RWorksheet_#4a

Tamonan

2024-10-14

```
# 1.a Create a data frame.
Data_Frame <- 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
Height = c(66.0, 68.0, 64.5, 65.0, 70.0, 64.0, 70.0, 71.0, 72.0, 64.0, 74.75, 67.0, 71.0, 71.0, 77.0, 71
Gender = c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M", "F", "M", "M", "M", "M", "F", "F", "M
)
Data_Frame
```

```
##    Shoe_Size Height Gender
## 1        6.5  66.00      F
## 2        9.0  68.00      F
## 3        8.5  64.50      F
## 4        8.5  65.00      F
## 5       10.5  70.00      M
## 6        7.0  64.00      F
## 7        9.5  70.00      F
## 8        9.0  71.00      F
## 9       13.0  72.00      M
## 10       7.5  64.00      F
## 11      10.5  74.75      M
## 12       8.5  67.00      F
## 13      12.0  71.00      M
## 14      10.5  71.00      M
## 15      13.0  77.00      M
## 16      11.5  72.00      M
## 17       8.5  59.00      F
## 18       5.0  62.00      F
## 19      10.0  72.00      M
## 20       6.5  66.00      F
## 21       7.5  64.00      F
## 22       8.5  67.00      M
## 23      10.5  73.00      M
## 24       8.5  69.00      F
## 25      10.5  72.00      M
## 26      11.0  70.00      M
## 27       9.0  69.00      M
## 28      13.0  70.00      M
```

```
# 1.b b.    Create a subset by males and females with their corresponding shoe size and height.
# Subset for Females
female_subset <- subset(Data_Frame, Gender == "F", select = c(Shoe_Size, Height))
female_subset
```

```
##    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
```

```r
# Subset for Males
male_subset <- subset(Data_Frame, Gender == "M", select = c(Shoe_Size, Height))
male_subset
```

```
##    Shoe_Size Height
## 5       10.5  70.00
## 9       13.0  72.00
## 11      10.5  74.75
## 13      12.0  71.00
## 14      10.5  71.00
## 15      13.0  77.00
## 16      11.5  72.00
## 19      10.0  72.00
## 22       8.5  67.00
## 23      10.5  73.00
## 25      10.5  72.00
## 26      11.0  70.00
## 27       9.0  69.00
## 28      13.0  70.00
```

```r
# 1.c Find the mean of shoe size and height of the respondents.
# Mean of Shoe Size
mean_shoe_size <- mean(Data_Frame$Shoe_Size)
mean_shoe_size
```

```
## [1] 9.410714
```

```r
# Mean of Height
mean_height <- mean(Data_Frame$Height)
mean_height
```

```
## [1] 68.58036
```

```r
# 1.d Is there a relationship between shoe size and height? Why?
# NO...
```

```r
# 2. Construct character vector months to a factor with factor() and assign the result to factor_months_
# Create the character vector for months
months_vector <- c("March", "April", "January", "November", "January", "September", "October", "Septembe
# Convert months_vector to a factor
factor_months_vector <- factor(months_vector)
# Print the factor version
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
```

```r
# Print levels of the factor
levels(factor_months_vector)
```

```
##  [1] "April"     "August"    "December"  "February"  "January"   "July"
##  [7] "March"     "May"       "November"  "October"   "September"
```

```r
#3. Then check the summary() of the months_vector and factor_months_vector. | Inter- pret the results o
# Get summary of the original character vector
summary(months_vector)
```

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

```r
# Get summary of the factor vector
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
```

```r
# 4.    Create a vector and factor for the table below.
# Create the character vector for directions
directions_vector <- c("East", "West", "North", "West", "West", "West", "North", "North")

# Convert it to a factor with a specified order of levels
factor_directions_vector <- factor(directions_vector, levels = c("East", "West", "North"))

# Print the factor vector with the specified order of levels
print(factor_directions_vector)
```

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

```r
# 5. Enter the data below in Excel with file name = import_march.csv
read.table(file = "import_march.csv", header=TRUE, sep=",")
```

```
##   Students Strategy.1 Strategy.2 Strategy.3  X
## 1     Male          8         10          8 NA
## 2                   4          8          6 NA
## 3                   0          6          4 NA
## 4   Female         14          4         15 NA
## 5                  10          2         12 NA
## 6                   6          0          9 NA
## 7                  NA         NA         NA NA
```

```r
# 6. Prompt the user for a number
user_input <- readline(prompt = "Enter a number from 1 to 50: ")
```

```
## Enter a number from 1 to 50:
```

```r
# Convert the input to a numeric value
number <- as.numeric(user_input)

# Check conditions and display the appropriate message
if (!is.na(number)) {
  if (number == 20) {
    cat("TRUE\n")
  } else if (number >= 1 && number <= 50) {
    cat("You entered:", number, "\n")
  } else {
    cat("The number selected is beyond the range of 1 to 50\n")
  }
} else {
  cat("Invalid input. Please enter a number.\n")
}
```

```
## Invalid input. Please enter a number.
```

```r
# 7. Function to calculate the minimum number of bills
min_bills <- function(price) {
  bills <- c(1000, 500, 200, 100, 50)
  count <- 0

# Calculate minimum bills
  for (bill in bills) {
    count <- count + (price %/% bill)
    price <- price %% bill
  }

  cat("Minimum number of bills needed:", count, "\n")
}

price <- as.integer(readline(prompt = "Enter snack price (must be divisible by 50): "))
```

```
## Enter snack price (must be divisible by 50):
```

```r
min_bills(price)
```

```
## Minimum number of bills needed: NA
```

```r
# 8.a Create a dataframe for students' scores
scores <- 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(scores)
```

```
##    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
```

```r
# 8.b Calculate averages and display students with averages over 90
for (i in 1:nrow(scores)) {
  student <- scores[i, ]
  avg_score <- sum(student[2:5]) / 4

  if (avg_score > 90) {
    cat(student$Name, "'s average grade this semester is", avg_score, "\n")
  }
}
```

```r
# 8.c Identify tests with average score below 80
for (j in 2:5) {  # Iterate over Grade columns
  test_avg <- sum(scores[[j]]) / nrow(scores)

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

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

```r
# 8.d Output highest score for each student without using `max`
for (i in 1:nrow(scores)) {
  student <- scores[i, ]
  highest_score <- sort(as.numeric(student[2:5]), decreasing = TRUE)[1]

  if (highest_score > 90) {
    cat(student$Name, "'s highest grade this semester is", highest_score, "\n")
  }
}
```

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