

Rworksheet_Gregorio#4b

2023-11-08

#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)

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

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

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.

```
for (i in 1:5) {
  cat(paste0("\n", rep("*", i), "\n"), "\n")
}
```

```
## "*"
## "*" "*"
## "*" "*" "*"
## "*" "*" "*" "*"
## "*" "*" "*" "*" "*"
```

#3. n <- as.numeric(readline(prompt = "Enter a number to start the Fibonacci sequence:"))

a <- 0 b <- 1 c <- a + b

repeat { if (c > 500) { break } if (a == 0 & b == 1) { cat(b, " ") cat(c, " ") a <- b b <- c c <- a + b }

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? Show your codes and its result

```
ShoesData <- read.csv("Shoe sizes.csv")
head(ShoesData)
```

```
##   X shoe_size height Gender
## 1 1         6.5   66.0     F
## 2 2         9.0   68.0     F
## 3 3         8.5   64.5     F
```

```
## 4 4      8.5   65.0    F
## 5 5     10.5   70.0    M
## 6 6      7.0   64.0    F
```

4b. 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.

```
maleSub <- subset(ShoesData, Gender == "M")
femSub <- subset(ShoesData, Gender == "F")

cat("The number of observation in male subset:", nrow(maleSub), "\n")

## The number of observation in male subset: 14

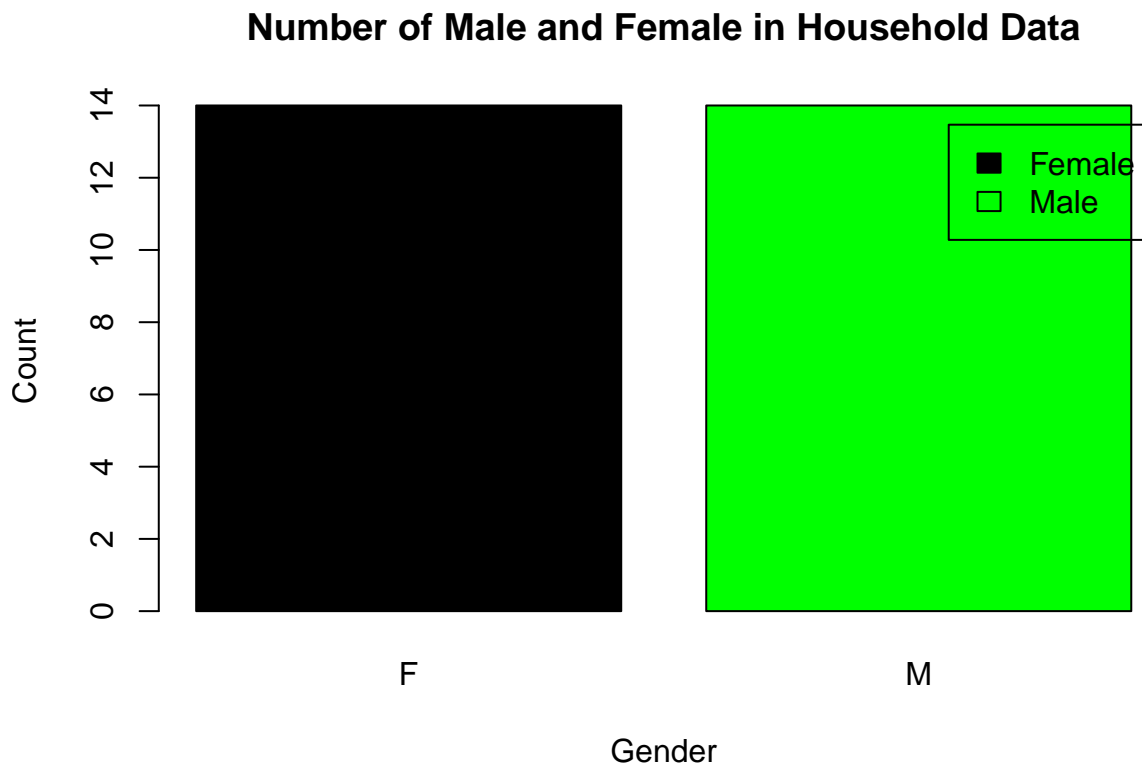
cat("The number of observation in female subset:", nrow(femSub), "\n")
```

```
## The number of observation in female subset: 14
```

4c. 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

```
GenderMF <- table(ShoesData$Gender)

barplot(GenderMF,
  main = "Number of Male and Female in Household Data",
  xlab = "Gender",
  ylab = "Count",
  col = c("black", "green"),
  legend.text = c("Female", "Male"))
```



#5.

```

spending_data <- data.frame(
  Category = c("Food", "Electricity", "Savings", "Miscellaneous"),
  Value = c(60, 10, 5, 25)
)

spending_data$Percentage <- spending_data$Value / sum(spending_data$Value) * 100

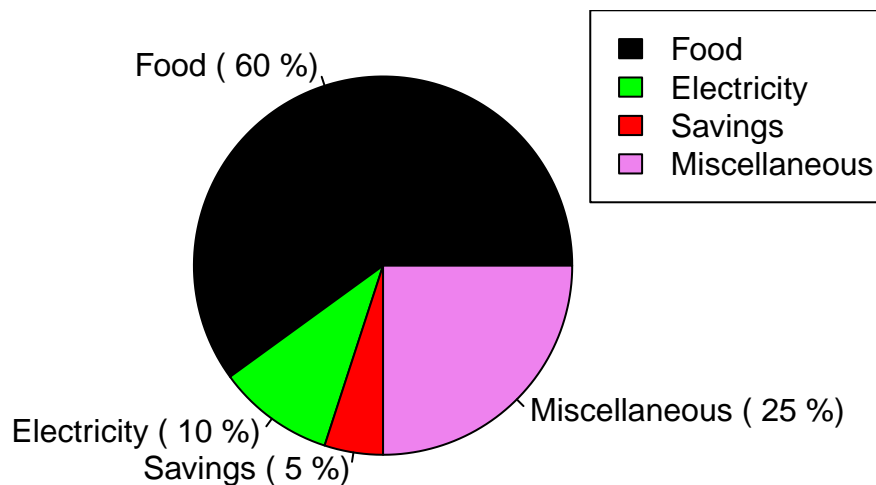
colors <- c("black", "green", "red", "violet")

pie(spending_data$Value,
  labels = paste(spending_data$Category, "(", spending_data$Percentage, "%)"),
  col = colors,
  main = "Monthly Income Spending of Dela Cruz Family")

legend("topright", spending_data$Category, fill = colors)

```

Monthly Income Spending of Dela Cruz Family



6. Use the iris dataset

```

data(iris)
data

## function (... , list = character(), package = NULL, lib.loc = NULL,
##   verbose = getOption("verbose"), envir = .GlobalEnv, overwrite = TRUE)
## {
##   fileExt <- function(x) {
##     db <- grepl("\\.[^.]+"\\.(gz|bz2|xz)$", x)
##     ans <- sub(".*\\.", "", x)
##     ans[db] <- sub(".*\\."\\.[^.]+"\\.(gz|bz2|xz)$", "\\1\\2",
##       x[db])
##     ans
##   }
##   my_read_table <- function(...) {
##     lcc <- Sys.getlocale("LC_COLLATE")
##     on.exit(Sys.setlocale("LC_COLLATE", lcc))
##     Sys.setlocale("LC_COLLATE", "C")
##     read.table(...)

```

```

## }
## stopifnot(is.character(list))
## names <- c(as.character(substitute(list(...))[-1L]), list)
## if (!is.null(package)) {
##   if (!is.character(package))
##     stop("'package' must be a character vector or NULL")
## }
## paths <- find.package(package, lib.loc, verbose = verbose)
## if (is.null(lib.loc))
##   paths <- c(path.package(package, TRUE), if (!length(package)) getwd(),
##             paths)
## paths <- unique(normalizePath(paths[file.exists(paths)]))
## paths <- paths[dir.exists(file.path(paths, "data"))]
## dataExts <- tools:::make_file_exts("data")
## if (length(names) == 0L) {
##   db <- matrix(character(), nrow = 0L, ncol = 4L)
##   for (path in paths) {
##     entries <- NULL
##     packageName <- if (file_test("-f", file.path(path,
##           "DESCRIPTION")))
##       basename(path)
##     else "."
##     if (file_test("-f", INDEX <- file.path(path, "Meta",
##           "data.rds"))) {
##       entries <- readRDS(INDEX)
##     }
##     else {
##       dataDir <- file.path(path, "data")
##       entries <- tools::list_files_with_type(dataDir,
##             "data")
##       if (length(entries)) {
##         entries <- unique(tools::file_path_sans_ext(basename(entries)))
##         entries <- cbind(entries, "")
##       }
##     }
##     if (NROW(entries)) {
##       if (is.matrix(entries) && ncol(entries) == 2L)
##         db <- rbind(db, cbind(packageName, dirname(path),
##               entries))
##       else warning(gettextf("data index for package %s is invalid and will be ignored",
##             sQuote(packageName)), domain = NA, call. = FALSE)
##     }
##   }
##   colnames(db) <- c("Package", "LibPath", "Item", "Title")
##   footer <- if (missing(package))
##     paste0("Use ", sQuote(paste("data(package = ", ".packages(all.available = TRUE)))"),
##           "\n", "to list the data sets in all *available* packages.")
##   else NULL
##   y <- list(title = "Data sets", header = NULL, results = db,
##             footer = footer)
##   class(y) <- "packageIQR"
##   return(y)
## }
## paths <- file.path(paths, "data")

```

```

## for (name in names) {
##   found <- FALSE
##   for (p in paths) {
##     tmp_env <- if (overwrite)
##       enviro
##     else new.env()
##     if (file_test("-f", file.path(p, "Rdata.rds"))) {
##       rds <- readRDS(file.path(p, "Rdata.rds"))
##       if (name %in% names(rds)) {
##         found <- TRUE
##         if (verbose)
##           message(sprintf("name=%s:\t found in Rdata.rds",
##             name), domain = NA)
##         thispkg <- sub(".*(?:[/]*)/data$", "\\1", p)
##         thispkg <- sub("_.*$", "", thispkg)
##         thispkg <- paste0("package:", thispkg)
##         objs <- rds[[name]]
##         lazyLoad(file.path(p, "Rdata"), enviro = tmp_env,
##           filter = function(x) x %in% objs)
##         break
##       }
##     else if (verbose)
##       message(sprintf("name=%s:\t NOT found in names() of Rdata.rds, i.e.,\n\t%s\n",
##         name, paste(names(rds), collapse = ",")),
##         domain = NA)
##   }
##   if (file_test("-f", file.path(p, "Rdata.zip"))) {
##     warning("zipped data found for package ", sQuote(basename(dirname(p))),
##       ".\nThat is defunct, so please re-install the package.",
##       domain = NA)
##     if (file_test("-f", fp <- file.path(p, "filelist")))
##       files <- file.path(p, scan(fp, what = "", quiet = TRUE))
##     else {
##       warning(gettextf("file 'filelist' is missing for directory %s",
##         sQuote(p)), domain = NA)
##       next
##     }
##   }
##   else {
##     files <- list.files(p, full.names = TRUE)
##   }
##   files <- files[grepl(name, files, fixed = TRUE)]
##   if (length(files) > 1L) {
##     o <- match(fileExt(files), dataExts, nomatch = 100L)
##     paths0 <- dirname(files)
##     paths0 <- factor(paths0, levels = unique(paths0))
##     files <- files[order(paths0, o)]
##   }
##   if (length(files)) {
##     for (file in files) {
##       if (verbose)
##         message("name=", name, ":\t file= ...", .Platform$file.sep,
##           basename(file), ":\t", appendLF = FALSE,
##           domain = NA)

```

```

##           ext <- fileExt(file)
##           if (basename(file) != paste0(name, ".", ext))
##             found <- FALSE
##           else {
##             found <- TRUE
##             zfile <- file
##             zipname <- file.path(dirname(file), "Rdata.zip")
##             if (file.exists(zipname)) {
##               Rdatadir <- tempfile("Rdata")
##               dir.create(Rdatadir, showWarnings = FALSE)
##               topic <- basename(file)
##               rc <- .External(C_unzip, zipname, topic,
##                             Rdatadir, FALSE, TRUE, FALSE, FALSE)
##               if (rc == 0L)
##                 zfile <- file.path(Rdatadir, topic)
##             }
##             if (zfile != file)
##               on.exit(unlink(zfile))
##             switch(ext, R = , r = {
##               library("utils")
##               sys.source(zfile, chdir = TRUE, envir = tmp_env)
##             }, RData = , rdata = , rda = load(zfile,
##             envir = tmp_env), TXT = , txt = , tab = ,
##             tab.gz = , tab.bz2 = , tab.xz = , txt.gz = ,
##             txt.bz2 = , txt.xz = assign(name, my_read_table(zfile,
##             header = TRUE, as.is = FALSE), envir = tmp_env),
##             CSV = , csv = , csv.gz = , csv.bz2 = ,
##             csv.xz = assign(name, my_read_table(zfile,
##             header = TRUE, sep = ";", as.is = FALSE),
##             envir = tmp_env), found <- FALSE)
##           }
##           if (found)
##             break
##         }
##         if (verbose)
##           message(if (!found)
##                 "*NOT* ", "found", domain = NA)
##       }
##       if (found)
##         break
##     }
##   if (!found) {
##     warning(gettextf("data set %s not found", sQuote(name)),
##            domain = NA)
##   }
##   else if (!overwrite) {
##     for (o in ls(envir = tmp_env, all.names = TRUE)) {
##       if (exists(o, envir = envir, inherits = FALSE))
##         warning(gettextf("an object named %s already exists and will not be overwritten",
##                         sQuote(o)))
##       else assign(o, get(o, envir = tmp_env, inherits = FALSE),
##                  envir = envir)
##     }
##   }
##   rm(tmp_env)

```

```
##      }
##    }
##    invisible(names)
## }
## <bytecode: 0x55c8177de848>
## <environment: namespace:utils>
```

6a. Check for the structure of the dataset using the `str()` function. Describe what you have seen in the output.

```
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 dataset contains information on iris blossoms. It contains information on the length and width of

6b. 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?

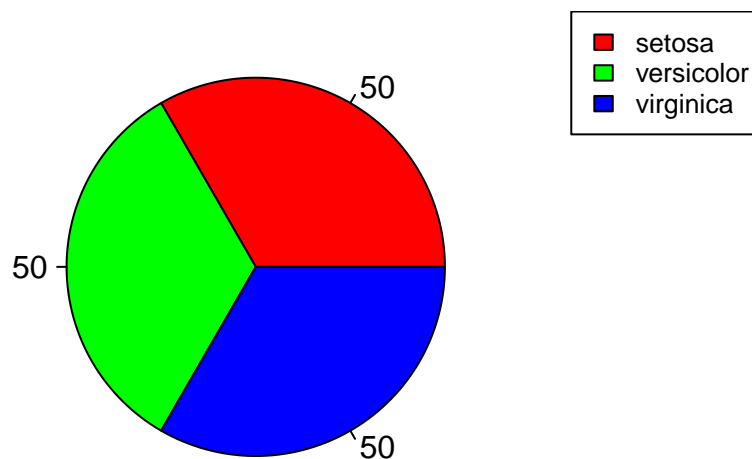
```
meanOfFlowers <- colMeans(iris[,1:4])
meanOfFlowers
```

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

6c. Create a pie chart for the Species distribution. Add title, legends, and colors. Write the R script and its result.

```
species_count <- table(iris$Species)
pie(species_count, labels = species_count, col = rainbow(length(species_count)), main = "Species Distribution",
legend("topright", names(species_count), cex = 0.8, fill = rainbow(length(species_count))))
```

Species Distribution



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

```
# Subset the iris data set into the three species.
setosa_subset <- subset(iris, Species == "setosa")
versicolor_subset <- subset(iris, Species == "versicolor")
virginica_subset <- subset(iris, Species == "virginica")
```

```
setosa_subset
```

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 1	5.1	3.5	1.4	0.2	setosa
## 2	4.9	3.0	1.4	0.2	setosa
## 3	4.7	3.2	1.3	0.2	setosa
## 4	4.6	3.1	1.5	0.2	setosa
## 5	5.0	3.6	1.4	0.2	setosa
## 6	5.4	3.9	1.7	0.4	setosa
## 7	4.6	3.4	1.4	0.3	setosa
## 8	5.0	3.4	1.5	0.2	setosa
## 9	4.4	2.9	1.4	0.2	setosa
## 10	4.9	3.1	1.5	0.1	setosa
## 11	5.4	3.7	1.5	0.2	setosa
## 12	4.8	3.4	1.6	0.2	setosa
## 13	4.8	3.0	1.4	0.1	setosa
## 14	4.3	3.0	1.1	0.1	setosa
## 15	5.8	4.0	1.2	0.2	setosa
## 16	5.7	4.4	1.5	0.4	setosa
## 17	5.4	3.9	1.3	0.4	setosa
## 18	5.1	3.5	1.4	0.3	setosa
## 19	5.7	3.8	1.7	0.3	setosa
## 20	5.1	3.8	1.5	0.3	setosa
## 21	5.4	3.4	1.7	0.2	setosa
## 22	5.1	3.7	1.5	0.4	setosa
## 23	4.6	3.6	1.0	0.2	setosa
## 24	5.1	3.3	1.7	0.5	setosa
## 25	4.8	3.4	1.9	0.2	setosa
## 26	5.0	3.0	1.6	0.2	setosa
## 27	5.0	3.4	1.6	0.4	setosa
## 28	5.2	3.5	1.5	0.2	setosa
## 29	5.2	3.4	1.4	0.2	setosa
## 30	4.7	3.2	1.6	0.2	setosa
## 31	4.8	3.1	1.6	0.2	setosa
## 32	5.4	3.4	1.5	0.4	setosa
## 33	5.2	4.1	1.5	0.1	setosa
## 34	5.5	4.2	1.4	0.2	setosa
## 35	4.9	3.1	1.5	0.2	setosa
## 36	5.0	3.2	1.2	0.2	setosa
## 37	5.5	3.5	1.3	0.2	setosa
## 38	4.9	3.6	1.4	0.1	setosa
## 39	4.4	3.0	1.3	0.2	setosa
## 40	5.1	3.4	1.5	0.2	setosa
## 41	5.0	3.5	1.3	0.3	setosa
## 42	4.5	2.3	1.3	0.3	setosa
## 43	4.4	3.2	1.3	0.2	setosa
## 44	5.0	3.5	1.6	0.6	setosa
## 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

versicolor_subset

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 51	7.0	3.2	4.7	1.4	versicolor
## 52	6.4	3.2	4.5	1.5	versicolor
## 53	6.9	3.1	4.9	1.5	versicolor
## 54	5.5	2.3	4.0	1.3	versicolor
## 55	6.5	2.8	4.6	1.5	versicolor
## 56	5.7	2.8	4.5	1.3	versicolor
## 57	6.3	3.3	4.7	1.6	versicolor
## 58	4.9	2.4	3.3	1.0	versicolor
## 59	6.6	2.9	4.6	1.3	versicolor
## 60	5.2	2.7	3.9	1.4	versicolor
## 61	5.0	2.0	3.5	1.0	versicolor
## 62	5.9	3.0	4.2	1.5	versicolor
## 63	6.0	2.2	4.0	1.0	versicolor
## 64	6.1	2.9	4.7	1.4	versicolor
## 65	5.6	2.9	3.6	1.3	versicolor
## 66	6.7	3.1	4.4	1.4	versicolor
## 67	5.6	3.0	4.5	1.5	versicolor
## 68	5.8	2.7	4.1	1.0	versicolor
## 69	6.2	2.2	4.5	1.5	versicolor
## 70	5.6	2.5	3.9	1.1	versicolor
## 71	5.9	3.2	4.8	1.8	versicolor
## 72	6.1	2.8	4.0	1.3	versicolor
## 73	6.3	2.5	4.9	1.5	versicolor
## 74	6.1	2.8	4.7	1.2	versicolor
## 75	6.4	2.9	4.3	1.3	versicolor
## 76	6.6	3.0	4.4	1.4	versicolor
## 77	6.8	2.8	4.8	1.4	versicolor
## 78	6.7	3.0	5.0	1.7	versicolor
## 79	6.0	2.9	4.5	1.5	versicolor
## 80	5.7	2.6	3.5	1.0	versicolor
## 81	5.5	2.4	3.8	1.1	versicolor
## 82	5.5	2.4	3.7	1.0	versicolor
## 83	5.8	2.7	3.9	1.2	versicolor
## 84	6.0	2.7	5.1	1.6	versicolor
## 85	5.4	3.0	4.5	1.5	versicolor
## 86	6.0	3.4	4.5	1.6	versicolor
## 87	6.7	3.1	4.7	1.5	versicolor
## 88	6.3	2.3	4.4	1.3	versicolor
## 89	5.6	3.0	4.1	1.3	versicolor
## 90	5.5	2.5	4.0	1.3	versicolor
## 91	5.5	2.6	4.4	1.2	versicolor
## 92	6.1	3.0	4.6	1.4	versicolor
## 93	5.8	2.6	4.0	1.2	versicolor
## 94	5.0	2.3	3.3	1.0	versicolor
## 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

virginica_subset

##	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
## 101	6.3	3.3	6.0	2.5	virginica
## 102	5.8	2.7	5.1	1.9	virginica
## 103	7.1	3.0	5.9	2.1	virginica
## 104	6.3	2.9	5.6	1.8	virginica
## 105	6.5	3.0	5.8	2.2	virginica
## 106	7.6	3.0	6.6	2.1	virginica
## 107	4.9	2.5	4.5	1.7	virginica
## 108	7.3	2.9	6.3	1.8	virginica
## 109	6.7	2.5	5.8	1.8	virginica
## 110	7.2	3.6	6.1	2.5	virginica
## 111	6.5	3.2	5.1	2.0	virginica
## 112	6.4	2.7	5.3	1.9	virginica
## 113	6.8	3.0	5.5	2.1	virginica
## 114	5.7	2.5	5.0	2.0	virginica
## 115	5.8	2.8	5.1	2.4	virginica
## 116	6.4	3.2	5.3	2.3	virginica
## 117	6.5	3.0	5.5	1.8	virginica
## 118	7.7	3.8	6.7	2.2	virginica
## 119	7.7	2.6	6.9	2.3	virginica
## 120	6.0	2.2	5.0	1.5	virginica
## 121	6.9	3.2	5.7	2.3	virginica
## 122	5.6	2.8	4.9	2.0	virginica
## 123	7.7	2.8	6.7	2.0	virginica
## 124	6.3	2.7	4.9	1.8	virginica
## 125	6.7	3.3	5.7	2.1	virginica
## 126	7.2	3.2	6.0	1.8	virginica
## 127	6.2	2.8	4.8	1.8	virginica
## 128	6.1	3.0	4.9	1.8	virginica
## 129	6.4	2.8	5.6	2.1	virginica
## 130	7.2	3.0	5.8	1.6	virginica
## 131	7.4	2.8	6.1	1.9	virginica
## 132	7.9	3.8	6.4	2.0	virginica
## 133	6.4	2.8	5.6	2.2	virginica
## 134	6.3	2.8	5.1	1.5	virginica
## 135	6.1	2.6	5.6	1.4	virginica
## 136	7.7	3.0	6.1	2.3	virginica
## 137	6.3	3.4	5.6	2.4	virginica
## 138	6.4	3.1	5.5	1.8	virginica
## 139	6.0	3.0	4.8	1.8	virginica
## 140	6.9	3.1	5.4	2.1	virginica
## 141	6.7	3.1	5.6	2.4	virginica
## 142	6.9	3.1	5.1	2.3	virginica
## 143	5.8	2.7	5.1	1.9	virginica
## 144	6.8	3.2	5.9	2.3	virginica
## 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
```

```
# Display the last six rows of each species.
tail(setosa_subset, 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_subset, 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_subset, 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
```

6e. 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.

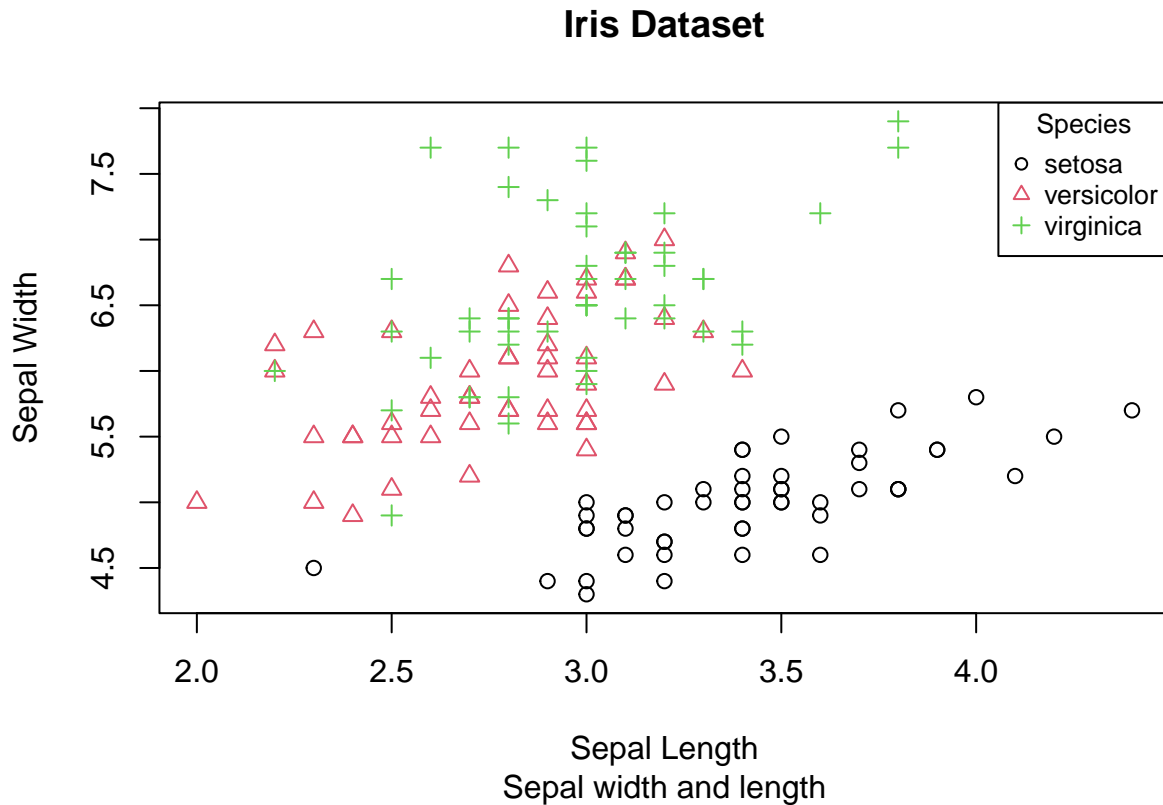
```
# Convert the "Species" column to a factor
iris$Species <- as.factor(iris$Species)
```

```
# Create a scatterplot
```

```
plot(
  Sepal.Length ~ Sepal.Width,
  data = iris,
  pch = as.integer(iris$Species), # Use different pch symbols for each species
  col = as.integer(iris$Species), # Use different colors for each species
  xlab = "Sepal Length",
  ylab = "Sepal Width",
  main = "Iris Dataset",
  sub = "Sepal width and length"
)
```

```
# Add a legend
```

```
legend("topright", legend = levels(iris$Species), col = 1:3, pch = 1:3, cex = 0.8, title = "Species")
```



6f. Interpret the result.

```
# The dataset consists of five variables (columns) and 150 observations (rows) in a data frame format.
# Petal.Length, Petal.Width, Sepal.Length, and Sepal.Width are the names of the four numerical variables.
# The factor variable Species, which represents the species of iris flowers, is the sixth variable. The
```

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_file <- read_excel("alexa_file.xlsx")
alexa_file

## # A tibble: 3,150 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 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_file$variation <- gsub("Black Dot", "BlackDot", alexa_file$variation)
alexa_file$variation <- gsub("Black Plus", "BlackPlus", alexa_file$variation)
alexa_file$variation <- gsub("Black Show", "BlackShow", alexa_file$variation)
alexa_file$variation <- gsub("Black Spot", "BlackSpot", alexa_file$variation)

alexa_file$variation <- gsub("White Dot", "WhiteDot", alexa_file$variation)
alexa_file$variation <- gsub("White Plus", "WhitePlus", alexa_file$variation)
alexa_file$variation <- gsub("White Show", "WhiteShow", alexa_file$variation)
alexa_file$variation <- gsub("White Spot", "WhiteSpot", alexa_file$variation)

alexa_file

## # A tibble: 3,150 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 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

```

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?

```

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

var_total <- alexa_file %>%
  count(alexa_file$variation)

var_total

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

```
save(var_total, file = "var_total.RData")
```

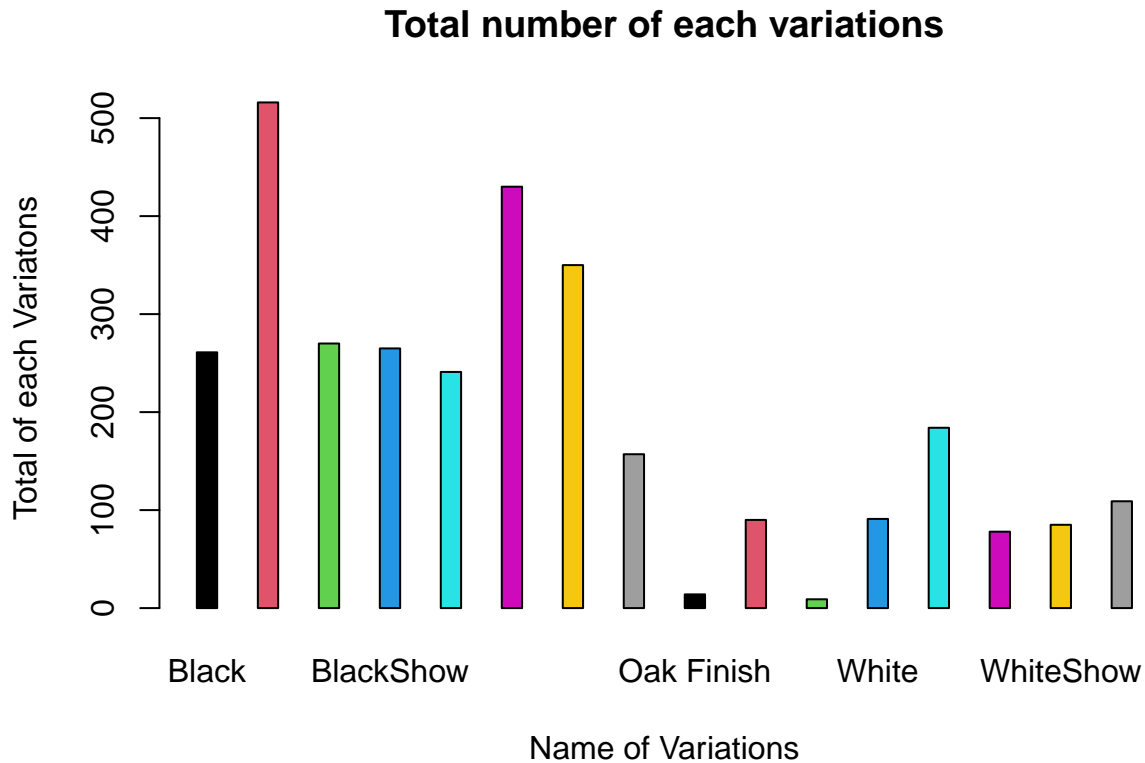
7c. From the variations.RData, create a barplot(). Complete the details of the chart which include the title, color, labels of each bar.

```
load("var_total.RData")
var_total
```

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

```
varNames <- var_total$`alexa_file$variation`
```

```
totalPlot <- barplot(var_total$n,
  names.arg = varNames,
  main = "Total number of each variations",
  xlab = "Name of Variations",
  ylab = "Total of each Variatons",
  col = 1:16,
  space = 2)
```



7d. Create a `barplot()` for the black and white variations. Plot it in 1 frame, side by side. Complete the

```

```r
blackVars <- var_total[var_total$`alexa_file$variation` %in% c("Black", "BlackPlus" , "BlackShow" ,"BlackDot", "BlackCross", "BlackStar", "BlackTriangle"),]

whiteVars <- var_total[var_total$`alexa_file$variation` %in% c("White", "WhiteDot", "WhitePlus", "WhiteStar", "WhiteTriangle"),]

par(mfrow = c(1,2))

barplot(height = blackVars$n,
 names.arg = blackVars$`alexa_file$variation`,
 col = c("black"),
 main = "Black Variations",
 xlab = "Variation",
 ylab = "Count",
 border = "black")

barplot(height = whiteVars$n,
 names.arg = whiteVars$`alexa_file$variation`,
 col = c("black"),
 main = "White Variations",
 xlab = "Variation",
 ylab = "Count",
 border = "black")

```

