

# RWorksheet\_\_Hallares#4b

Marc Christian P. Hallares

2025-11-17

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

```
vectorA <- 1:5
matrix_result <- matrix(0, nrow = 5, ncol = 5)

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

matrix_result
```

```
##      [,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) {
  for (j in 1:i) {
    cat("*" "\t")
  }
  cat("\n\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.

```

fib_sequence <- function() {
  start <- as.integer(readline(prompt = "Enter starting number: "))

  if (start <= 0) {
    print("Please enter a positive number")
    return()
  }

  a <- 0
  b <- 1
  count <- 0
  result <- c()

  repeat {
    if (count >= start) {
      if (b > 500) break
      result <- c(result, b)
    }

    temp <- a + b
    a <- b
    b <- temp
    count <- count + 1
  }

  print(result)
}

```

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

```

library(readxl)

data <- read_excel("import_march.xlsx")
head(data)

## # A tibble: 6 x 4
##   Students 'Strategy 1' 'Strategy 2' 'Strategy 3'
##   <chr>          <dbl>          <dbl>          <dbl>
## 1 Male              8              10              8
## 2 <NA>              4              8              6
## 3 <NA>              0              6              4
## 4 Female          14              4             15
## 5 <NA>             10              2             12
## 6 <NA>              6              0              9

```

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, Students == "Male")
female_data <- subset(data, Students == "Female")

cat("Number of Male observations:", nrow(male_data), "\n")
```

```
## Number of Male observations: 1
```

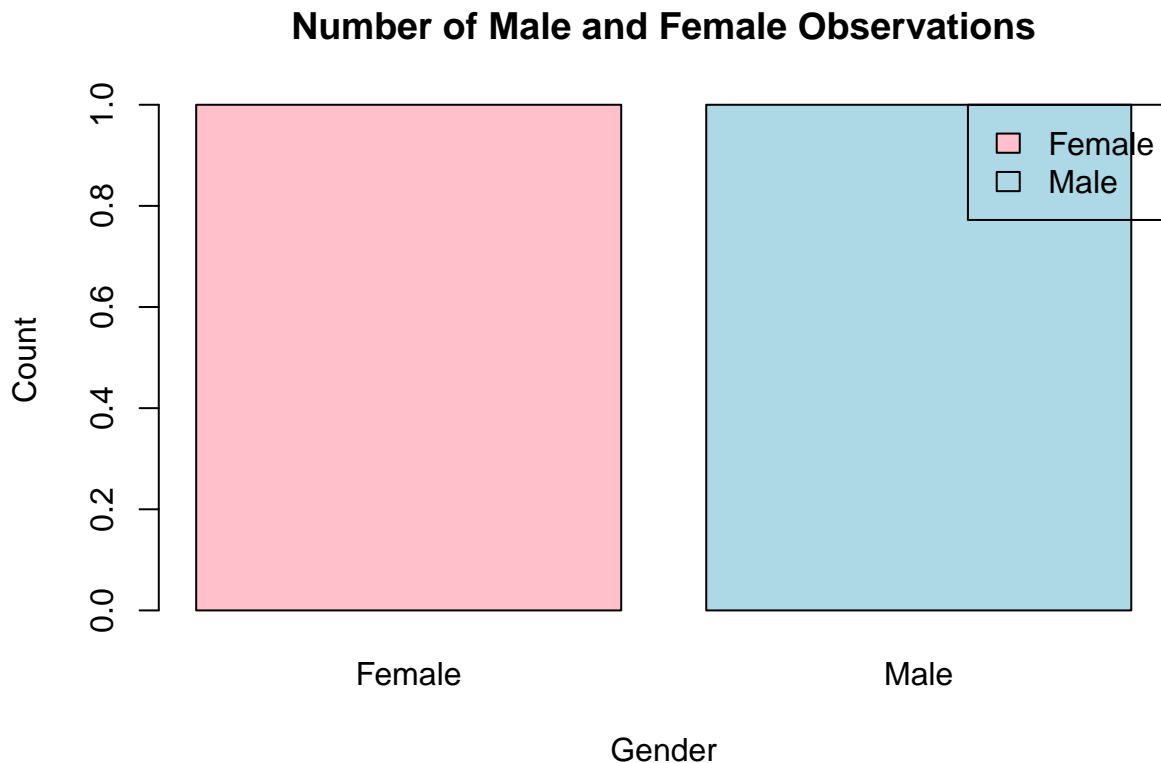
```
cat("Number of Female observations:", nrow(female_data), "\n")
```

```
## Number of Female observations: 1
```

- 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$Students)

barplot(
  gender_counts,
  main = "Number of Male and Female Observations",
  xlab = "Gender",
  ylab = "Count",
  col = c("pink", "lightblue"),
  legend.text = TRUE,
  args.legend = list(x = "topright", legend = c("Female", "Male"))
)
```



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

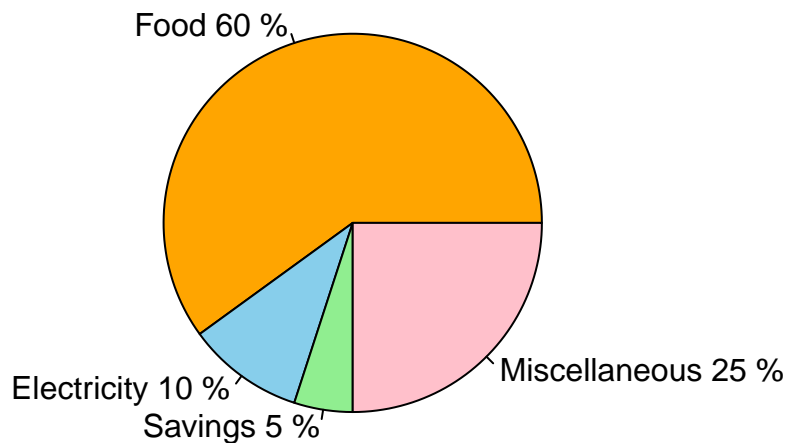
- a. Create a piechart that will include labels in percentage. Add some colors and title of the chart. Write the R scripts and show its output.

```
expenses <- c(60, 10, 5, 25)
categories <- c("Food", "Electricity", "Savings", "Miscellaneous")

percent <- round(expenses / sum(expenses) * 100)
labels <- paste(categories, percent, "%")

pie(
  expenses,
  labels = labels,
  col = c("orange", "skyblue", "lightgreen", "pink"),
  main = "Monthly Income Distribution of Dela Cruz Family"
)
```

## Monthly Income Distribution 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 <- data.frame(
  Sepal.Length = mean(iris$Sepal.Length),
  Sepal.Width = mean(iris$Sepal.Width),
  Petal.Length = mean(iris$Petal.Length),
  Petal.Width = mean(iris$Petal.Width)
)
```

```
iris_means
```

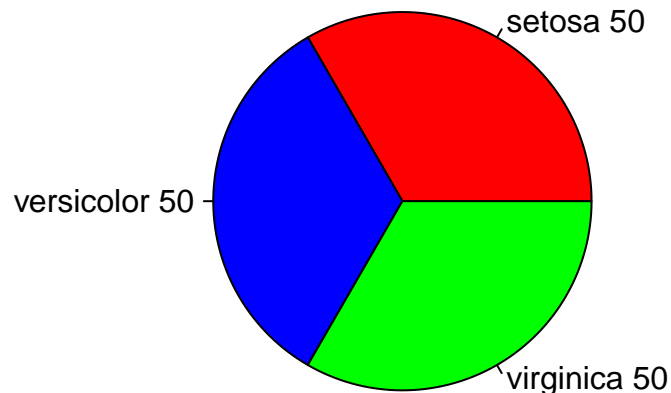
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
## 1      5.843333      3.057333      3.758      1.199333
```

- 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)

pie(
  species_counts,
  main = "Iris Species Distribution",
  col = c("red", "blue", "green"),
  labels = paste(names(species_counts), species_counts)
)
```

## Iris Species Distribution



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

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

tail(setosa, 6)
```

```
##   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, 6)
```

```
##      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
```

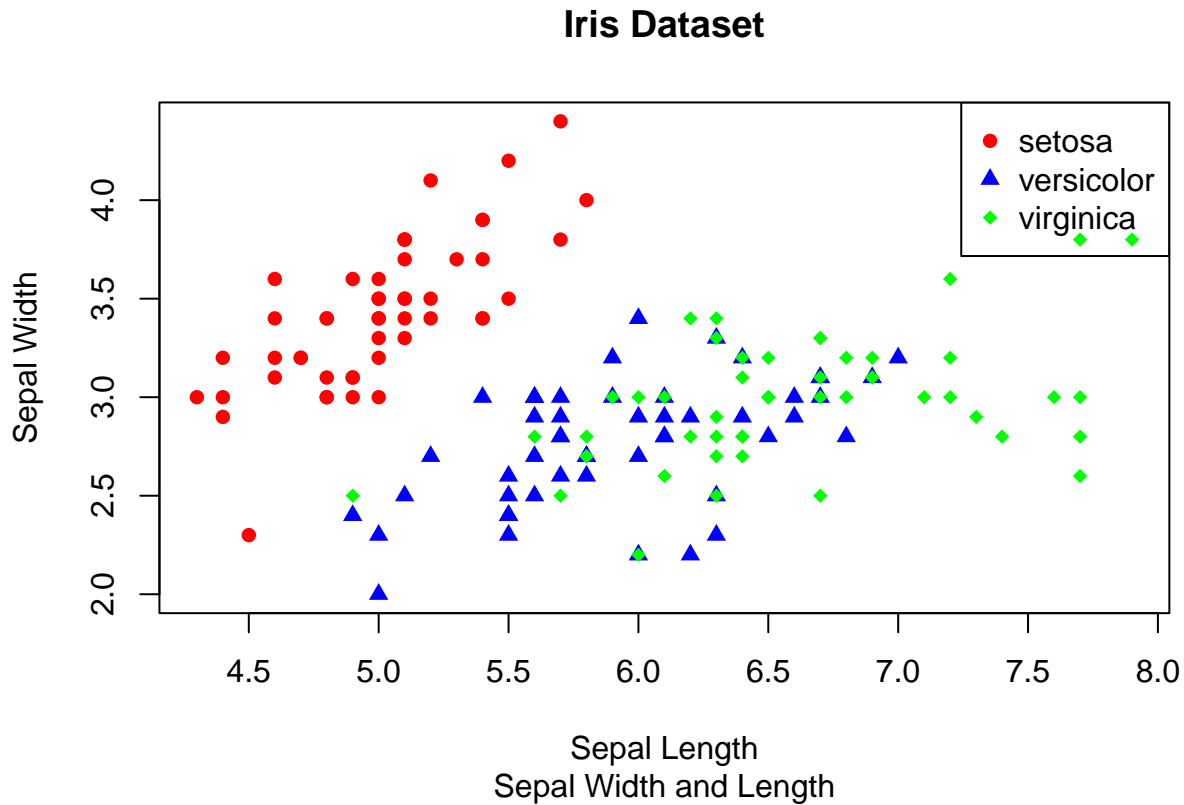
```
tail(virginica, 6)
```

```
##      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.

```
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$Species)],
  col = c("red", "blue", "green")[as.numeric(iris$Species)]
)

legend(
  "topright",
  legend = levels(iris$Species),
  pch = c(16, 17, 18),
  col = c("red", "blue", "green")
)
```



f. Interpret the result.

"The scatterplot shows clear species separation. Setosa has short, wide sepals, forming a distinct cluster."

```
## [1] "The scatterplot shows clear species separation. Setosa has short, wide sepals, forming a distinct cluster."
```

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, BlackSpot). 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)
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
```

```

alexas <- read_xlsx("alexas-file.xlsx")

alexas$variation <- gsub("Black Dot", "Black Dot", alexas$variation)
alexas$variation <- gsub("Black Plus", "Black Plus", alexas$variation)
alexas$variation <- gsub("Black Show", "Black Show", alexas$variation)
alexas$variation <- gsub("Black Spot", "Black Spot", alexas$variation)
alexas$variation <- gsub("White Dot", "White Dot", alexas$variation)
alexas$variation <- gsub("White Plus", "White Plus", alexas$variation)
alexas$variation <- gsub("White Show", "White Show", alexas$variation)
alexas$variation <- gsub("White Spot", "White Spot", alexas$variation)

head(alexas, 10)

```

```

## # A tibble: 8 x 5
##   rating date      variation verified_reviews feedback
##   <dbl> <chr>      <chr>      <chr>      <dbl>
## 1     5 2018-07-30 Black Dot  It works great!!      1
## 2     5 2018-07-30 Black Plus PHENOMENAL             1
## 3     4 2018-07-30 Black Show I used it to control my smart devices.      1
## 4     3 2018-07-30 Black Spot Very convenient         1
## 5     5 2018-07-31 White Dot  I love it!              1
## 6     4 2018-07-31 White Plus Nice sound quality         1
## 7     5 2018-07-31 White Show Great screen!           1
## 8     4 2018-07-31 White Spot Easy to use              1

```

- 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?

```

variations <- alexas %>% count(variation)
variations

```

```

## # A tibble: 8 x 2
##   variation      n
##   <chr>    <int>
## 1 Black Dot      1
## 2 Black Plus     1
## 3 Black Show     1
## 4 Black Spot     1
## 5 White Dot      1
## 6 White Plus     1
## 7 White Show     1
## 8 White Spot     1

```

```

save(variations, 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.

```

barplot(
  variations$n,
  names.arg = variations$variation,

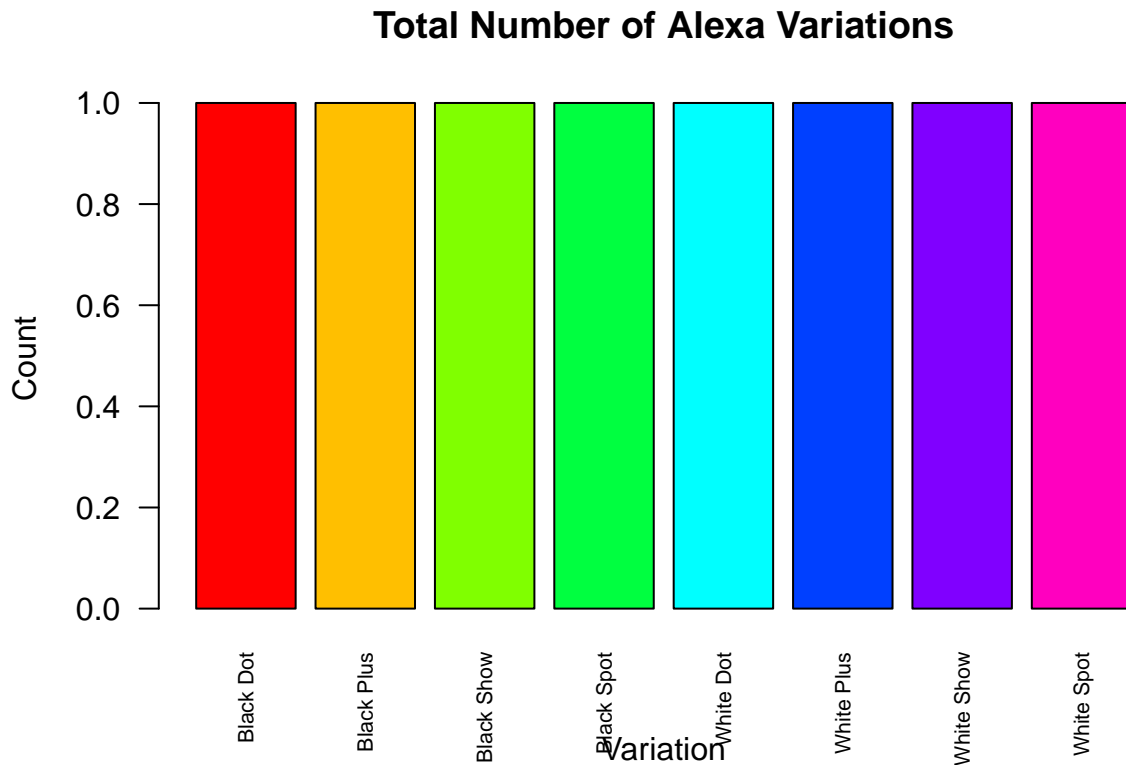
```



```

main = "Total Number of Alexa Variations",
xlab = "Variation",
ylab = "Count",
col = rainbow(nrow(variations)),
las = 2,
cex.names = 0.7
)

```



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

```

black_vars <- variations[grepl("Black", variations$variation), ]
white_vars <- variations[grepl("White", variations$variation), ]

par(mfrow = c(1, 2))

barplot(
  black_vars$n,
  names.arg = black_vars$variation,
  main = "Black Variations",
  xlab = "Variation",
  ylab = "Count",
  col = "black",
  las = 2,
  cex.names = 0.8
)

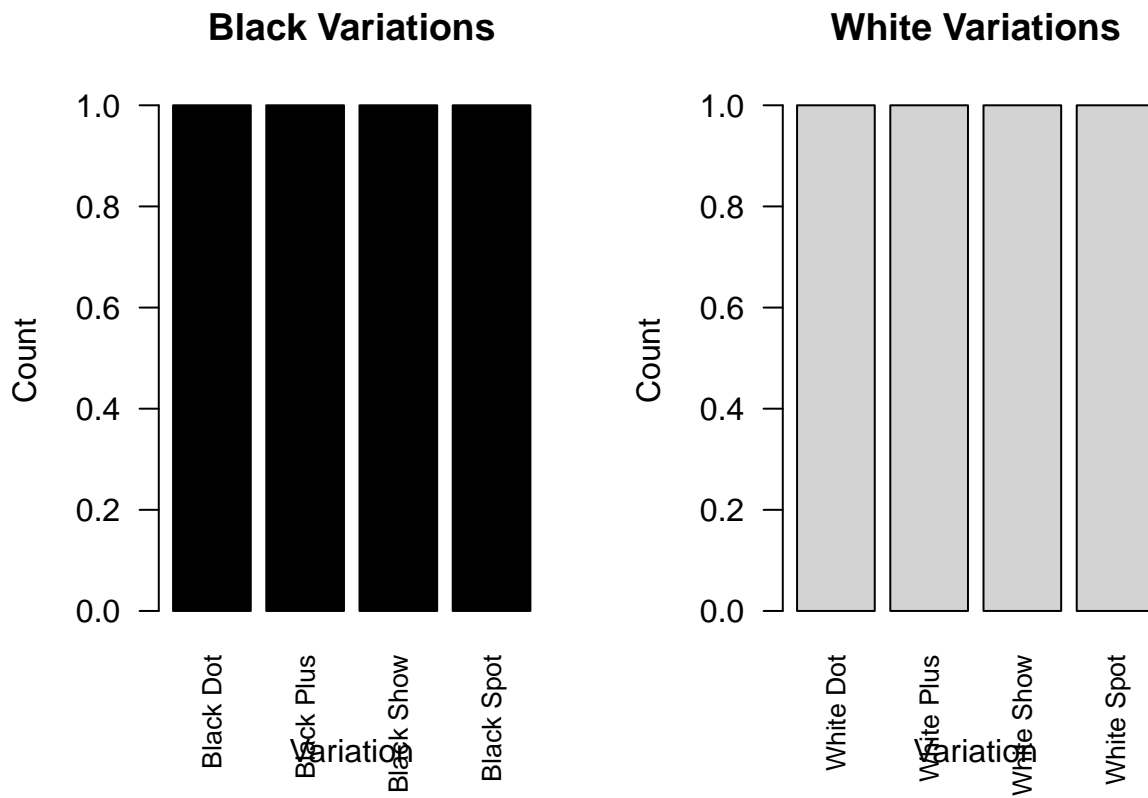
barplot(

```

```

white_vars$n,
names.arg = white_vars$variation,
main = "White Variations",
xlab = "Variation",
ylab = "Count",
col = "lightgray",
las = 2,
cex.names = 0.8
)

```



```

par(mfrow = c(1, 1))

```