RWORKSHEET_GENTAPAO#3A

BEA JULIETTE L. GENTAPAO

2025-10-13

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

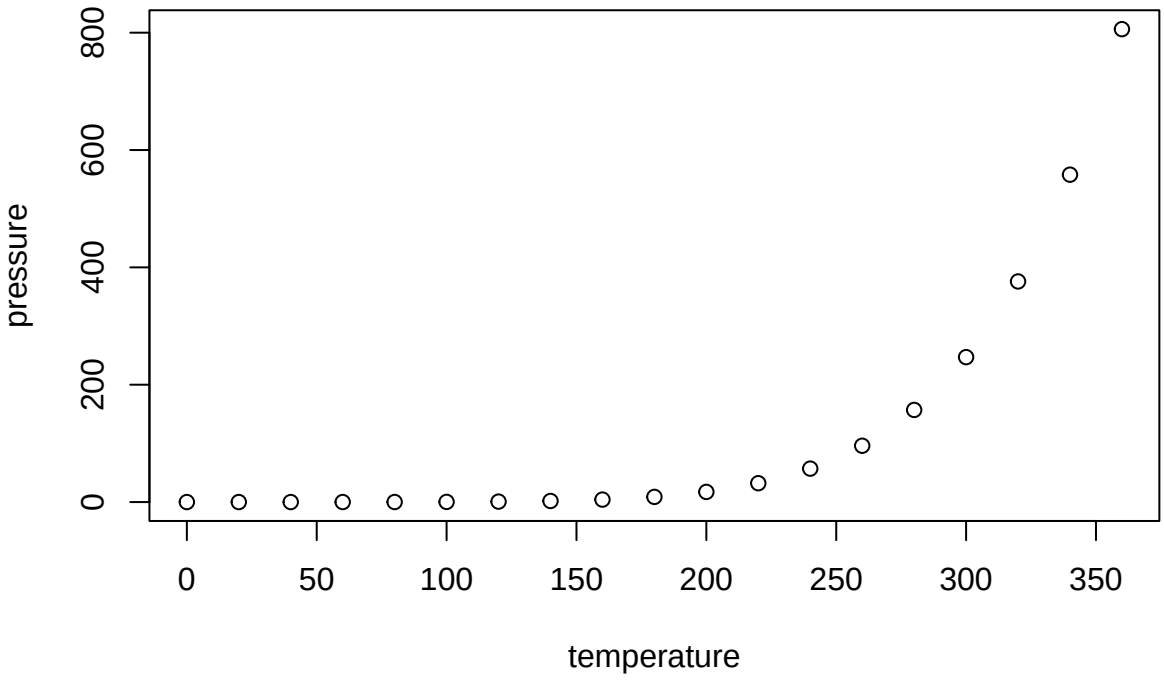
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
summary(cars)
```

```
##      speed           dist
##  Min.   : 4.0   Min.   :  2.00
##  1st Qu.:12.0   1st Qu.: 26.00
##  Median :15.0   Median : 36.00
##  Mean   :15.4   Mean   : 42.98
##  3rd Qu.:19.0   3rd Qu.: 56.00
##  Max.   :25.0   Max.   :120.00
```

Including Plots

You can also embed plots, for example:



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.

Using Vectors

1. Working with LETTERS and letters vectors

a. Produce a vector that contains the first 11 letters

```
first_11 <- LETTERS[1:11]
first_11
```

```
##  [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K"
```

b. Produce a vector that contains the odd numbered letters

```
odd_letters <- LETTERS[seq(1, 26, by = 2)]
odd_letters
```

```
##  [1] "A" "C" "E" "G" "I" "K" "M" "O" "Q" "S" "U" "W" "Y"
```

c. Produce a vector that contains the vowels

```
vowels <- LETTERS[c(1, 5, 9, 15, 21)]
vowels
```

```
## [1] "A" "E" "I" "O" "U"
```

d. Produce a vector that contains the last 5 lowercase letters

```
last_5 <- letters[22:26]
last_5
```

```
## [1] "v" "w" "x" "y" "z"
```

e. Produce a vector that contains letters between 15 to 24 letters in lowercase

```
letters_15_24 <- letters[15:24]
letters_15_24
```

```
##  [1] "o" "p" "q" "r" "s" "t" "u" "v" "w" "x"
```

2. Average Temperatures in Philippine Cities

a. Create a character vector for the cities

```
city <- c("Tuguegarao City", "Manila", "Iloilo City",
          "Tacloban", "Samal Island", "Davao City")
city
```

```
## [1] "Tuguegarao City" "Manila"          "Iloilo City"     "Tacloban"
## [5] "Samal Island"    "Davao City"
```

b. Create a vector for temperatures

```
temp <- c(42, 39, 34, 34, 30, 27)
temp
```

```
## [1] 42 39 34 34 30 27
```

c. Create a dataframe to combine city and temp

```
city_temp_df <- data.frame(city, temp)
city_temp_df
```

```
##              city temp
## 1 Tuguegarao City   42
## 2          Manila   39
## 3     Iloilo City   34
## 4        Tacloban   34
## 5    Samal Island   30
## 6      Davao City   27
```

d. Name the columns using names() function

```
names(city_temp_df) <- c("City", "Temperature")
city_temp_df
```

```
##              City Temperature
## 1 Tuguegarao City          42
## 2          Manila          39
## 3     Iloilo City          34
## 4        Tacloban          34
## 5    Samal Island          30
## 6      Davao City          27
```

e. Print the structure using str() function

```
str(city_temp_df)
```

```
## 'data.frame':    6 obs. of  2 variables:
##  $ City       : chr  "Tuguegarao City" "Manila" "Iloilo City" "Tacloban" ...
##  $ Temperature: num  42 39 34 34 30 27
```

Description of output: The `str()` function displays the structure of the dataframe. It shows that `city_temp_df` is a data frame with 6 observations (rows) and 2 variables (columns). The `City` column is a character type with 6 entries, and the `Temperature` column is numeric with 6 values. This gives us a compact view of the data types and dimensions.

f. Display the content of row 3 and row 4

```
city_temp_df[3:4, ]
```

```
##          City Temperature
## 3 Iloilo City          34
## 4    Tacloban          34
```

g. Display the city with highest and lowest temperature

```
# City with highest temperature
highest_temp <- city_temp_df[which.max(city_temp_df$Temperature), ]
cat("City with highest temperature:\n")
```

```
## City with highest temperature:
```

```
print(highest_temp)
```

```
##              City Temperature
## 1 Tuguegarao City          42
```

```
# City with lowest temperature
lowest_temp <- city_temp_df[which.min(city_temp_df$Temperature), ]
cat("\nCity with lowest temperature:\n")
```

```
##
## City with lowest temperature:
```

```
print(lowest_temp)
```

```
##         City Temperature
## 6 Davao City          27
```

Using Matrices

2. Create a matrix with specific values

a. Create matrix of 1 to 8 and 11 to 14 with 4 columns and 3 rows

```
matrix_data <- matrix(c(1:8, 11:14), nrow = 3, ncol = 4)
matrix_data
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    4    7   12
## [2,]    2    5    8   13
## [3,]    3    6   11   14
```

b. Multiply the matrix by two

```
multiplied_matrix <- matrix_data * 2
multiplied_matrix
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    8   14   24
## [2,]    4   10   16   26
## [3,]    6   12   22   28
```

c. Display the content of row 2

```
matrix_data[2, ]
```

```
## [1]  2  5  8 13
```

d. Display columns 3 and 4 in rows 1 and 2

```
matrix_data[1:2, 3:4]
```

```
##      [,1] [,2]
## [1,]    7   12
## [2,]    8   13
```

e. Display only columns 2 and 3, row 3

```
matrix_data[3, 2:3]
```

```
## [1]  6 11
```

f. Display only column 4

```
matrix_data[, 4]
```

```
## [1] 12 13 14
```

g. Name the rows and columns

```
dimnames(multiplied_matrix) <- list(
  c("isa", "dalawa", "tatlo"),
  c("uno", "dos", "tres", "quatro")
```

```
)
multiplied_matrix
```

```
##        uno dos tres quatro
## isa      2   8   14     24
## dalawa   4  10   16     26
## tatlo    6  12   22     28
```

h. Reshape the matrix to 2 columns and 6 rows

```
dim(matrix_data) <- c(6, 2)
matrix_data
```

```
##      [,1] [,2]
## [1,]    1    7
## [2,]    2    8
## [3,]    3   11
## [4,]    4   12
## [5,]    5   13
## [6,]    6   14
```

Using Arrays

3. Create an array with specific values

a. Create a three-dimensional array

```
array_values <- c(1, 2, 3, 6, 7, 8, 9, 0, 3, 4, 5, 1)
repeated_values <- rep(array_values, each = 2)
three_dim_array <- array(repeated_values, dim = c(2, 4, 3))
three_dim_array
```

```
## , , 1
##
##      [,1] [,2] [,3] [,4]
## [1,]    1    2    3    6
## [2,]    1    2    3    6
##
## , , 2
##
##      [,1] [,2] [,3] [,4]
## [1,]    7    8    9    0
## [2,]    7    8    9    0
##
## , , 3
##
##      [,1] [,2] [,3] [,4]
## [1,]    3    4    5    1
## [2,]    3    4    5    1
```

b. How many dimensions does your array have?

```
num_dimensions <- length(dim(three_dim_array))
cat("Number of dimensions:", num_dimensions)
```

```
## Number of dimensions: 3
```

The array has **3 dimensions**.

c. Name the rows, columns, and dimensions

```
dimnames(three_dim_array) <- list(
  c("a", "b"),                                    # row names (lowercase)
  c("A", "B", "C", "D"),                          # column names (uppercase)
  c("1st-Dimensional Array",
    "2nd-Dimensional Array",
    "3rd-Dimensional Array")                      # dimension names
)
three_dim_array
```

```
## , , 1st-Dimensional Array
##
##   A B C D
## a 1 2 3 6
## b 1 2 3 6
##
```

```
## , , 2nd-Dimensional Array
##
##   A B C D
## a 7 8 9 0
## b 7 8 9 0
##
## , , 3rd-Dimensional Array
##
##   A B C D
## a 3 4 5 1
## b 3 4 5 1
```