

RWORKSHEET_GENTAPAO#1

BEA JULIETTE L. GENTAPAO

2025-09-01

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>.

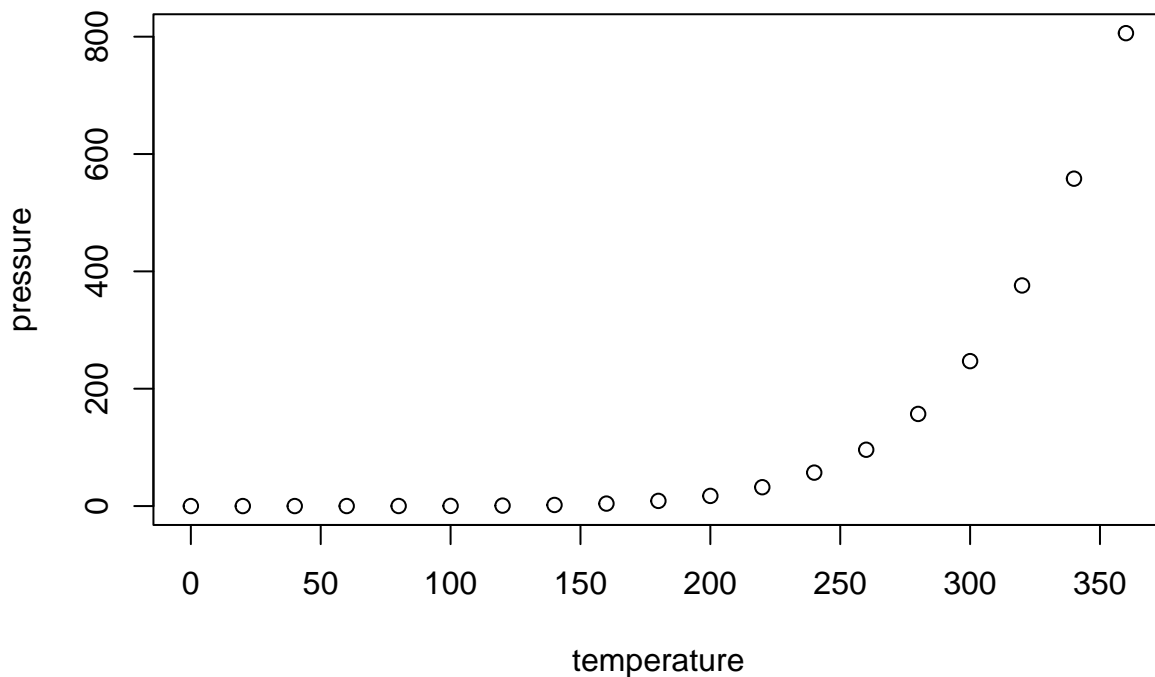
When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
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.

1. Set up a vector named age

```
age <- c(34, 28, 22, 36, 27, 18, 52, 39, 42, 29, 35, 31, 27, 22, 37, 34, 19, 20, 57, 49, 50, 37, 46, 25, 17, 37, 42, 53, 41, 51, 35, 24, 33, 41)
```

1a. How many data points?

```
length(age) # Output: 34 data points
```

1b. R code and output

```
age # Output: 34 28 22 36 27 18 52 39 42 29 35 31 27 22 37 34 19 20 57 49 50 37 46 25 17 37 42 53 41 51 35 24 33 41
```

2. Find the reciprocal of the values for age

```
reciprocal_age <- 1/age reciprocal_age # Output: 0.02941176 0.03571429 0.04545455 0.02777778 0.03703704 0.05555556...
```

3. Assign new_age <- c(age, 0, age)

```
new_age <- c(age, 0, age) new_age # What happens: The new_age vector contains the original age vector, followed by 0, # followed by the age vector again. Total length is 34 + 1 + 34 = 69 elements.
```

4. Sort the values for age

```
sort(age) # Output: 17 18 19 20 22 22 24 25 27 27 28 29 31 33 34 34 35 35 36 37 37 37 39 41 41 42 42 46 49 50 51 52 53 57
```

5. Find the minimum and maximum value for age

```
min(age) # Output: 17
```

```
max(age) # Output: 57
```

6. Set up a vector named data

```
data <- c(2.4, 2.8, 2.1, 2.5, 2.4, 2.2, 2.5, 2.3, 2.5, 2.3, 2.4, 2.7)
```

6a. How many data points?

```
length(data) # Output: 12 data points
```

6b. R code and output

```
data # Output: 2.4 2.8 2.1 2.5 2.4 2.2 2.5 2.3 2.5 2.3 2.4 2.7
```

7. Double every value of the data

```
doubled_data <- data * 2 doubled_data # Output: 4.8 5.6 4.2 5.0 4.8 4.4 5.0 4.6 5.0 4.6 4.8 5.4 # What happens: Each value in the data vector is multiplied by 2
```

8. Generate sequences

8.1 Integers from 1 to 100

```
seq1 <- seq(1, 100) seq1 # Output: 1 2 3 4 5 ... 98 99 100
```

8.2 Numbers from 20 to 60

```
seq2 <- seq(20, 60) seq2 # Output: 20 21 22 23 24 ... 58 59 60
```

8.3 Mean of numbers from 20 to 60

```
mean(seq(20, 60)) # Output: 40
```

8.4 Sum of numbers from 51 to 91

```
sum(seq(51, 91)) # Output: 2911
```

8.5 Integers from 1 to 1,000

```
seq5 <- seq(1, 1000)
```

8a. How many data points from 8.1 to 8.4?

```
length(seq1) # 100 length(seq2) # 41 length(seq(20, 60)) # 41 (for mean calculation) length(seq(51, 91)) # 41 (for sum calculation)
```

8c. For 8.5 find only maximum data points until 10

```
seq5[1:10] # Output: 1 2 3 4 5 6 7 8 9 10
```

9. Print integers between 1 and 100 not divisible by 3, 5, and 7

```
filtered <- Filter(function(i) { all(i %% c(3,5,7) != 0) }, seq(100)) filtered # Output: 1 2 4 8 11 13 16 17 19 22 23 26 29 31 32 34 37 38 41 43 44 46 47 52 53 58 59 61 62 64 67 68 71 73 74 76 79 82 83 86 88 89 92 94 97
```

10. Generate a sequence backwards from 100 to 1

```
backward_seq <- seq(100, 1) backward_seq # Output: 100 99 98 97 96 ... 4 3 2 1
```

11. Natural numbers below 25 that are multiples of 3 or 5

```
multiples <- Filter(function(i) { i %% 3 == 0 | i %% 5 == 0 }, seq(1, 24))  
multiples # Output: 3 5 6 9 10  
12 15 18 20 21 24  
sum(multiples) # Output: 168
```

11a. How many data points from 10 to 11?

```
length(backward_seq) # 100 data points  
length(multiples) # 11 data points
```

12. Statement grouping with braces

The code `{x <- 0 + x + 5 +}` produces an error

Description: The error occurs because the `+` operator at the end expects another value.

The expression is incomplete. When `{` is entered, R waits for more input (shown by `+`).

Even with `}`, the syntax is incorrect because the expression ends with an operator.

Correct way:

```
x <- 0  
x <- x + 5  
x # Output: 5
```

13. Set up a vector named score

```
score <- c(72, 86, 92, 63, 88, 89, 91, 92, 75, 75, 77)
```

Find `x[2]` and `x[3]`

```
score[2] # Output: 86
```

```
score[3] # Output: 92
```

14. Create a vector with NA values

```
a <- c(1, 2, NA, 4, NA, 6, 7)
```

14a. Change NA to 999

```
print(a, na.print="-999") # Output: 1 2 -999 4 -999 6 7
```

14b. Description: The `na.print` parameter displays NA values as “-999” in the output,

but it doesn’t actually change the values in the vector. The NA values remain NA

in the actual data structure; only the display is modified.

15. Create a vector and check class

```
x <- c(2, 3, 4) class(x) # Output: “numeric”
```

Change class to foo

```
class(x) <- “foo” class(x) # Output: “foo”
```

Additional Example

Note: `readline()` works interactively in console, not in scripts

Example output when run line by line:

Input your name: John

Input your age: 25

Output: “My name is John and I am 25 years old.”

R.version.string shows the R version being used

```
name = readline(prompt=“Input your name:”) age = readline(prompt=“Input your age:”) print(paste(“My name is”, name, “and I am”, age, “years old.”)) print(R.version.string)
```