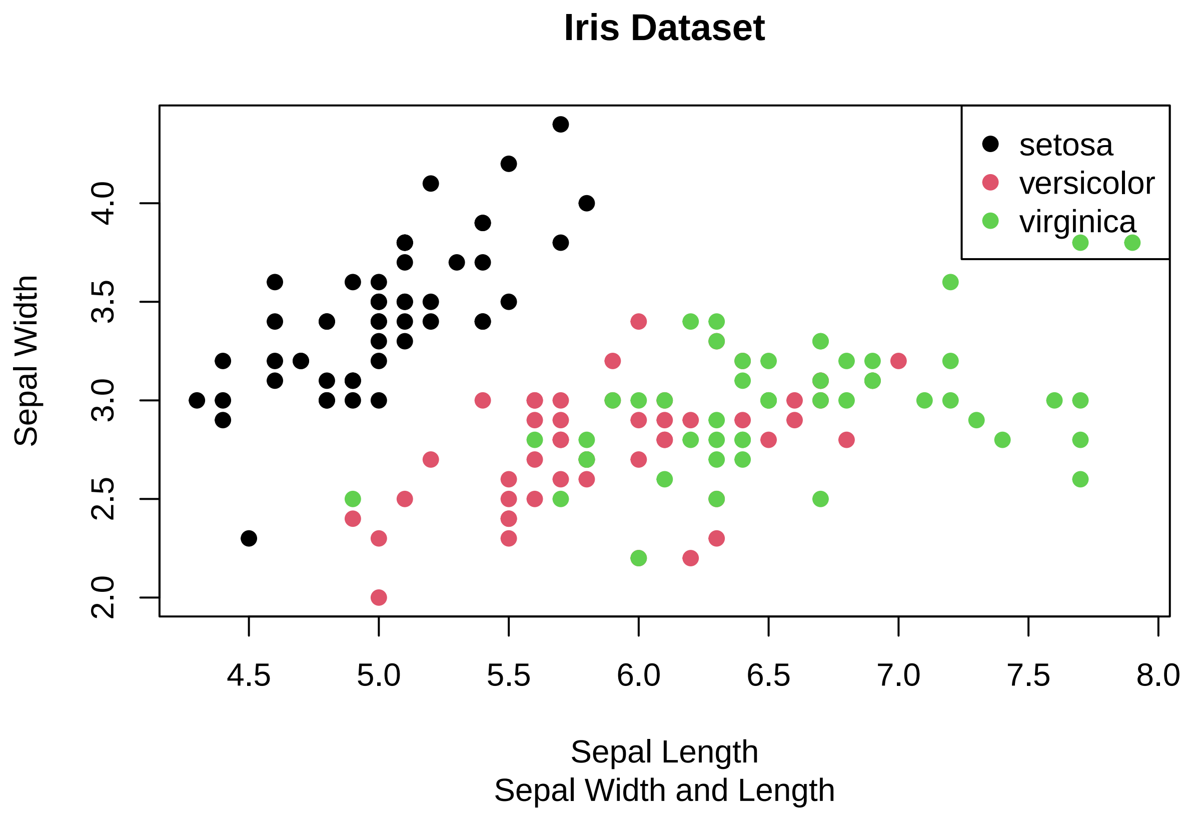
```
tail(virginica)
```

```
##     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.

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

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

plot(iris$Sepal.Length, iris$Sepal.Width,
col = iris$Species,
pch = 19,
main = "Iris Dataset",
sub = "Sepal Width and Length",
xlab = "Sepal Length",
ylab = "Sepal Width")
legend("topright", legend = levels(iris$Species), col = 1:3, pch = 19)
```



f. Interpret the result.

The three species show a clear distinction based on Sepal Length and Width, with minimal overlap. Setosa is easily identifiable due to its shorter Sepal Length, while Versicolor and Virginica have some overlap but can still be differentiated, as Virginica typically has longer Sepal Lengths.

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).

```
install.packages("readxl")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

```
library(readxl)
alexa <- read_xlsx("/cloud/project/alexa_file.xlsx")
```

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

```
alexa$variation <- gsub("Black\\s+", "Black", alexa$variation)
alexa$variation <- gsub("White\\s+", "White", alexa$variation)
```

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.

```
install.packages("dplyr")
```

```
## Installing package into '/cloud/lib/x86_64-pc-linux-gnu-library/4.4'
## (as 'lib' is unspecified)
```

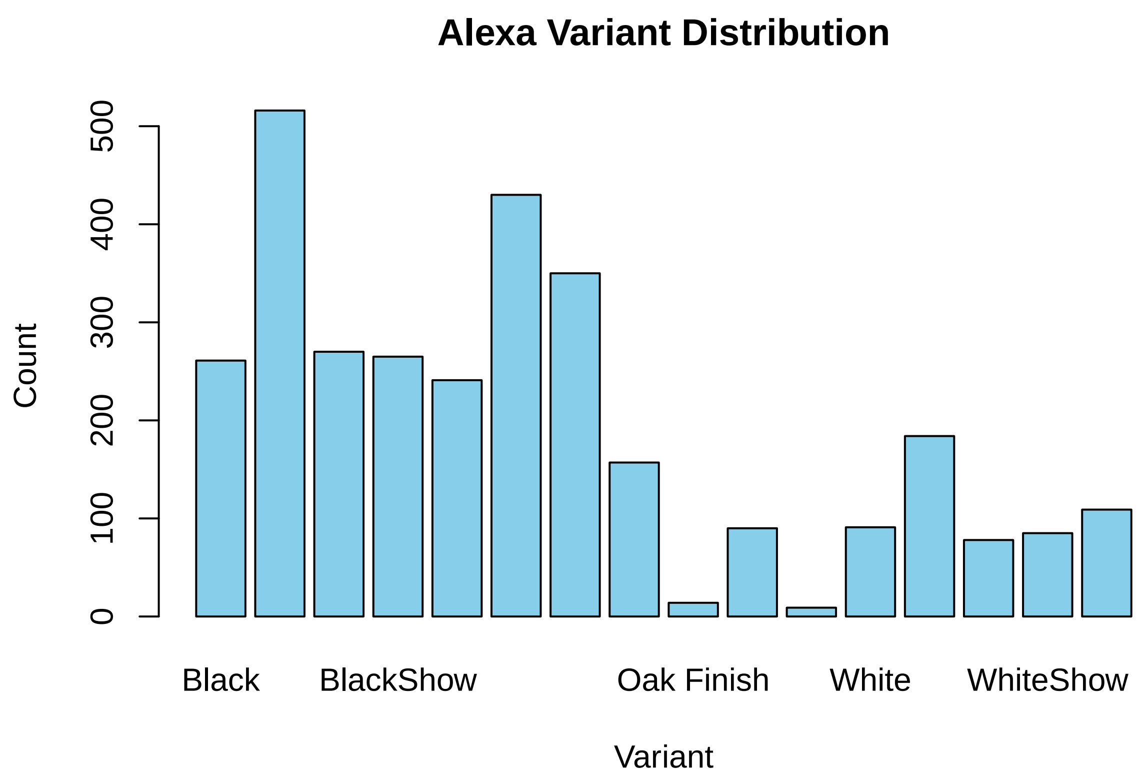
```
library(dplyr)
```

```
##
## 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
```

```
variationCount <- alexa %>% count(variation)
save(variationCount, 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.

```
load("variations.RData")
barplot(variationCount$n, names.arg = variationCount$variation, col = "skyblue", main = "Alexa Variant 
```



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

```r
variantCount <- alexa %>%
  group_by(variation) %>%
  summarize(count = n())
```

```r
black_variants <- variantCount %>%
  filter(grepl("Black", variation))
white_variants <- variantCount %>%
  filter(grepl("White", variation))

print(black_variants)
```

```
## # A tibble: 5 x 2
##   variation count
##   <chr>     <int>
## 1 Black       261
## 2 BlackDot    516
## 3 BlackPlus   270
## 4 BlackShow   265
## 5 BlackSpot   241
```

```r
print(white_variants)
```

```
## # A tibble: 5 x 2
##   variation count
##   <chr>     <int>
## 1 White        91
## 2 WhiteDot    184
## 3 WhitePlus    78
## 4 WhiteShow    85
## 5 WhiteSpot   109
```

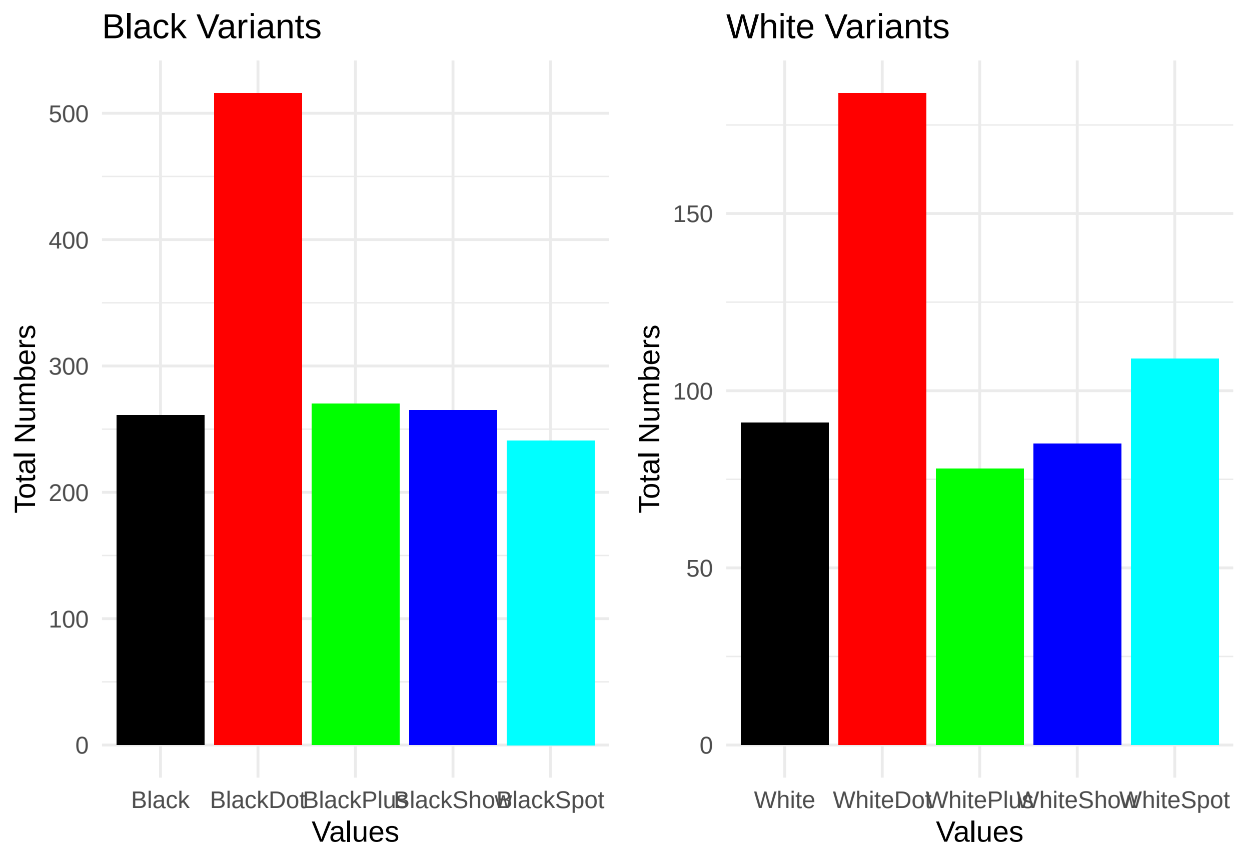
```r
black_plot <- ggplot(black_variants, aes(x = variation, y = count, fill = variation)) +
  geom_bar(stat = "identity") +
  labs(title = "Black Variants", x = "Values", y = "Total Numbers") +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_fill_manual(values = c("black", "red", "green", "blue", "cyan"))
```

```r
white_plot <- ggplot(white_variants, aes(x = variation, y = count, fill = variation)) +
  geom_bar(stat = "identity") +
  labs(title = "White Variants", x = "Values", y = "Total Numbers") +
  theme_minimal() +
  theme(legend.position = "none") +
  scale_fill_manual(values = c("black", "red", "green", "blue", "cyan"))
```

```r
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
grid.arrange(black_plot, white_plot, ncol = 2)
```


Black Variants
Total Numbers
Black
BlackDot
BlackPlus
BlackShow
BlackSpot
Values
White Variants
Total Numbers
White
WhiteDot
WhitePlus
WhiteShow
WhiteSpot
Values