RWORKSHEET_GENTAPAO#4A

BEA JULIETTE L. GENTAPAO

2025-12-22

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

Data Frames and Analysis

1. Shoe Size and Height Data

a. Create a data frame and describe the data

```
# Create the data frame
shoe_data <- data.frame(
  Shoe_Size = c(6.5, 9.0, 8.5, 8.5, 10.5, 7.0, 9.5, 9.0, 13.0,
                7.5, 10.5, 8.5, 12.0, 10.5, 13.0, 11.5, 8.5, 5.0,
                10.0, 6.5, 7.5, 8.5, 10.5, 8.5, 10.5, 11.0, 9.0, 13.0),
  Height = c(66.0, 68.0, 64.5, 65.0, 70.0, 64.0, 70.0, 71.0, 72.0,
             64.0, 74.5, 67.0, 71.0, 71.0, 77.0, 72.0, 59.0, 62.0,
             72.0, 66.0, 64.0, 67.0, 73.0, 69.0, 72.0, 70.0, 69.0, 70.0),
  Gender = c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M",
             "F", "M", "M", "M", "M", "F", "F", "M", "F", "F", "M",
             "M", "F", "M", "M", "M", "M")
)

cat("Data Description:\n")
```

```
## Data Description:
```

```
str(shoe_data)
```

```
## 'data.frame':    28 obs. of  3 variables:
##  $ Shoe_Size: num  6.5 9 8.5 8.5 10.5 7 9.5 9 13 7.5 ...
##  $ Height   : num  66 68 64.5 65 70 64 70 71 72 64 ...
##  $ Gender   : chr  "F" "F" "F" "F" ...
```

```
summary(shoe_data)
```

```
##    Shoe_Size          Height         Gender
##  Min.   : 5.000   Min.   :59.00   Length:28
##  1st Qu.: 8.500   1st Qu.:65.75   Class :character
##  Median : 9.000   Median :69.50   Mode  :character
##  Mean   : 9.411   Mean   :68.57
##  3rd Qu.:10.500   3rd Qu.:71.25
##  Max.   :13.000   Max.   :77.00
```

Description: The data frame contains 28 observations with 3 variables: Shoe_Size (numeric), Height (numeric), and Gender (character). The data represents shoe sizes ranging from 5.0 to 13.0, heights from 59.0 to 77.0 inches, and includes both male and female respondents.

b. Create subset by males and females

```
# Subset for males
males_data <- shoe_data[shoe_data$Gender == "M", c("Shoe_Size", "Height")]
cat("Males - Shoe Size and Height:\n")
```

```
## Males - Shoe Size and Height:
```

```
print(males_data)
```

```
##    Shoe_Size Height
## 5       10.5   70.0
## 9       13.0   72.0
## 11      10.5   74.5
## 13      12.0   71.0
## 14      10.5   71.0
## 15      13.0   77.0
## 16      11.5   72.0
## 19      10.0   72.0
## 22       8.5   67.0
## 23      10.5   73.0
## 25      10.5   72.0
## 26      11.0   70.0
## 27       9.0   69.0
## 28      13.0   70.0
```

```
# Subset for females
females_data <- shoe_data[shoe_data$Gender == "F", c("Shoe_Size", "Height")]
cat("\nFemales - Shoe Size and Height:\n")
```

```
##
## Females - Shoe Size and Height:
```

```
print(females_data)
```

```
##    Shoe_Size Height
## 1        6.5   66.0
## 2        9.0   68.0
## 3        8.5   64.5
```

```
## 4        8.5   65.0
## 6        7.0   64.0
## 7        9.5   70.0
## 8        9.0   71.0
## 10       7.5   64.0
## 12       8.5   67.0
## 17       8.5   59.0
## 18       5.0   62.0
## 20       6.5   66.0
## 21       7.5   64.0
## 24       8.5   69.0
```

c. Find the mean of shoe size and height

```
mean_shoe_size <- mean(shoe_data$Shoe_Size)
mean_height <- mean(shoe_data$Height)

cat("Mean Shoe Size:", mean_shoe_size, "\n")
```

```
## Mean Shoe Size: 9.410714
```

```
cat("Mean Height:", mean_height, "inches\n")
```

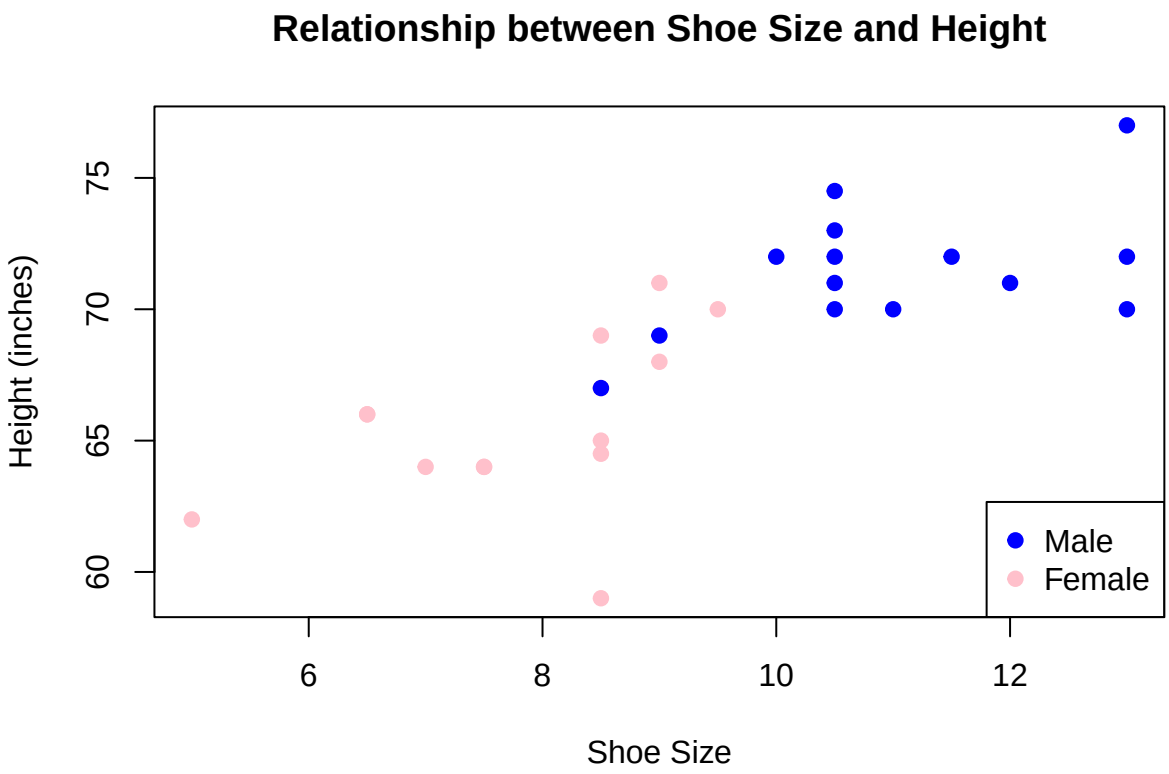
```
## Mean Height: 68.57143 inches
```

d. Relationship between shoe size and height

```
# Calculate correlation
correlation <- cor(shoe_data$Shoe_Size, shoe_data$Height)
cat("Correlation coefficient:", correlation, "\n")
```

```
## Correlation coefficient: 0.7766089
```

```
# Create a simple scatter plot representation
plot(shoe_data$Shoe_Size, shoe_data$Height,
     main = "Relationship between Shoe Size and Height",
     xlab = "Shoe Size", ylab = "Height (inches)",
     pch = 19, col = ifelse(shoe_data$Gender == "M", "blue", "pink"))
legend("bottomright", legend = c("Male", "Female"),
       col = c("blue", "pink"), pch = 19)
```



Answer: Yes, there is a positive relationship between shoe size and height. The correlation coefficient of 0.777 indicates a strong positive correlation. This means that generally, as shoe size increases, height tends to increase as well. This relationship is biologically logical since taller individuals typically have larger feet to support their body frame.

Working with Factors

2. Create factor for months

```
months_vector <- c("March", "April", "January", "November", "January",
                   "September", "October", "September", "November", "August",
                   "January", "November", "November", "February", "May", "August",
                   "July", "December", "August", "August", "September", "November",
                   "February", "April")

factor_months_vector <- factor(months_vector)
print(factor_months_vector)
```

```
##  [1] March     April     January   November  January   September October
##  [8] September November  August    January   November  November  February
## [15] May       August    July      December  August    August    September
## [22] November  February  April
## 11 Levels: April August December February January July March May ... September
```

3. Compare summary of months_vector and factor_months_vector

```
cat("Summary of months_vector (character):\n")
```

```
## Summary of months_vector (character):
```

```
summary(months_vector)
```

```
##    Length     Class      Mode
##        24 character character
```

```
cat("\n\nSummary of factor_months_vector (factor):\n")
```

```
##
##
## Summary of factor_months_vector (factor):
```

```
summary(factor_months_vector)
```

```
##     April    August  December  February   January      July     March       May
##         2         4         1         2         3         1         1         1
##  November   October September
##         5         1         3
```

Interpretation:

- The `months_vector` (character vector) summary only shows the length, class, and mode, which is not very informative for categorical data.
- The `factor_months_vector` summary provides a frequency count of each month, showing how many times each month appears in the data.
- The factor version is much more useful in this case because it gives us meaningful insights into the distribution of months, making it easy to see that November appears most frequently (4 times), followed by August, January, and September (3 times each).

4. Create vector and factor for directions

```
direction <- c("East", "West", "North")
frequency <- c(1, 4, 3)
```

```r
# Create factor with specified order
factor_data <- factor(direction, levels = c("East", "West", "North"))
print(factor_data)
```

```
## [1] East  West  North
## Levels: East West North
```

```r
# Display with frequency
direction_table <- data.frame(Direction = factor_data, Frequency = frequency)
print(direction_table)
```

```
##   Direction Frequency
## 1      East         1
## 2      West         4
## 3     North         3
```

```r
# Apply new order
new_order_data <- factor(factor_data, levels = c("East", "West", "North"))
print(new_order_data)
```

```
## [1] East  West  North
## Levels: East West North
```

5. Import Excel data

Create and save the CSV file first

```
# Create the data
import_data <- data.frame(
  Student_Number = c(1, 2, 3, 4, 5, 6, 7, 8, 9, 10),
  Pre_Test = c(55, 54, 47, 57, 51, 61, 57, 54, 63, 58),
  Post_Test = c(61, 60, 56, 63, 56, 63, 59, 62, 69, 63)
)

# Save as CSV
write.csv(import_data, file = "import_march.csv", row.names = FALSE)
cat("CSV file created successfully!\n")
```

```
## CSV file created successfully!
```

a. Import the excel file using read.table()

```
imported_data <- read.table("import_march.csv", header = TRUE, sep = ",")
cat("Data imported successfully!\n")
```

```
## Data imported successfully!
```

b. View the dataset

```
print(imported_data)
```

```
##    Student_Number Pre_Test Post_Test
## 1               1       55        61
## 2               2       54        60
## 3               3       47        56
## 4               4       57        63
## 5               5       51        56
## 6               6       61        63
## 7               7       57        59
## 8               8       54        62
## 9               9       63        69
## 10             10       58        63
```

```
str(imported_data)
```

```
## 'data.frame':    10 obs. of  3 variables:
##  $ Student_Number: int  1 2 3 4 5 6 7 8 9 10
##  $ Pre_Test      : int  55 54 47 57 51 61 57 54 63 58
##  $ Post_Test     : int  61 60 56 63 56 63 59 62 69 63
```

Using Conditional Statements (IF-ELSE)

6. Full Search - Random Number Selection

```
# Function to check selected number
check_number <- function(selected_num) {
  if (selected_num < 1 || selected_num > 50) {
    return("The number selected is beyond the range of 1 to 50")
  } else if (selected_num == 20) {
    return("TRUE")
  } else {
    return(selected_num)
  }
}

# Test with different numbers
cat("Testing with number 20:\n")
```

```
## Testing with number 20:
```

```
print(check_number(20))
```

```
## [1] "TRUE"
```

```
cat("\nTesting with number 35:\n")
```

```
##
## Testing with number 35:
```

```
print(check_number(35))
```

```
## [1] 35
```

```
cat("\nTesting with number 60:\n")
```

```
##
## Testing with number 60:
```

```
print(check_number(60))
```

```
## [1] "The number selected is beyond the range of 1 to 50"
```

```
cat("\nTesting with number 0:\n")
```

```
##
## Testing with number 0:
```

```
print(check_number(0))
```

```
## [1] "The number selected is beyond the range of 1 to 50"
```

7. Minimum Bills Calculator

```
# Function to calculate minimum number of bills
minimum_bills <- function(price) {
  bills <- c(1000, 500, 200, 100, 50)
  bill_count <- 0
  remaining <- price
```

```
  cat("Price of snack:", price, "pesos\n")
  cat("Bills breakdown:\n")

  for (bill in bills) {
    if (remaining >= bill) {
      num_bills <- remaining %/% bill
      bill_count <- bill_count + num_bills
      cat(bill, "peso bill(s):", num_bills, "\n")
      remaining <- remaining %% bill
    }
  }

  cat("\nMinimum number of bills needed:", bill_count, "\n")
  return(bill_count)
}

# Test examples
cat("Example 1:\n")
```

```
## Example 1:
```

```
minimum_bills(350)
```

```
## Price of snack: 350 pesos
## Bills breakdown:
## 200 peso bill(s): 1
## 100 peso bill(s): 1
## 50 peso bill(s): 1
##
## Minimum number of bills needed: 3
```

```
## [1] 3
```

```
cat("\n\nExample 2:\n")
```

```
##
##
## Example 2:
```

```
minimum_bills(1750)
```

```
## Price of snack: 1750 pesos
## Bills breakdown:
## 1000 peso bill(s): 1
## 500 peso bill(s): 1
## 200 peso bill(s): 1
## 50 peso bill(s): 1
##
## Minimum number of bills needed: 4
```

```
## [1] 4
```

```
cat("\n\nExample 3:\n")
```

```
##
##
## Example 3:
```

```r
minimum_bills(650)
```

```
## Price of snack: 650 pesos
## Bills breakdown:
## 500 peso bill(s): 1
## 100 peso bill(s): 1
## 50 peso bill(s): 1
##
## Minimum number of bills needed: 3
```

```
## [1] 3
```

8. Student Math Scores Analysis

a. Create dataframe from the table

```
students_df <- data.frame(
  Name = c("Annie", "Thea", "Steve", "Hanna"),
  Grade1 = c(85, 65, 75, 95),
  Grade2 = c(65, 75, 55, 75),
  Grade3 = c(85, 90, 80, 100),
  Grade4 = c(100, 90, 85, 90)
)

print(students_df)
```

```
##    Name Grade1 Grade2 Grade3 Grade4
## 1 Annie     85     65     85    100
## 2  Thea     65     75     90     90
## 3 Steve     75     55     80     85
## 4 Hanna     95     75    100     90
```

b. Output average score of students with average > 90

```
# Calculate average for each student without rowMeans
for (i in 1:nrow(students_df)) {
  total <- students_df[i, 2] + students_df[i, 3] +
           students_df[i, 4] + students_df[i, 5]
  average <- total / 4

  if (average > 90) {
    cat(students_df$Name[i], "'s average grade this semester is",
        average, "\n")
  }
}
```

c. Output tests with average score < 80

```
# Calculate average for each test without mean function
for (test in 2:5) {
  total <- 0
  for (i in 1:nrow(students_df)) {
    total <- total + students_df[i, test]
  }
  average <- total / nrow(students_df)

  if (average < 80) {
    test_number <- test - 1
    cat("The", test_number, "test was difficult.\n")
  }
}
```

```
## The 2 test was difficult.
```

d. Output students with highest grade > 90

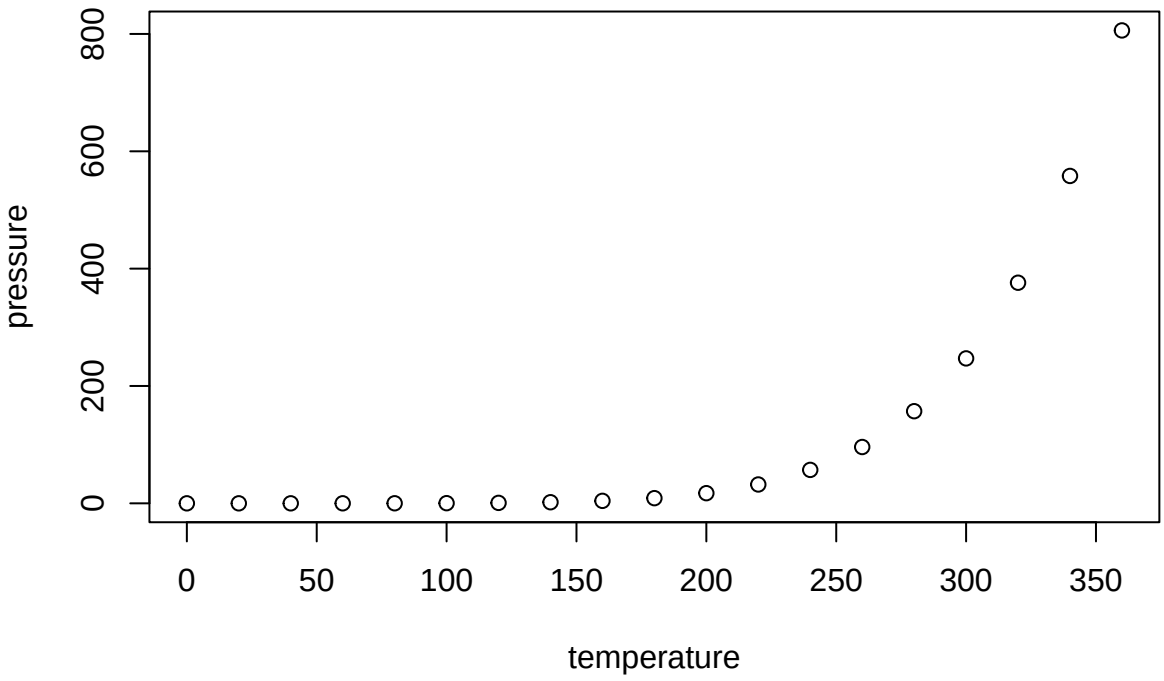
```
# Find highest grade for each student without max function
for (i in 1:nrow(students_df)) {
  highest <- students_df[i, 2]

  # Check each grade
  if (students_df[i, 3] > highest) {
    highest <- students_df[i, 3]
  }
  if (students_df[i, 4] > highest) {
    highest <- students_df[i, 4]
  }
  if (students_df[i, 5] > highest) {
    highest <- students_df[i, 5]
  }

  if (highest > 90) {
    cat(students_df$Name[i], "'s highest grade this semester is",
        highest, "\n")
  }
}
```

```
## Annie 's highest grade this semester is 100
## Hanna 's highest grade this semester is 100
```

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.