RWorksheet_Laurente#4A

Karl

2024-10-14

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.

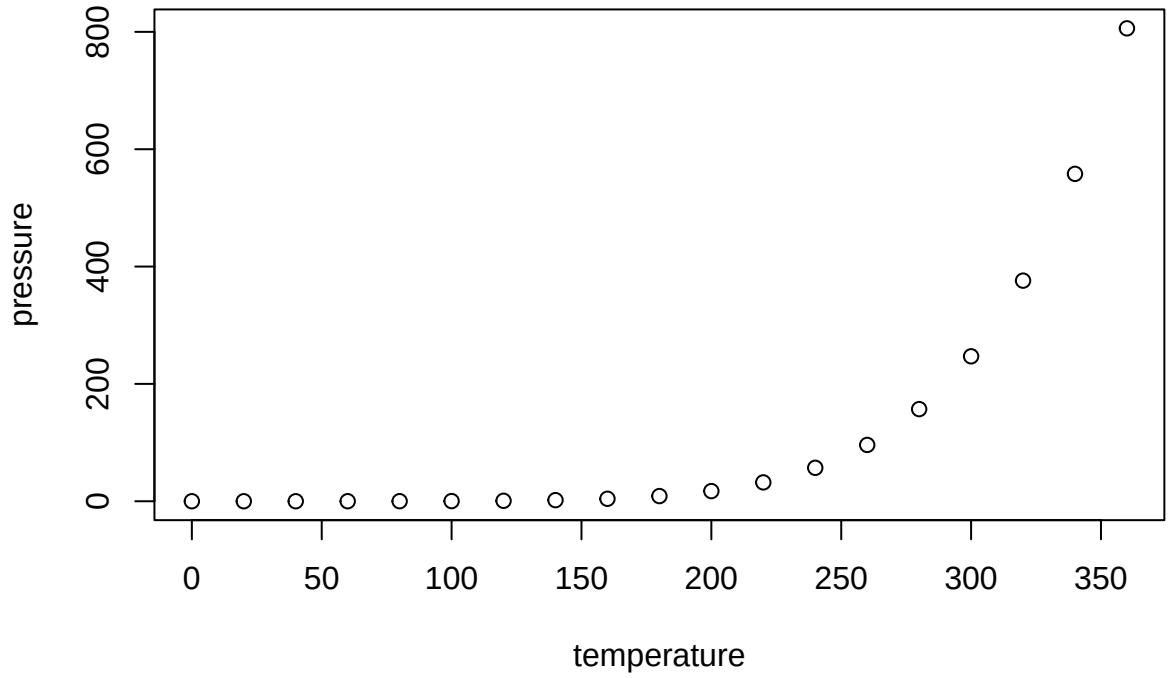
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
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.

#CS 101 #Karl Andrei G. Laurente #BSIT 2-B

```
#1.)
#a.)
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, 
Gender <- c("F", "F", "F", "F", "M", "F", "F", "F", "M", "F", "M", "F", "M", "M", "M", "M", "F", "F", "
length(Shoe_Size)
```

```
## [1] 28
```

```
length(Height)
```

```
## [1] 28
```

```
length(Gender)
```

```
## [1] 28
```

```r
a_one <- data.frame(Shoe_Size, Height, Gender)
a_one
```

```
##    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
```

```r
#The data shows the shoe sizes, height, and gender of each individual customer. The shoe sizes are in U
```

```r
#1.)
#b.)
male_only <- a_one[a_one$Gender == "M", ]
male_only
```

```
##    Shoe_Size Height Gender
## 5       10.5  70.00      M
## 9       13.0  72.00      M
## 11      10.5  74.75      M
## 13      12.0  71.00      M
## 14      10.5  71.00      M
## 15      13.0  77.00      M
## 16      11.5  72.00      M
## 19      10.0  72.00      M
## 22       8.5  67.00      M
## 23      10.5  73.00      M
## 25      10.5  72.00      M
```

```
## 26      11.0  70.00      M
## 27       9.0  69.00      M
## 28      13.0  70.00      M
```

```
female_only <- a_one[a_one$Gender == "F", ]
female_only
```

```
##    Shoe_Size Height Gender
## 1        6.5   66.0      F
## 2        9.0   68.0      F
## 3        8.5   64.5      F
## 4        8.5   65.0      F
## 6        7.0   64.0      F
## 7        9.5   70.0      F
## 8        9.0   71.0      F
## 10       7.5   64.0      F
## 12       8.5   67.0      F
## 17       8.5   59.0      F
## 18       5.0   62.0      F
## 20       6.5   66.0      F
## 21       7.5   64.0      F
## 24       8.5   69.0      F
```

```
#1.)
#c.)
mean(a_one$Shoe_Size)
```

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

```
mean(a_one$Height)
```

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

```
#1.)
#d.)
#The shorter their height is, the smaller their shoe size gets and vice versa. That's because shoe size
```

```
#2.)
months <- c("March","April","January","November","January", "September","October","September","November"
months
```

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

```
factor_months <- factor(months, levels = c("January", "February", "March", "April", "May", "June", "July
factor_months
```

```
##  [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
## 12 Levels: January February March April May June July August ... December
```

```
sort(factor_months)
```

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

```
#3.)
summary(months)
```

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

```
summary(factor_months)
```

```
##   January  February     March     April       May      June      July    August
##         3         2         1         2         1         0         1         4
## September   October  November  December
##         3         1         5         1
```

```
#The summary on months display the length of the vector which has 24 elements, also presents the data t
#The summary on factor_months only displays how much a string of characters repeat itself inside the ve
```

```
#4.)
#a.)
four_a <- data.frame(
          Direction = c("East", "West", "North"),
          Frequency = c(1, 4, 3)
)
four_a
```

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

```
new_order_data <- factor(four_a, levels = c("East", "West", "North"))
new_order_data
```

```
## Direction Frequency
##      <NA>      <NA>
## Levels: East West North
```

```r
#5.)
#a.)
read.table(file = "import_march.csv", header = TRUE, sep = ",")
```

```
##   Column1 Strategy.1 Strategy.2 Strategy.3
## 1    Male          8         10          8
## 2                  4          8          6
## 3                  0          6          4
## 4  Female         14          4         15
## 5                 10          2         12
## 6                  6          0          9
```

```r
#5.)
#b.)
read.table(file = "import_march.csv", header = TRUE, sep = ",")
```

```
##   Column1 Strategy.1 Strategy.2 Strategy.3
## 1    Male          8         10          8
## 2                  4          8          6
## 3                  0          6          4
## 4  Female         14          4         15
## 5                 10          2         12
## 6                  6          0          9
```

```r
#6.)
#a.)
six_eyy <- as.integer(readline(prompt ="Enter a value between 1 to 50: "))
```

```
## Enter a value between 1 to 50:
```

```r
six_eyy <- 20
six_eyyy <- if (six_eyy == 20){
  print("TRUE")
} else if (six_eyy < 50 & six_eyy > 0){
  print(six_eyy)
} else {
  print("The number selected is beyond the range of 1 to 50")
}
```

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

```r
six_eyyy
```

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

```r
#7.)
#a.)
price <- as.integer(readline(prompt = "Enter price of snack: "))
```

```
## Enter price of snack:
```

```r
price <- 1500
if (price %% 1000 == 0){
  seven_one <- price/1000
  cat("Minimum number of bills is", seven_one, "of 1000")
} else if (price %% 500 == 0){
  seven_two <- price/500
  cat("Minimum number of bills is", seven_two, "of 500")
} else if (price %% 200 == 0){
  seven_three <- price/200
  cat("Minimum number of bills is", seven_three, "of 200")
} else if (price %% 100 == 0){
  seven_four <- price/100
  cat("Minimum number of bills is", seven_four, "of 100")
} else if (price %% 50 == 0){
  seven_five <- price/50
  cat("Minimum number of bills is", seven_five, "of 50")
} else {
  print("Only input numbers divisible by 50")
}
```

```
## Minimum number of bills is 3 of 500
```

```r
#8.)
#a.)
eight_a <- 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)
)
eight_a
```

```
##    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.)
dim(eight_a)
```

```
## [1] 4 5
```

```r
Annie <- sum(eight_a[1, c(2,3,4,5)])
Annie_mean <- Annie/4
Annie_mean
```

```
## [1] 83.75
```

```r
#8.)
#c.)
Grade_one <- sum(eight_a[, 2])
Grade_two <- sum(eight_a[, 3])
Grade_three <- sum(eight_a[, 4])
Grade_four <- sum(eight_a[, 5])

Grade_one_mean <- Grade_one/4
Grade_two_mean <- Grade_two/4
Grade_three_mean <- Grade_three/4
Grade_four_mean <- Grade_four/4

Grade_one_mean
```

```
## [1] 80
```

```r
Grade_two_mean
```

```
## [1] 67.5
```

```r
Grade_three_mean
```

```
## [1] 88.75
```

```r
Grade_four_mean
```

```
## [1] 91.25
```

```r
if (Grade_one_mean < 80){
  print("The Grade 1 Test was difficult")
}

if (Grade_two_mean < 80){
  print("The Grade 2 Test was difficult")
}
```

```
## [1] "The Grade 2 Test was difficult"
```

```r
if (Grade_three_mean < 80){
  print("The Grade 3 Test was difficult")
}
if (Grade_four_mean < 80){
  print("The Grade 4 Test was difficult")
}
```

```r
#8.)
#d.)
Annie_grades <- data.frame(
              Grade_1 = 85,
              Grade_2 = 65,
```

```
              Grade_3 = 85,
              Grade_4 = 100
)
Thea_grades <- data.frame(
              Grade_1 = 65,
              Grade_2 = 75,
              Grade_3 = 90,
              Grade_4 = 90
)
Steve_grades <- data.frame(
              Grade_1 = 75,
              Grade_2 = 55,
              Grade_3 = 80,
              Grade_4 = 85
)
Hanna_grades <- data.frame(
              Grade_1 = 95,
              Grade_2 = 75,
              Grade_3 = 100,
              Grade_4 = 90
)
Annie_final <- Annie_grades[Annie_grades > 90]
Annie_highest <- sort(Annie_final, decreasing = TRUE)[1]
Annie_highest
```

```
## [1] 100
```

```
Thea_final <- Thea_grades[Thea_grades > 90]
Thea_highest <- sort(Thea_final, decreasing = TRUE)[1]
Thea_highest
```

```
## [1] NA
```

```
Steve_final <- Steve_grades[Steve_grades > 90]
Steve_highest <- sort(Steve_final, decreasing = TRUE)[1]
Steve_highest
```

```
## [1] NA
```

```
Hanna_final <- Hanna_grades[Hanna_grades > 90]
Hanna_highest <- sort(Hanna_final, decreasing = TRUE)[1]
Hanna_highest
```

```
## [1] 100
```