STA 445 HW2

Paige Hawkinson

02-27-2024

Problem 1

Create a vector of three elements (2,4,6) and name that vector `vec_a`. Create a second vector, `vec_b`, that contains (8,10,12). Add these two vectors together and name the result `vec_c`.

```
vec_a <- c(2, 4, 6)
vec_b <- c(8, 10, 12)
vec_a + vec_b
```

```
## [1] 10 14 18
```

Problem 2

Create a vector, named `vec_d`, that contains only two elements (14,20). Add this vector to `vec_a`. What is the result and what do you think R did (look up the recycling rule using Google)? What is the warning message that R gives you?

```
vec_d <- c(14, 20)
vec_d + vec_a
```

```
## Warning in vec_d + vec_a: longer object length is not a multiple of shorter
## object length
```

```
## [1] 16 24 20
```

The result is a 1x3 vector. It looks like R added 14+2 and 20+4. 20 did not have anything added to it due to vec_d being a 1x2 vector. The warning message R gives me is 'longer object length is not a multiple of shorter object length.

Problem 3

Next add 5 to the vector vec_a. What is the result and what did R do? Why doesn't in give you a warning message similar to what you saw in the previous problem?

```
5 + vec_a
```

```
## [1]  7  9 11
```

The result is a 1x3 vector where 5 was added to each value in vec_a. Since 5 is a vector, R adds the scalar value to the vector quantities.

Problem 4

Generate the vector of integers $\{1, 2, \ldots 5\}$ in two different ways.

a. First using the `seq()` function

```
seq(from=1, to=5)
```

```
## [1] 1 2 3 4 5
```

b. Using the `a:b` shortcut.

```
1:5
```

```
## [1] 1 2 3 4 5
```

Problem 5

Generate the vector of even numbers $\{2, 4, 6, \ldots, 20\}$

a. Using the seq() function

```
seq(from=2, to=20, by=2)
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

b. Using the a:b shortcut and some subsequent algebra.

```
shortcut <- 1:10
2*shortcut
```

```
##  [1]  2  4  6  8 10 12 14 16 18 20
```

Problem 6

Generate a vector of 21 elements that are evenly placed between 0 and 1 using the `seq()` command and name this vector `x`.

```
x <- seq(from=0, to=1, length.out=21)
```

Problem 7

Generate the vector $\{2, 4, 8, 2, 4, 8, 2, 4, 8\}$ using the `rep()` command to replicate the vector c(2,4,8).

```
vec_e <- c(2, 4, 8)
rep(vec_e, 3)
```

```
## [1] 2 4 8 2 4 8 2 4 8
```

Problem 8

Generate the vector $\{2, 2, 2, 2, 4, 4, 4, 4, 8, 8, 8, 8\}$ using the `rep()` command. You might need to check the help file for rep() to see all of the options that rep() will accept. In particular, look at the optional argument `each=`.

```
rep(vec_e, each=4)
```

```
##  [1] 2 2 2 2 4 4 4 4 8 8 8 8
```

Problem 9

In this problem, we will work with the matrix

$$\begin{bmatrix} 2 & 4 & 6 & 8 & 10 \\ 12 & 14 & 16 & 18 & 20 \\ 22 & 24 & 26 & 28 & 30 \end{bmatrix}$$

a. Create the matrix in two ways and save the resulting matrix as `M`.
b. Create the matrix using some combination of the `seq()` and `matrix()` commands.

```
M <- matrix( c(2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30), nrow=3, byrow=TRUE)
M
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]    2    4    6    8   10
## [2,]   12   14   16   18   20
## [3,]   22   24   26   28   30
```

ii. Create the same matrix by some combination of multiple `seq()` commands and either the `rbind()` or `cbind()` command.

```
R1 <- seq(2, 10, by=2)
R2 <- seq(12, 20, by=2)
R3 <- seq(22, 30, by=2)
M.matrix <- rbind(R1, R2, R3)
M.matrix
```

```
##    [,1] [,2] [,3] [,4] [,5]
## R1    2    4    6    8   10
## R2   12   14   16   18   20
## R3   22   24   26   28   30
```

b. Extract the second row out of M.

```
M[2, ]
```

```
## [1] 12 14 16 18 20
```

c. Extract the element in the third row and second column of M

```
M[3, 2]
```

```
## [1] 24
```

Problem 10

The following code creates a `data.frame` and then has two different methods for removing the rows with `NA` values in the column `Grade`. Explain the difference between the two.

```
df <- data.frame(name= c('Alice','Bob','Charlie','Daniel'),
                 Grade = c(6,8,NA,9))

df[ -which(  is.na(df$Grade) ), ]

df[  which( !is.na(df$Grade) ), ]
```

In the first command after assigning our data frame to df, the is.na function is coding the values that are not NA as FALSE while the values that are NA are coded at TRUE. The negative sign in front of which is telling R to remove that NA value from the data set. In the second command, the values that are not NA are coded as TRUE while the value with NA is coded as FALSE. The exclamation mark in front of the is.na command is telling R to remove the NA value from the data set.

Problem 11

Create and manipulate a list.

a. Create a list named my.test with elements + x = c(4,5,6,7,8,9,10) + y = c(34,35,41,40,45,47,51) + slope = 2.82 + p.value = 0.000131

```
x <- c(4,5,6,7,8,9,10)
y <- c(34,35,41,40,45,47,51)
slope <- 2.82
p.value = 0.000131
my.test <- list(x=x, y=y, slope=slope, p.value=p.value)
```

b. Extract the second element in the list.

```
my.test[ 2 ]
```

```
## $y
## [1] 34 35 41 40 45 47 51
```

c. Extract the element named `p.value` from the list.

```
my.test[ 'p.value' ]
```

```
## $p.value
## [1] 0.000131
```