RWorksheet_Josue#4b

Miguel F. Josue Jr.

2024-10-29

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)
matrix5x5 <- matrix(0, nrow = 5, ncol = 5)

for (i in 1:5) {
  for (j in 1:5) {
    matrix5x5[i, j] <- vectorA[abs(i - j) + 1] - 1
  }
}
matrix5x5
```

```
##      [,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.

```
start <- as.integer(readline(prompt = "Enter the starting number:"))
```

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

```
start <- 5
a <- 0
b <- 1
cat("Fibonacci sequence starting from", start, "and up to 500:\n")
```

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

```
repeat {
  fib <- a + b
  if (fib > 500) break
  if (fib >= start) cat(fib, " ")
  a <- b
  b <- fib
}
```

```
## 5  8  13  21  34  55  89  144  233  377
```

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

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

```
#4A.What is the R script for importing an excel or a csv file? Display the first 6 rows of the dataset?
shoe_data <- read.csv("shoesize_data.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
```

```
#4B. Create a subset for gender(female and male). How many observations are there inMale? How about in
male <- subset(shoe_data, Gender == "M")
male
```

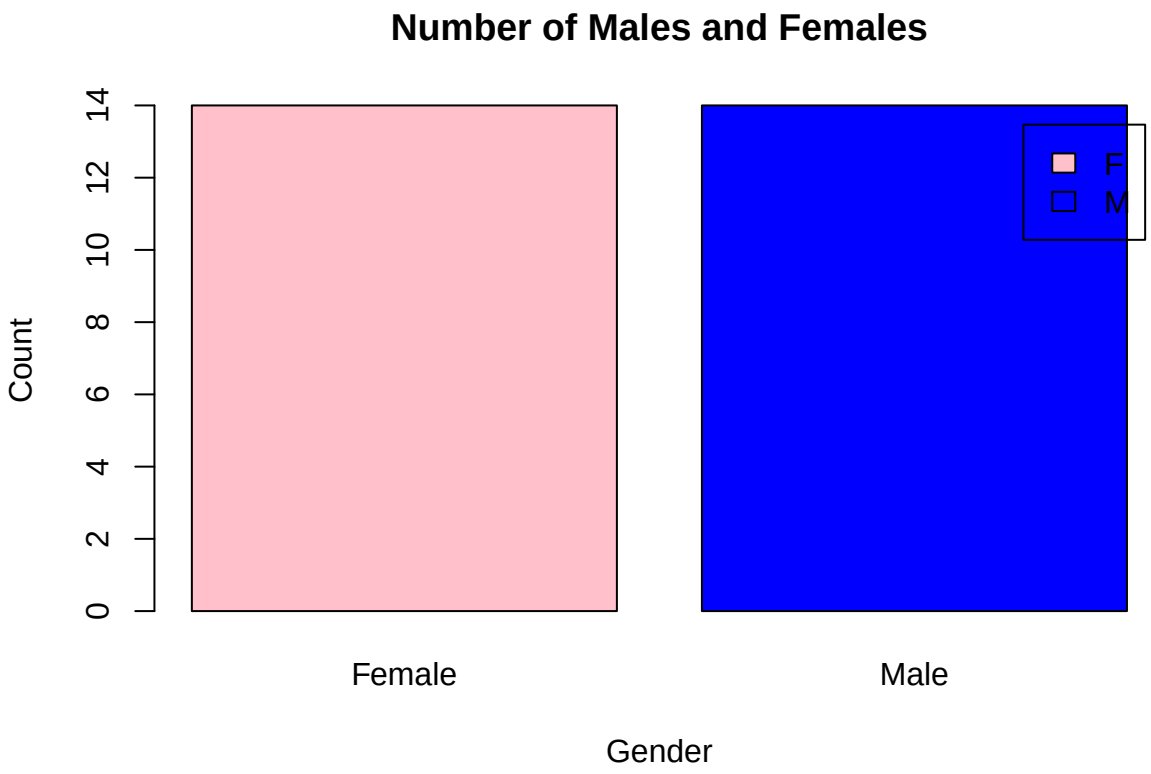
```
##    Shoe_Size Height Gender
## 5       10.5   70.0      M
## 9       13.0   72.0      M
## 11      10.5   74.5      M
## 13      12.0   71.0      M
## 14      10.5   71.0      M
## 15      13.0   77.0      M
## 16      11.5   72.0      M
## 19      10.0   72.0      M
```

```
## 22       8.5   67.0      M
## 23      10.5   73.0      M
## 25      10.5   72.0      M
## 26      11.0   70.0      M
## 27       9.0   69.0      M
## 28      13.0   70.0      M
```

```r
female <- subset(shoe_data, Gender == "F")
female
```

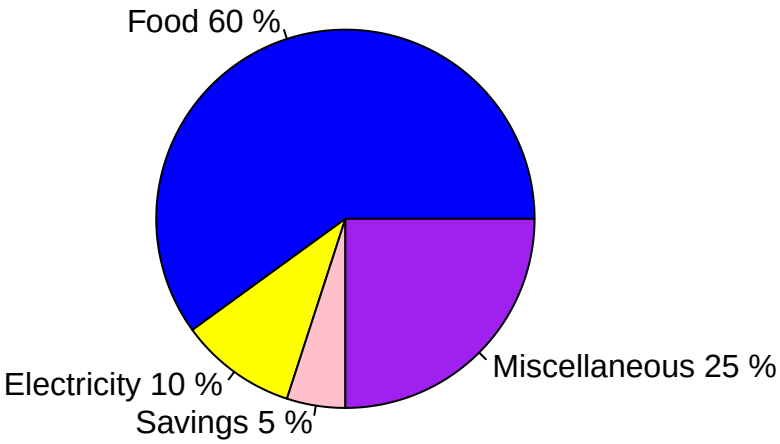
```
##    Shoe_Size Height Gender
## 1        6.5   66.0      F
## 2        9.0   68.0      F
## 3        8.5   64.5      F
## 4        8.5   65.0      F
## 6        7.0   64.0      F
## 7        9.5   70.0      F
## 8        9.0   71.0      F
## 10       7.5   64.0      F
## 12       8.5   67.0      F
## 17       8.5   59.0      F
## 18       5.0   62.0      F
## 20       6.5   66.0      F
## 21       7.5   64.0      F
## 24       8.5   69.0      F
```

```r
#4C. Create a graph for the number of males and females for Household Data. Use plot(),chart type = bar
#C. Barplot for genders
gender <- table(shoe_data$Gender)

barplot(gender,
        main = "Number of Males and Females",
        xlab = "Gender",
        ylab = "Count",
        col = c("Pink", "Blue"),
        names.arg = c("Female", "Male"),
        legend = rownames(gender))
```



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

```
#5A. Create a piechart that will include labels in percentage.Add some colors and title of the chart. W
expenses <- c(Food = 60, Electricity = 10, Savings = 5, Miscellaneous = 25)
percentages <- round(expenses / sum(expenses) * 100)
pie(expenses,
    labels = paste(names(expenses), percentages, "%"),
    col =  c("blue", "yellow", "pink", "purple"),
    main = "Dela Cruz Family Monthly Expenses")
```



6. Use the iris dataset.

```
data(iris)
```

```
#6A. Check for the structure of the dataset using the str() function. Describe what you have seen in th
str(iris)
```

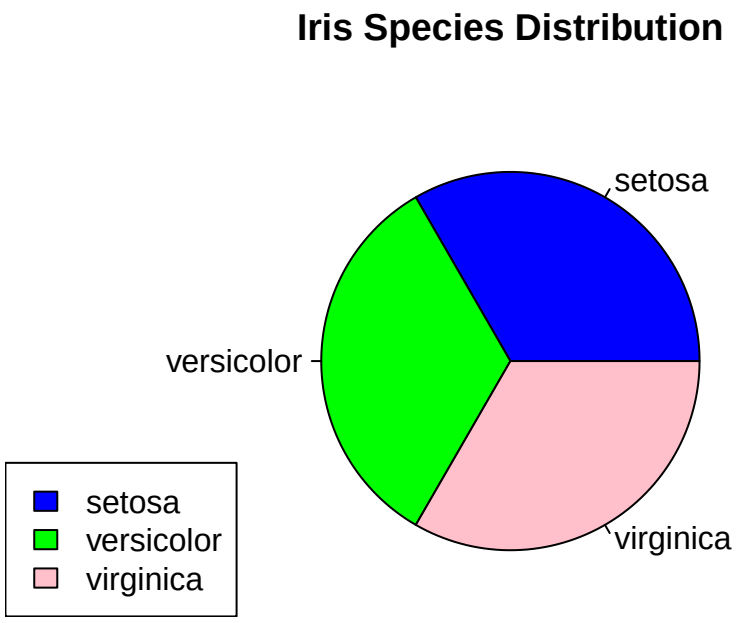
```
## 'data.frame':    150 obs. of  5 variables:
##  $ Sepal.Length: num  5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
##  $ Sepal.Width : num  3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
##  $ Petal.Length: num  1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
##  $ Petal.Width : num  0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
##  $ Species     : Factor w/ 3 levels "setosa","versicolor",..: 1 1 1 1 1 1 1 1 1 1 ...
```

```
# The data set provided information about the width and length of both sepals and petals, the species t
```

```
#6B. Create an R object that will contain the mean of the sepal.length,sepal.width,petal.length,and pet
mean <- colMeans(iris[, 1:4])
mean
```

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

```r
#6C. Create a pie chart for the Species distribution. Add title, legends, and colors. Write the R scrip
species <- table(iris$Species)
colors <- c("blue", "green", "pink")
pie(species,
    main = " Iris Species Distribution ",
    col = colors,
    labels = names(species))

legend("bottomleft", legend = names(species), fill = colors)
```




```r
#6D. Subset the species into setosa, versicolor, and virginica. Write the R scripts and show the last s
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
```

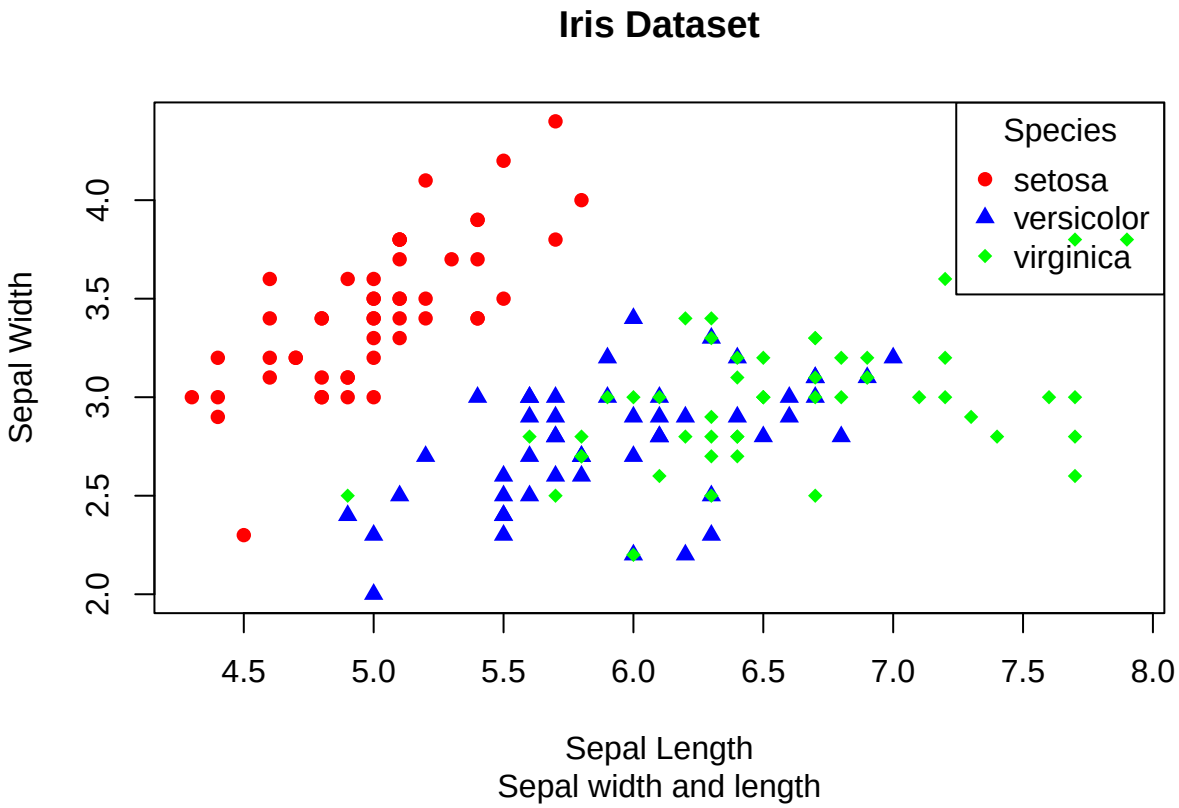
```r
tail(versicolor)
```

```
##     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
```

```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
```

```r
#4E. Create a scatterplot of the sepal.length and sepal.width using the differentspecies(setosa,versico
species_colors <- c("setosa" = "red", "versicolor" = "blue", "virginica" = "green")
species_pch <- c("setosa" = 16, "versicolor" = 17, "virginica" = 18)
plot(iris$Sepal.Length, iris$Sepal.Width,
     main = "Iris Dataset",
     sub = "Sepal width and length",
     xlab = "Sepal Length",
     ylab = "Sepal Width",
     col = species_colors[iris$Species],
     pch = species_pch[iris$Species])

legend("topright", legend = names(species_colors),
       col = species_colors,
       pch = species_pch,
       title = "Species")
```



6F. Interpret the result.

Setosa: This flower type stands out clearly. Its points are clustered in a single area with shorter yet , wider sepals than the others. This means that Setosa has more compact sepals, making it easy to tell apart from the others

Versicolor: Versicolor's points are more spread out and overlap a bit with Virginica's, meaning it shares similar sepal sizes with Virginica. Its sepals are generally longer than Setosa's but still vary in width.

Virginica: Virginica has the longest sepals overall, and it also has some overlap with Versicolor. This makes it a bit harder to distinguish from Versicolor based on these measurements alone.

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).

```
library(readxl)
alexa <- read_excel("alexa_file.xlsx")
alexa
```

```
## # A tibble: 3,150 x 5
##    rating date                variation           verified_reviews      feedback
##     <dbl> <dttm>              <chr>               <chr>                    <dbl>
##  1      5 2018-07-31 00:00:00 Charcoal Fabric     Love my Echo!                1
##  2      5 2018-07-31 00:00:00 Charcoal Fabric     Loved it!                    1
##  3      4 2018-07-31 00:00:00 Walnut Finish       Sometimes while play~        1
##  4      5 2018-07-31 00:00:00 Charcoal Fabric     I have had a lot of ~        1
##  5      5 2018-07-31 00:00:00 Charcoal Fabric     Music                        1
```

```
##  6      5 2018-07-31 00:00:00 Heather Gray Fabric I received the echo ~        1
##  7      3 2018-07-31 00:00:00 Sandstone Fabric    Without having a cel~        1
##  8      5 2018-07-31 00:00:00 Charcoal Fabric     I think this is the ~        1
##  9      5 2018-07-30 00:00:00 Heather Gray Fabric looks great                  1
## 10      5 2018-07-30 00:00:00 Heather Gray Fabric Love it! I've listen~        1
## # i 3,140 more rows
```

```
#7A. Rename the white and black variants by using gsub() function.
alexa$variation <- gsub("Black  Dot", "BlackDot", alexa$variation)
alexa$variation <- gsub("Black  Plus", "BlackPlus", alexa$variation)
alexa$variation <- gsub("Black  Show", "BlackShow", alexa$variation)
alexa$variation <- gsub("Black  Spot", "BlackSpot", alexa$variation)
alexa$variation <- gsub("White  Spot", "WhiteSpot", alexa$variation)
alexa$variation <- gsub("White  Show", "WhiteShow", alexa$variation)
alexa$variation <- gsub("White  Plus", "WhitePlus", alexa$variation)
alexa$variation <- gsub("White  Dot", "WhiteDot", alexa$variation)
alexa[1520:2000, ]
```

```
## # A tibble: 481 x 5
##    rating date                variation verified_reviews                feedback
##     <dbl> <dttm>              <chr>     <chr>                              <dbl>
##  1      5 2018-07-30 00:00:00 BlackShow Does everything I could ask an~        1
##  2      5 2018-07-30 00:00:00 BlackShow I like how clear the screen is~        1
##  3      5 2018-07-30 00:00:00 WhiteShow It is so easy to use and is ev~        1
##  4      5 2018-07-30 00:00:00 WhiteShow I love the Echo Show. I'm goin~        1
##  5      5 2018-07-30 00:00:00 BlackShow I love it it is awesome                1
##  6      5 2018-07-30 00:00:00 BlackShow Very Good to be able to listen~        1
##  7      5 2018-07-30 00:00:00 BlackShow Love it!                               1
##  8      5 2018-07-30 00:00:00 BlackShow Great sound and the screen is ~        1
##  9      5 2018-07-30 00:00:00 BlackShow This has been perfect for bein~        1
## 10      5 2018-07-30 00:00:00 BlackShow I really love this item!  So m~        1
## # i 471 more rows
```

Write the R scripts and show an example of the output by getting a snippet. To embed an image into Rmd, use the function below:

```
library(knitr)
include_graphics("sampleimg.png")
```

rating <dbl>	date <S3: POSIXct>	variation <chr>
5	2018-07-30	Black Show
5	2018-07-30	Black Show
5	2018-07-30	White Show
5	2018-07-30	White Show
5	2018-07-30	Black Show
5	2018-07-30	Black Show
5	2018-07-30	Black Show
5	2018-07-30	Black Show
5	2018-07-30	Black Show
5	2018-07-30	Black Show

7B. 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.

```
library(dplyr)
```

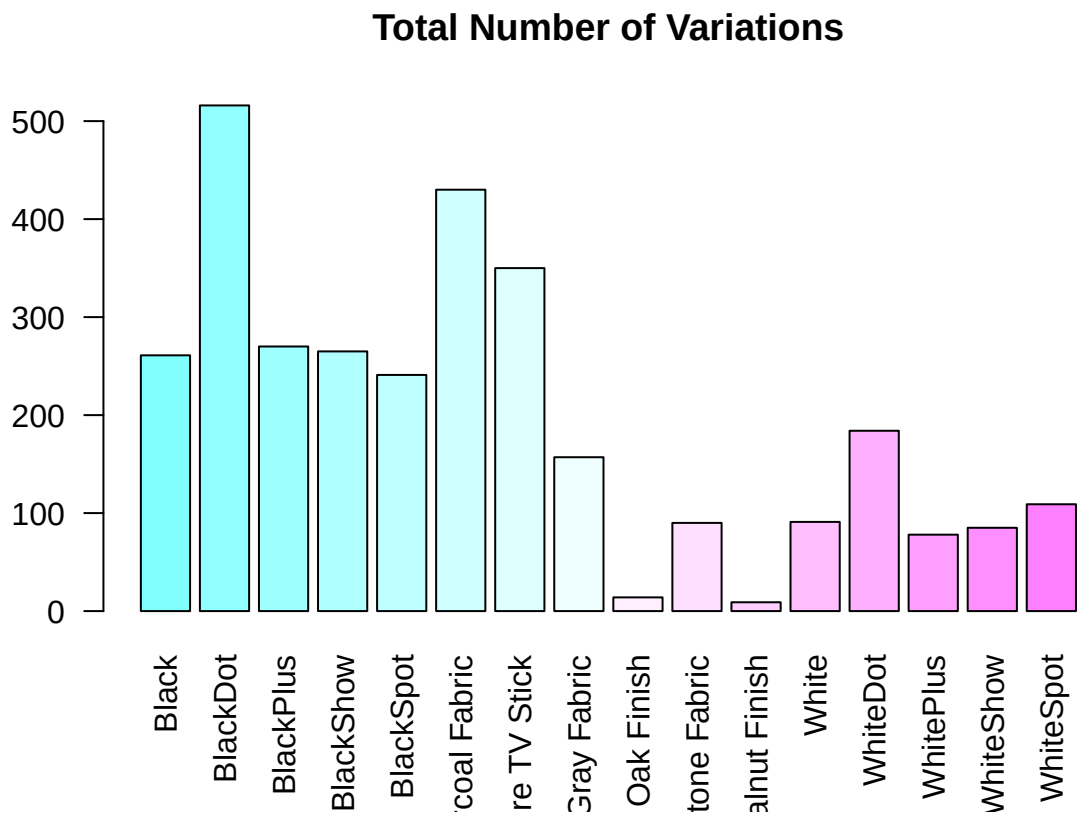
```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
alexa_var <- alexa %>%
  count(alexa$variation)
save(alexa_var, file = "variations.RData")
alexa_var
```

```
## # A tibble: 16 x 2
##    `alexa$variation`                n
##    <chr>                        <int>
##  1 Black                          261
##  2 BlackDot                       516
##  3 BlackPlus                      270
##  4 BlackShow                      265
##  5 BlackSpot                      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 WhiteDot                       184
## 14 WhitePlus                       78
## 15 WhiteShow                       85
## 16 WhiteSpot                      109
```

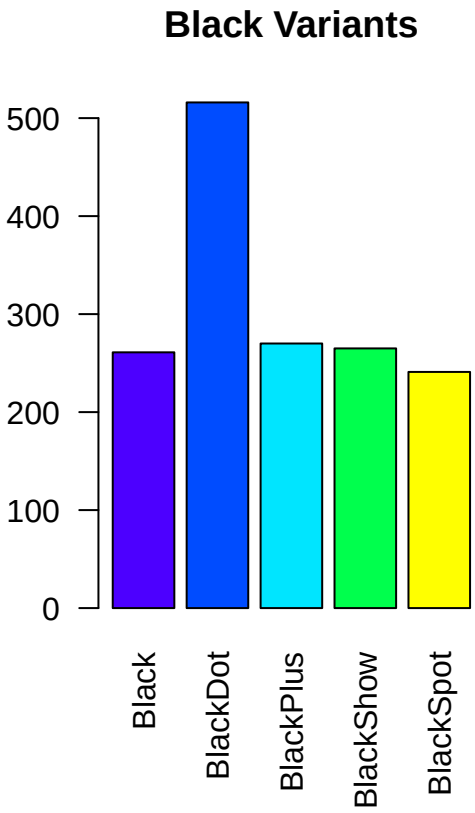
7C.From the variations.RData, create a barplot(). Complete the details of the chart which include the title, color, labels of each bar.

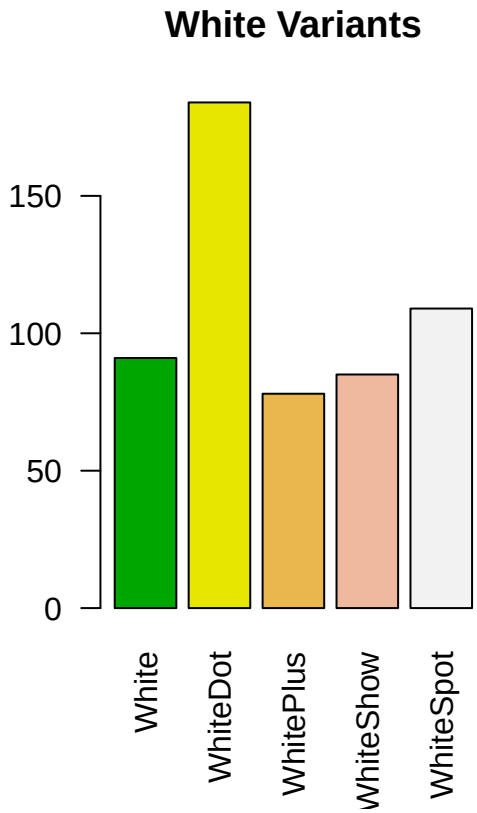
```
load("variations.RData")
var_bar <- barplot(
  alexa_var$n,
  names.arg = alexa_var$`alexa$variation`,
  col = cm.colors(length(alexa_var$n)),
  main = "Total Number of Variations",
  las = 2,
)
```



7D.Create a barplot() for the black and white variations. Plot it in 1 frame, side by side. Complete the details of the chart.

```
load("variations.RData")
par(mfrow = c(1, 2))

black <- alexa_var %>%
  filter(`alexa$variation` %in% c("Black", "BlackDot", "BlackPlus", "BlackShow", "BlackSpot"))

barplot(
  height = black$n,
  names.arg = black$`alexa$variation`,
  col = topo.colors(length(black$n)),
  main = "Black Variants",
  las = 2
)

white <- alexa_var %>%
  filter(`alexa$variation` %in% c("White", "WhiteDot", "WhitePlus", "WhiteShow", "WhiteSpot"))

barplot(
  height = white$n,
  names.arg = white$`alexa$variation`,
  col = terrain.colors(length(white$n)),
  main = "White Variants",,
  las = 2
)
```


Black Variants
500
400
300
200
100
0
Black
BlackDot
BlackPlus
BlackShow
BlackSpot


White Variants
150
100
50
0
White
WhiteDot
WhitePlus
WhiteShow
WhiteSpot