RWorksheet_Alpanghe#4b

Jersey Gabriel E. Alpanghe

2024-10-30

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

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

print(matrixB)
```

```
##      [,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 2.

```
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.

```
# Get the starting number from the user
start_number <- readline("Enter the starting number for the Fibonacci sequence: ")
```

```
## Enter the starting number for the Fibonacci sequence:
```

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

# Check if the input is valid
if (is.na(start_number) || start_number < 1) {
  cat("Invalid input. Please enter a positive integer.\n")
```

```
} else {
  a <- 0
  b <- 1

  cat(a, " ")
    repeat {
    c <- a + b
        cat(c, " ")
        a <- b
    b <- c

    if (c > 500) {
      break
    }
  }
}
```

```
## Invalid input. Please enter a positive integer.
```

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

```
shoes <- read.csv("shoes.csv")
head(shoes)
```

```
##   Shoe.Size Height Gender Shoe.Size.1 Height.1 Gender.1
## 1       6.5   66.0      F        13.0       77        M
## 2       9.0   68.0      F        11.5       72        M
## 3       8.5   64.5      F         8.5       59        F
## 4       8.5   65.0      F         5.0       62        F
## 5      10.5   70.0      M        10.0       72        M
## 6       7.0   64.0      F         6.5       66        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(shoes, Gender == "M")
male_count <- nrow(male_data)

female_data <- subset(shoes, Gender == "F")
female_count <- nrow(female_data)

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

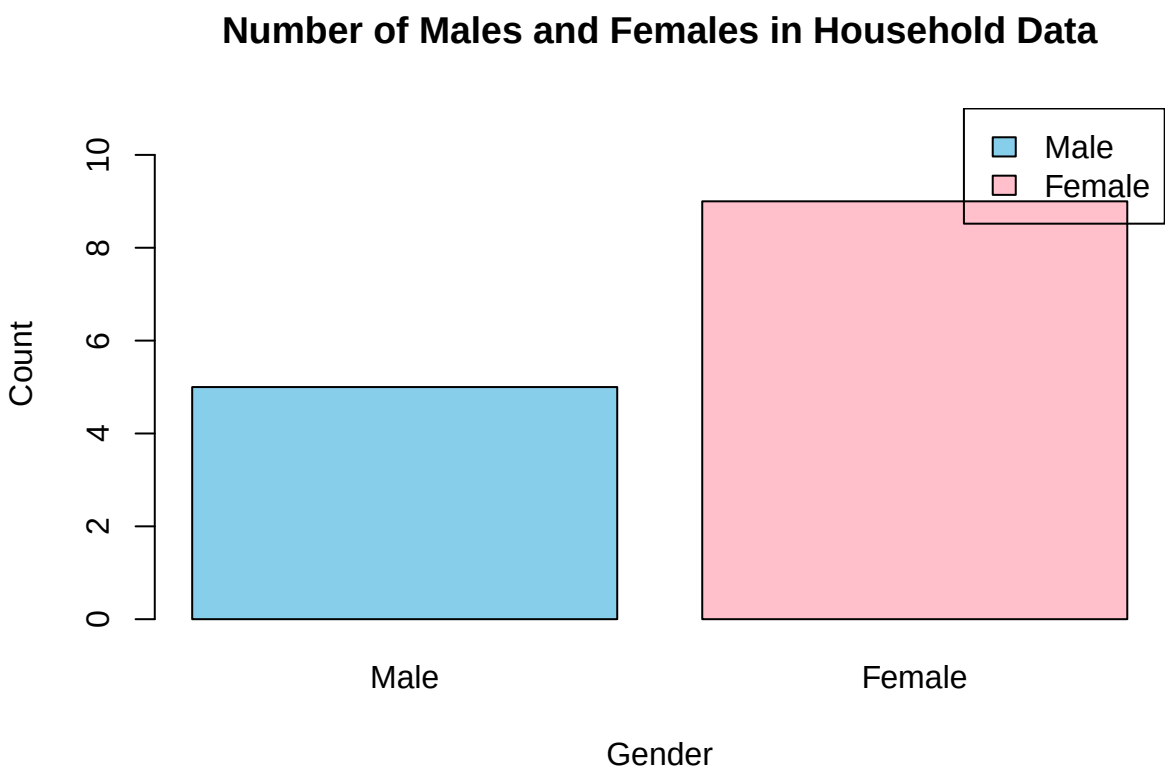
```
## Number of Males:  5
```

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

```
## Number of Females:  9
```

```
# Count the number of Male and Female observations
num_male <- nrow(male_data)
num_female <- nrow(female_data)

# Create a vector with the counts
gender_counts <- c(num_male, num_female)
```

```r
# Define labels and colors for the plot
gender_labels <- c("Male", "Female")
colors <- c("skyblue", "pink")

# Create a bar plot
barplot(
  gender_counts,
  names.arg = gender_labels,
  col = colors,
  main = "Number of Males and Females in Household Data",
  xlab = "Gender",
  ylab = "Count",
  ylim = c(0, max(gender_counts) + 2)
)

# Add a legend
legend("topright", legend = gender_labels, fill = colors)
```

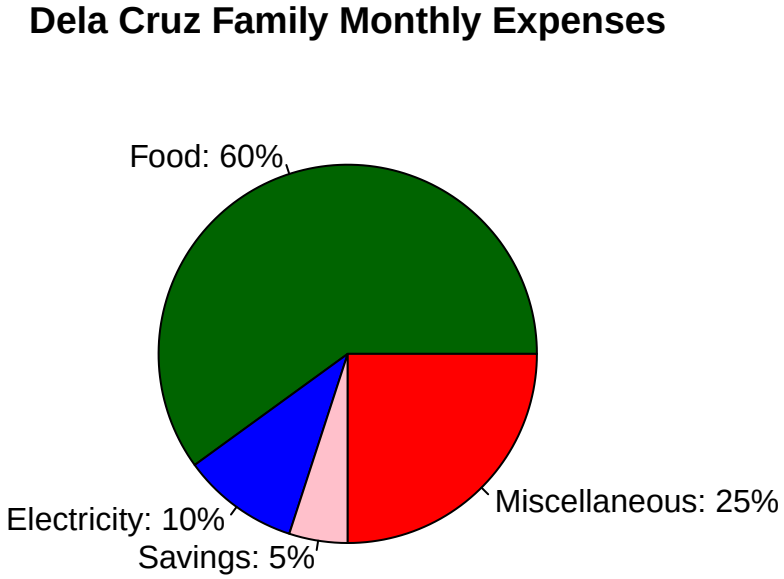


5. The monthly income of Dela Cruz family was spent on the following:

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
expenses <- c(Food = 60, Electricity = 10, Savings = 5, Miscellaneous = 25)

percent_labels <- paste0(names(expenses), ": ", round(expenses / sum(expenses) * 100, 1), "%")

pie(
  expenses,
  labels = percent_labels,
```

```
  main = "Dela Cruz Family Monthly Expenses",
  col = c("darkgreen", "blue", "pink", "red") # Add colors for each category
)
```



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.

```
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?

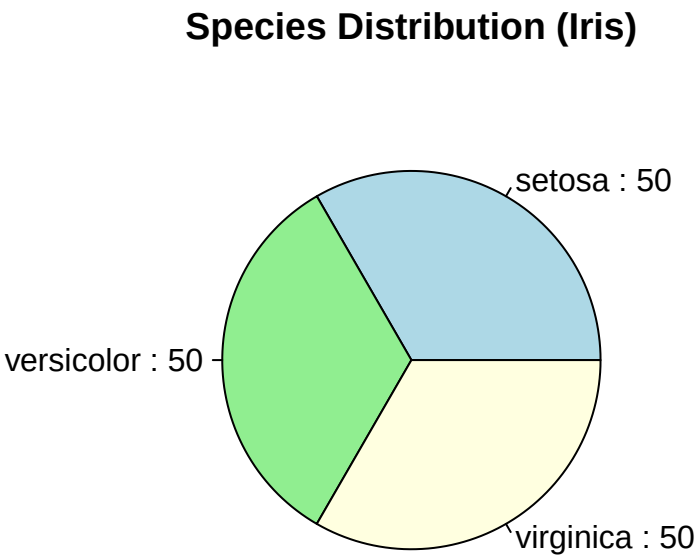
```
mean_values <- colMeans(iris[, 1:4])

mean_values
```

```
## 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 <- table(iris$Species)

pie(
  species, main = "Species Distribution (Iris)",
  col = c("lightblue", "lightgreen", "lightyellow"), labels = paste(names(species), ":", species), lege
)
```

```
## Warning in text.default(1.1 * P$x, 1.1 * P$y, labels[i], xpd = TRUE, adj =
## ifelse(P$x < : "legend.text" is not a graphical parameter
## Warning in text.default(1.1 * P$x, 1.1 * P$y, labels[i], xpd = TRUE, adj =
## ifelse(P$x < : "legend.text" is not a graphical parameter
## Warning in text.default(1.1 * P$x, 1.1 * P$y, labels[i], xpd = TRUE, adj =
## ifelse(P$x < : "legend.text" is not a graphical parameter

## Warning in title(main = main, ...): "legend.text" is not a graphical parameter
```



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 <- 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
```

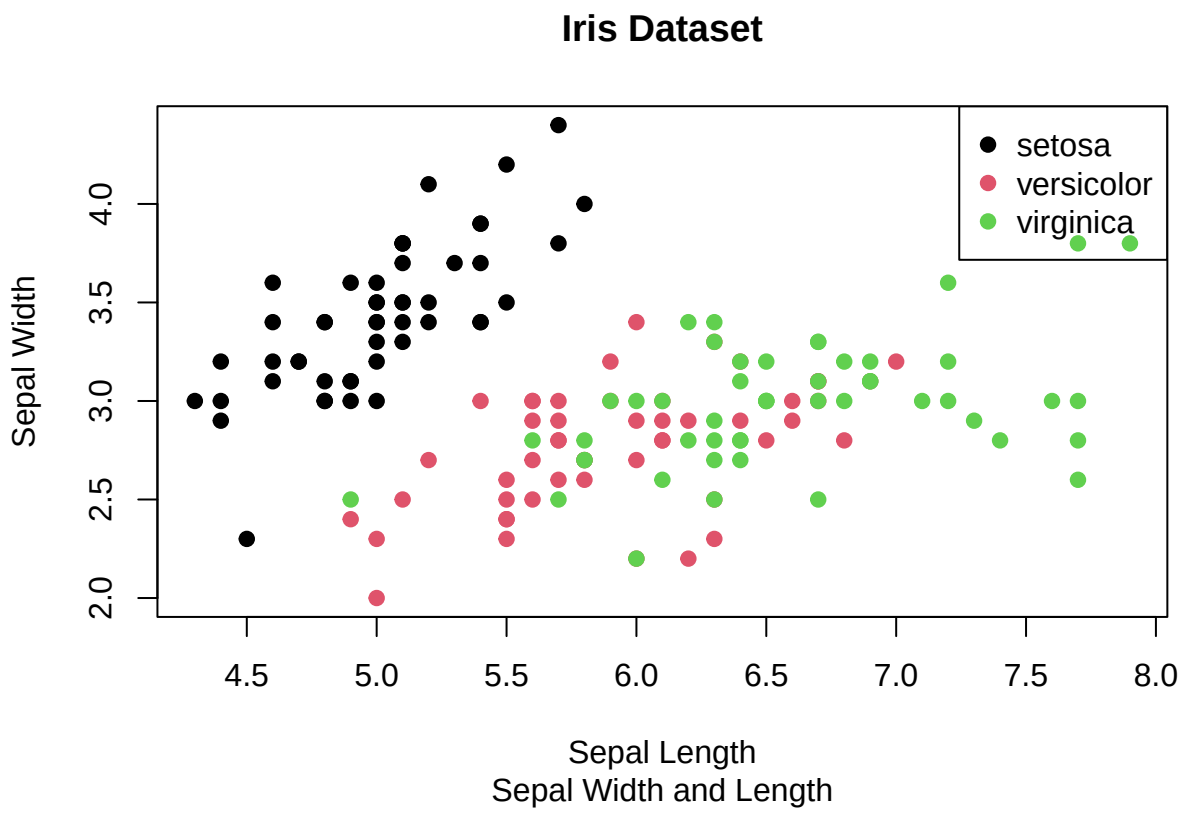
```r
tail(virginica)
```

```
##     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
```

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
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 = 19, # Symbol for points
  col = as.numeric(iris$Species)
)

legend(
  "topright",
  legend = levels(iris$Species),
  col = 1:3, pch = 19
)
```



f. Interpret the result.

The different colors for each species show clear clusters, indicating that the species have distinct sepal measurements. Setosa tends to be well-separated from versicolor and virginica in terms of both Sepal.Length and Sepal.Width. versicolor and virginica, however, may overlap slightly, especially in sepal width, but still display a general trend that helps distinguish them.

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).

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

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

alexa_data$variation <- gsub("Black\\s+Dot", "BlackDot", alexa_data$variation)
alexa_data$variation <- gsub("Black\\s+Plus", "BlackPlus", alexa_data$variation)
alexa_data$variation <- gsub("Black\\s+Show", "BlackShow", alexa_data$variation)
alexa_data$variation <- gsub("Black\\s+Spot", "BlackSpot", alexa_data$variation)

alexa_data$variation <- gsub("White\\s+Dot", "WhiteDot", alexa_data$variation)
alexa_data$variation <- gsub("White\\s+Plus", "WhitePlus", alexa_data$variation)
alexa_data$variation <- gsub("White\\s+Show", "WhiteShow", alexa_data$variation)
alexa_data$variation <- gsub("White\\s+Spot", "WhiteSpot", alexa_data$variation)

head(alexa_data$variation)
```

```
## [1] "Charcoal Fabric"     "Charcoal Fabric"     "Walnut Finish"
## [4] "Charcoal Fabric"     "Charcoal Fabric"     "Heather Gray Fabric"
```

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
```

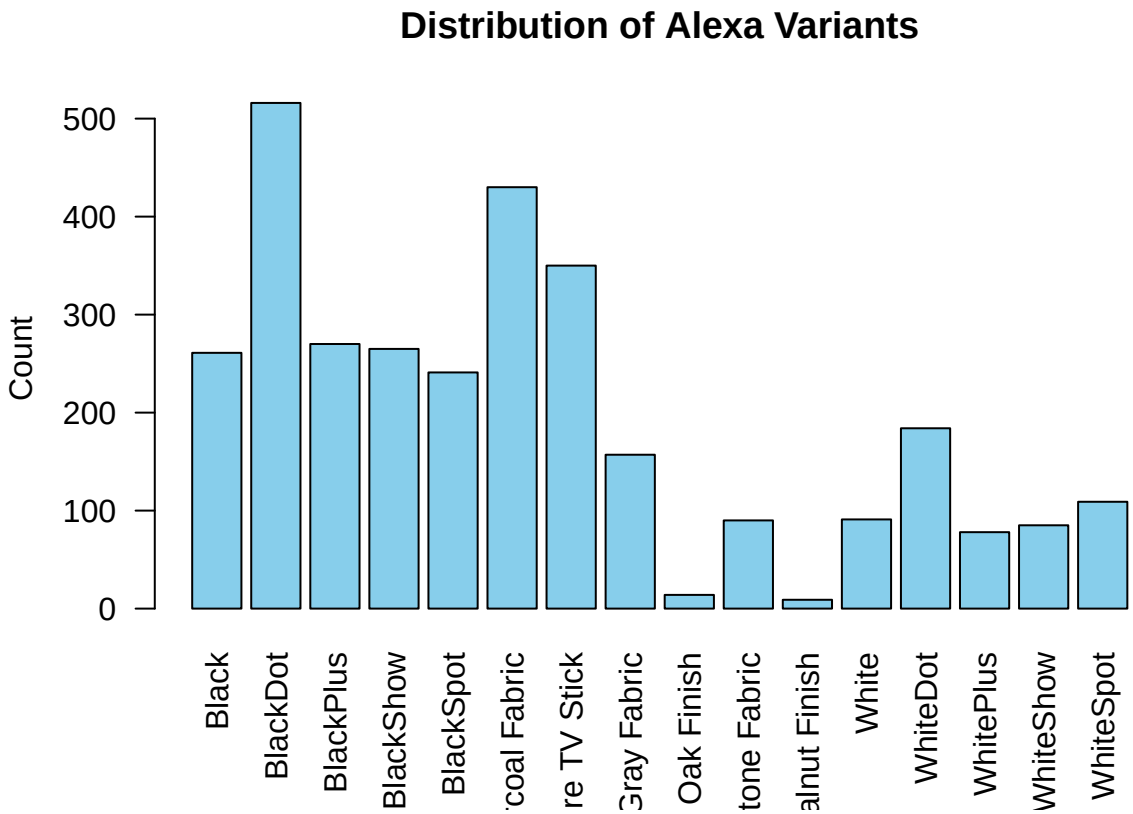
```
variation_counts <- alexa_data %>%
    count(variation)

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,
        main = "Distribution of Alexa Variants",
        col = "skyblue",
        ylab = "Count",
        las = 2)
```



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

```
black_white_counts <- variation_counts %>%
    filter(grepl("Black", variation) | grepl("White", variation))

barplot(height = black_white_counts$n,
```

```
        names.arg = black_white_counts$variation,
        beside = TRUE,
        col = c("black", "grey"),
        main = "Black and White Alexa Variants",
        xlab = "Variations",
        ylab = "Count",
        legend.text = c("Black Variants", "White Variants"))
```