

RWorksheet__Benedicto#4b

Gabriel R. Benedicto

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

```
vectorA <- c(1, 2, 3, 4, 5)
matrix_5x5 <- matrix( nrow = 5, ncol = 5)

for (i in 1:5) {
  for (j in 1:5) {
    matrix_5x5[i, j] <- abs(i - j)
  }
}

print(matrix_5x5)
```

```
##      [,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) {
  line <- rep('*', i)
  cat(line, sep = " ")
  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: "))
start <- 1
a <- start
b <- 1
cat(a, b, sep = " ")
```

```
## 1 1
```

```
repeat {
  next_term <- a + b
  if (!is.na(next_term) && next_term > 500) {
    break
  }
  cat(next_term, " ")
  a <- b
  b <- next_term
}
```

```
## 2 3 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.

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

```
data <- read.csv("Shoe_sizes.csv")
```

```
head(data)
```

```
##   Show.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.

```
male_data <- subset(data, Gender == "M")
female_data <- subset(data, Gender == "F")
```

```
num_males <- nrow(male_data)
num_females <- nrow(female_data)
```

```
num_males
```

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

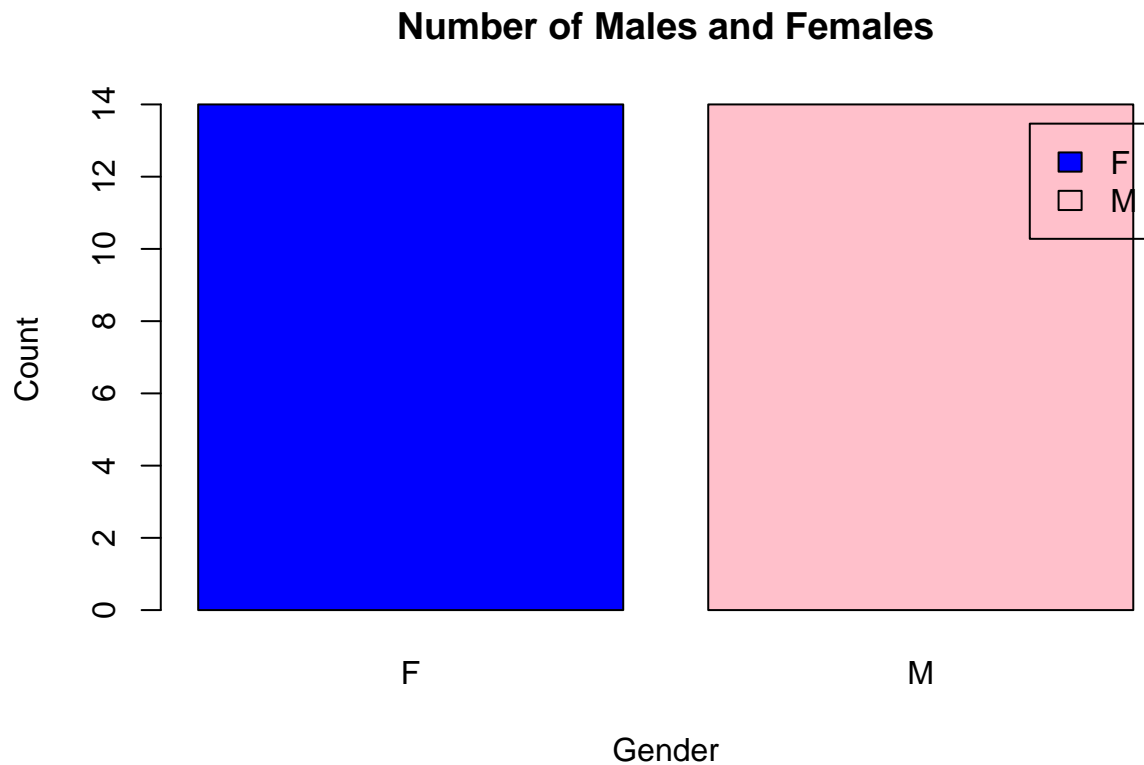
```
num_females
```

```
## [1] 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_counts <- table(data$Gender)
```

```
barplot(gender_counts,
  main = "Number of Males and Females",
  xlab = "Gender",
  ylab = "Count",
  col = c("blue", "pink"),
  legend = rownames(gender_counts))
```

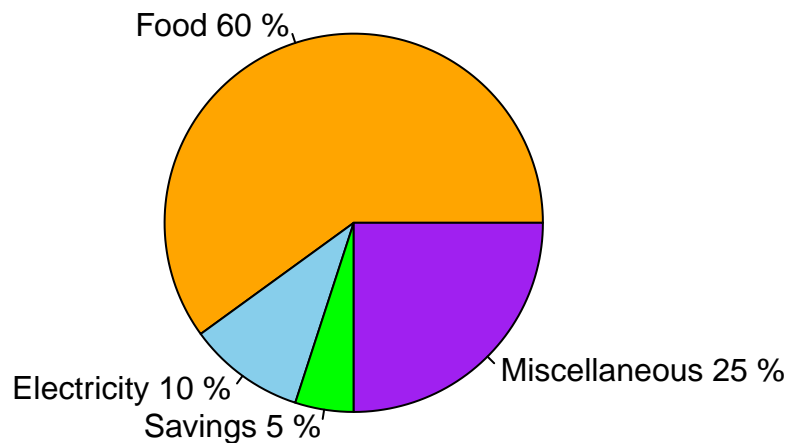


5. The monthly income of Dela Cruz family was spent on the following: Food Electricity Savings Miscellaneous 60 10 5 25

- 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.

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

Dela Cruz Family Monthly Expenses



6. Use the iris dataset. `data(iris)`

- a. Check for the structure of the dataset using the `str()` function. Describe what you have seen in the output. #It will show that iris is a data frame with 150 observations and 5 variables: Sepal.Length: Numeric, lengths of sepals (in cm). Sepal.Width: Numeric, widths of sepals (in cm). Petal.Length: Numeric, lengths of petals (in cm). Petal.Width: Numeric, widths of petals (in cm). Species: Factor with 3 levels - setosa, versicolor, virginica. This gives us an idea of what data types are present in each column and how many levels the Species factor has. (it is the same on Sir Henry Ofori's Presentation)

```
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?

```
means <- colMeans(subset(iris, select = -Species))
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_counts <- table(iris$Species)

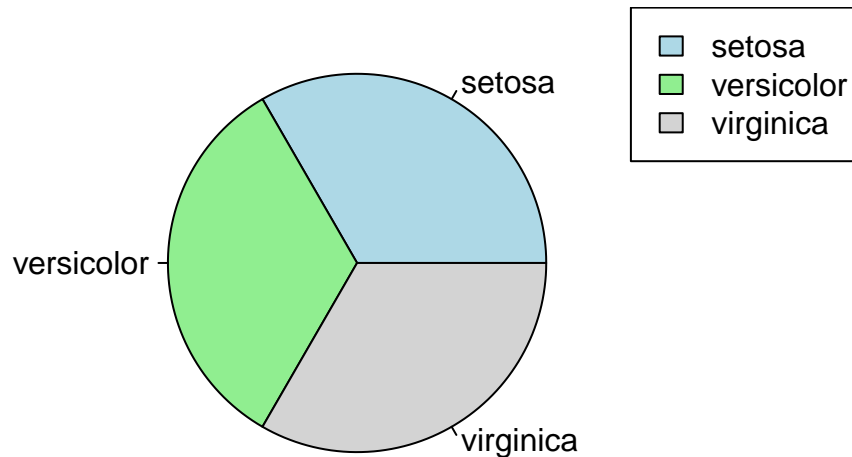
colors <- c("lightblue", "lightgreen", "lightgray")

pie(species_counts,
    main = "Species Distribution in Iris Dataset",
    col = colors,
```

```
labels = names(species_counts))
```

```
legend("topright", legend = names(species_counts), fill = colors)
```

Species Distribution in Iris Dataset



- 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)
```

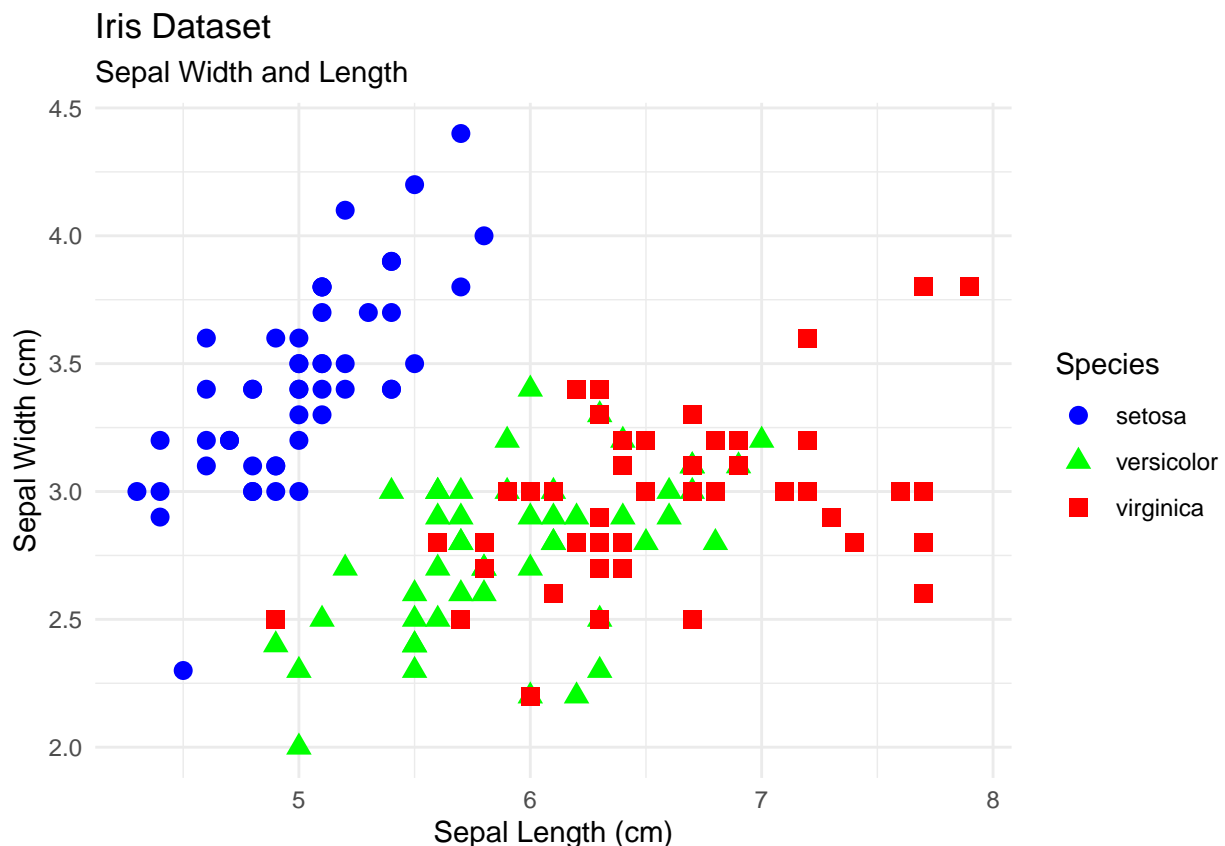
```
##      Sepal.Length Sepal.Width Petal.Length Petal.Width  Species
## 145          6.7         3.3         5.7         2.5  virginica
## 146          6.7         3.0         5.2         2.3  virginica
```

```
## 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.

```
library(ggplot2)

ggplot(data = iris, aes(x = Sepal.Length, y = Sepal.Width, color = Species, shape = Species)) +
  geom_point(size = 3) +
  labs(
    title = "Iris Dataset",
    subtitle = "Sepal Width and Length",
    x = "Sepal Length (cm)",
    y = "Sepal Width (cm)"
  ) +
  theme_minimal() +
  scale_color_manual(values = c("setosa" = "blue", "versicolor" = "green", "virginica" = "red"))
```



- f. Interpret the result. #The scatter plot of Sepal.Length vs. Sepal.Width shows the relationship between these two measurements for each species:

#Setosa: Typically has higher sepal lengths and widths, clustering separately from the other two species.

#Versicolor and Virginica: Overlap more in their sepal dimensions, but Virginica generally has the largest sepal dimensions.

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).

```
library(readxl)

alexa_data <- read_excel("alexa_file.xlsx")

head(alexa_data)

## # A tibble: 6 x 5
##   rating date          variation      verified_reviews      feedback
##   <dbl> <dtm>          <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 playi~ 1
## 4     5 2018-07-31 00:00:00 Charcoal Fabric I have had a lot of f~ 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 a~ 1
```

Figure 4: Snippet of Alexa Variations a. Rename the white and black variants by using gsub() function.

```
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)

# Fix "White" variants
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)

alexa_data$variation[1052:2000]

## [1] "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "WhiteSpot"
## [7] "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "WhiteSpot"
## [13] "WhiteSpot" "BlackSpot" "WhiteSpot" "WhiteSpot" "WhiteSpot" "WhiteSpot" "WhiteSpot"
## [19] "BlackSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [25] "WhiteSpot" "BlackSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [31] "BlackSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot"
## [37] "BlackSpot" "WhiteSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [43] "BlackSpot" "WhiteSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [49] "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [55] "WhiteSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot"
## [61] "BlackSpot" "WhiteSpot" "WhiteSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [67] "BlackSpot" "WhiteSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [73] "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [79] "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [85] "BlackSpot" "WhiteSpot" "WhiteSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [91] "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "WhiteSpot"
## [97] "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot"
## [103] "BlackSpot" "WhiteSpot" "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot"
## [109] "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "WhiteSpot"
## [115] "BlackSpot" "WhiteSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot" "BlackSpot"
```

[illegible]

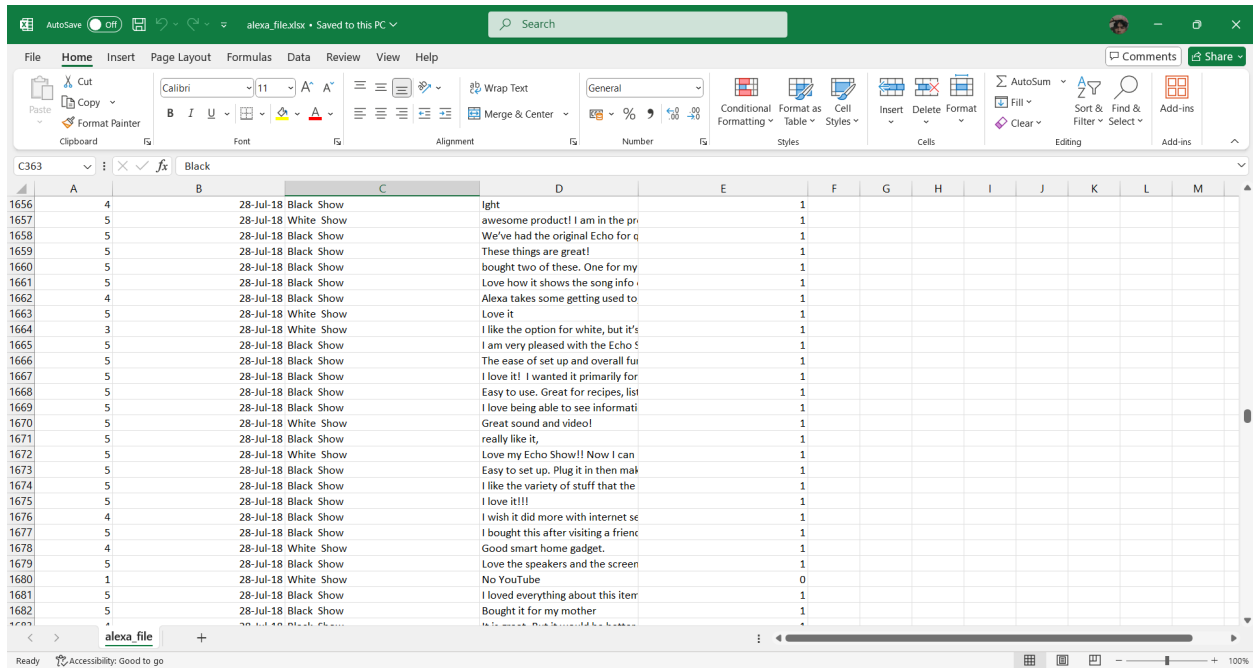
[illegible]

```
## [769] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [775] "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [781] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [787] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [793] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [799] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus"
## [805] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [811] "WhitePlus" "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus"
## [817] "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [823] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [829] "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [835] "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [841] "WhitePlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [847] "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [853] "WhitePlus" "BlackPlus" "WhitePlus" "WhitePlus" "BlackPlus" "BlackPlus"
## [859] "BlackPlus" "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus"
## [865] "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [871] "BlackPlus" "WhitePlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [877] "BlackPlus" "BlackPlus" "BlackPlus" "WhitePlus" "WhitePlus" "BlackPlus"
## [883] "BlackPlus" "BlackPlus" "BlackPlus" "WhitePlus" "WhitePlus" "WhitePlus"
## [889] "BlackPlus" "WhitePlus" "WhitePlus" "BlackPlus" "BlackPlus" "WhitePlus"
## [895] "BlackPlus" "WhitePlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus"
## [901] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [907] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [913] "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus" "WhitePlus" "WhitePlus"
## [919] "BlackPlus" "BlackPlus" "WhitePlus" "WhitePlus" "BlackPlus" "BlackPlus"
## [925] "BlackPlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [931] "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [937] "BlackPlus" "BlackPlus" "WhitePlus" "WhitePlus" "BlackPlus" "BlackPlus"
## [943] "WhitePlus" "BlackPlus" "WhitePlus" "BlackPlus" "BlackPlus" "BlackPlus"
## [949] "BlackPlus"
```

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)

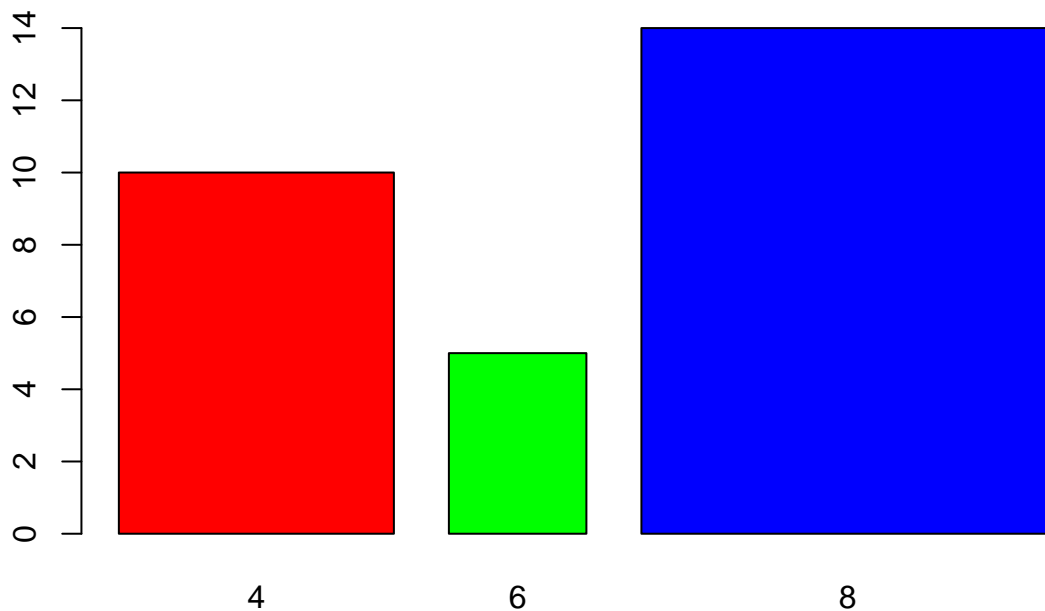
knitr::include_graphics("alexa-file.png")
```



```
values <- c(10, 5, 14)
names <- c(4, 6, 8)
colors <- c("red", "green", "blue")
```

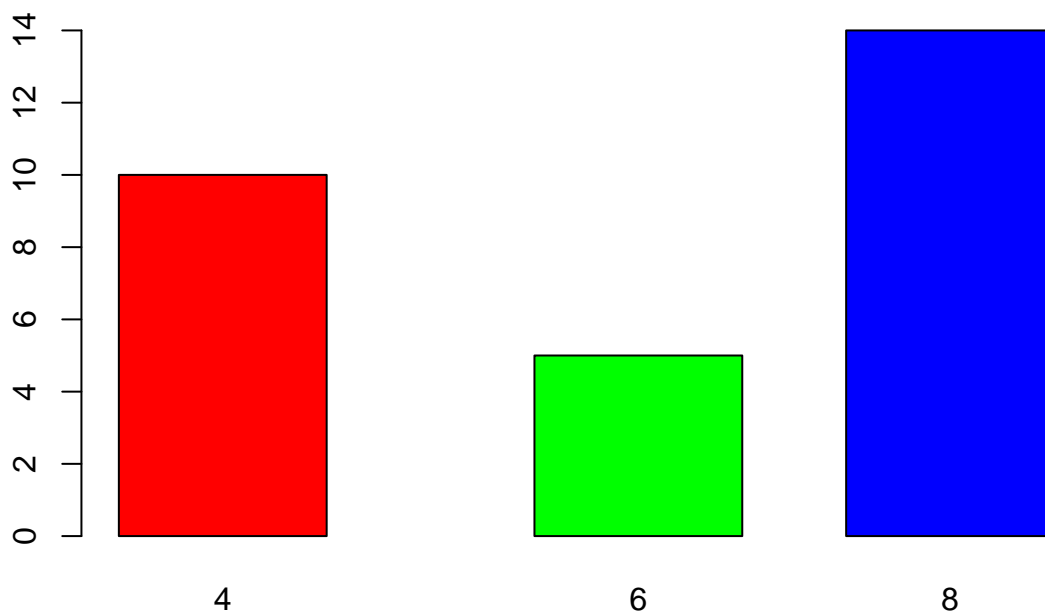
```
barplot(values, names.arg=names, col=colors, main="Change bar width", width=c(1, 0.5, 1.5))
```

Change bar width



```
barplot(values, names.arg=names, col=colors, main="Change space between bars", space=c(0.2, 1, 0.5))
```

Change space between bars



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.

```
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)

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

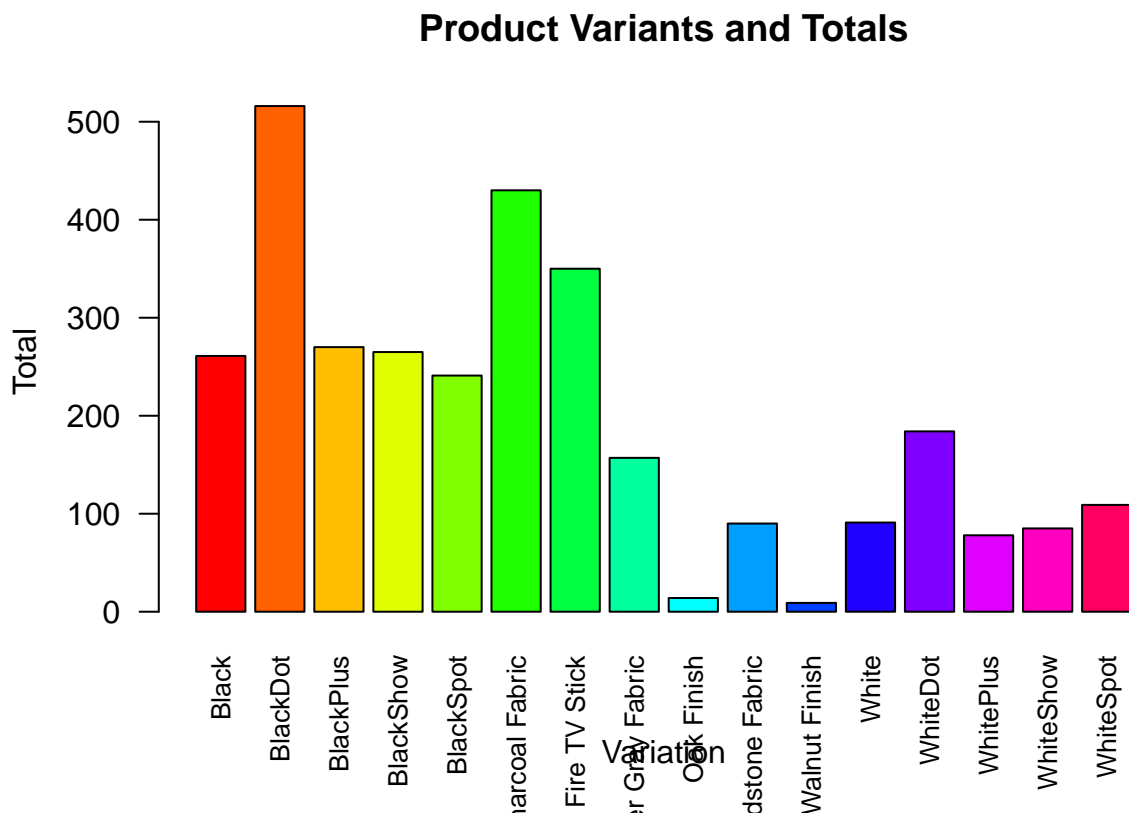
print(variation_counts)
```

```
## # A tibble: 16 x 2
##   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
```

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

```
barplot(
  variation_counts$n,
  names.arg = variation_counts$variation,
  col = rainbow(length(variation_counts$variation)),
  main = "Product Variants and Totals",
  xlab = "Variation",
  ylab = "Total",
  las = 2,
  cex.names = 0.8
)
```



- 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(dplyr)

variation_counts <- alexa_data %>%
  filter(grepl("Black|White", variation)) %>% # Filter for Black and White variations
  count(variation)
```

```

black_counts <- variation_counts %>%
  filter(grepl("Black", variation))
white_counts <- variation_counts %>%
  filter(grepl("White", variation))

par(mfrow = c(1, 2))

variations <- variation_counts$variation
counts <- variation_counts$n

barplot(black_counts$n,
  names.arg = black_counts$variation,
  col = c("black", "pink", "green", "blue", "cyan"),
  main = "Black Variants",
  xlab = "Total Numbers",
  ylab = "Variants",
  ylim = c(0, max(black_counts$n) * 1.2))

barplot(white_counts$n,
  names.arg = white_counts$variation,
  col = c("black", "pink", "green", "blue", "cyan"),
  main = "White Variants",
  xlab = "Total Numbers",
  ylab = "Variants",
  ylim = c(0, max(white_counts$n) * 1.2))

```

