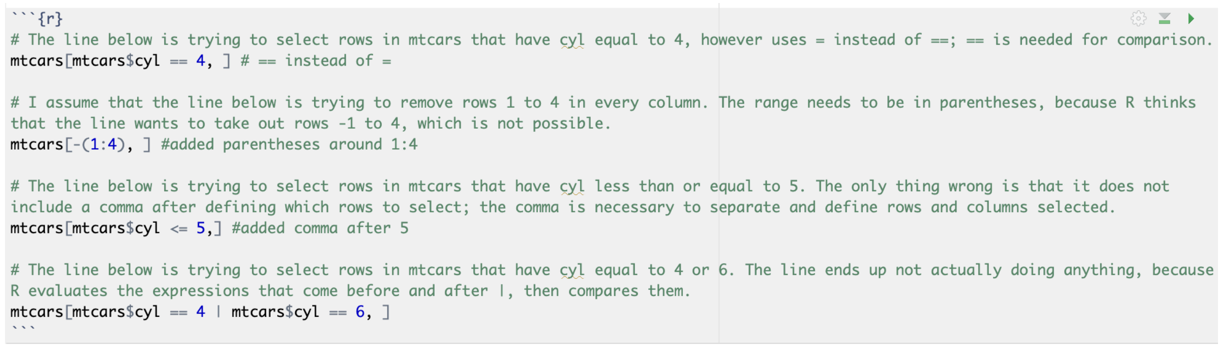
1) Subsetting vectors:
    a) x[i], x[c(i, j, z)]
        i) *negative values drop elements in corresponding position
    b) Ex:



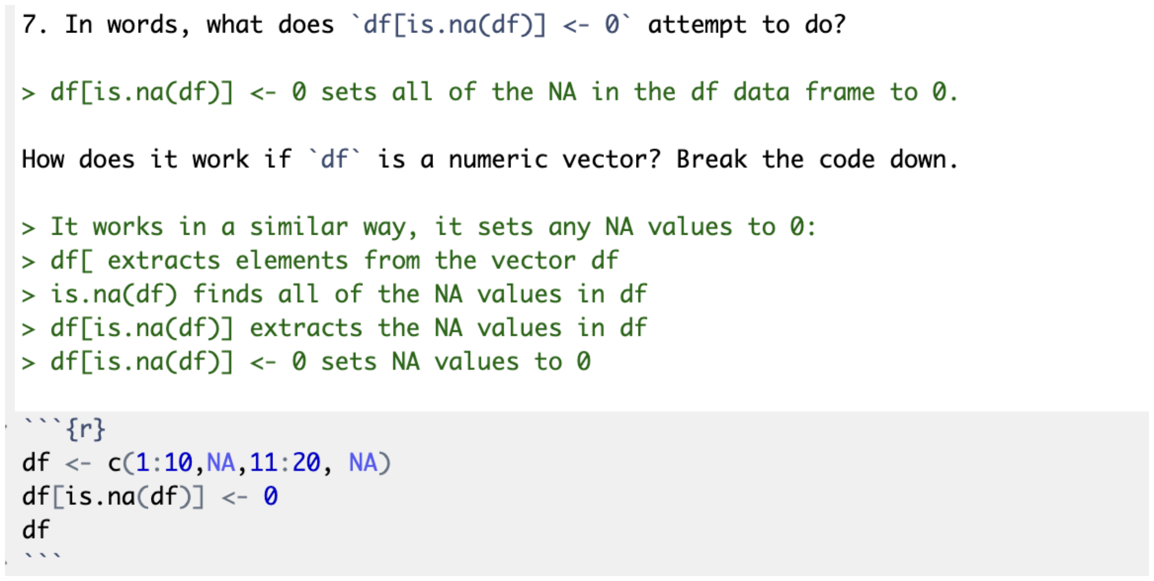
~~~~
```{r}
# The line below is trying to select rows in mtcars that have cyl equal to 4, however uses = instead of ==; == is needed for comparison.
mtcars[mtcars$cyl == 4, ] # == instead of =

# I assume that the line below is trying to remove rows 1 to 4 in every column. The range needs to be in parentheses, because R thinks
that the line wants to take out rows -1 to 4, which is not possible.
mtcars[-(1:4), ] #added parentheses around 1:4

# The line below is trying to select rows in mtcars that have cyl less than or equal to 5. The only thing wrong is that it does not
include a comma after defining which rows to select; the comma is necessary to separate and define rows and columns selected.
mtcars[mtcars$cyl <= 5,] #added comma after 5

# The line below is trying to select rows in mtcars that have cyl equal to 4 or 6. The line ends up not actually doing anything, because
R evaluates the expressions that come before and after |, then compares them.
mtcars[mtcars$cyl == 4 | mtcars$cyl == 6, ]
```
~~~~

2) Subsetting data frames:
    a) df[row, column], df[c(row1, row2), c(column1, column2)]
        i) *df[,column] selects all rows and a column
        ii) *df[row,] selects row and all columns
    b) Ex:



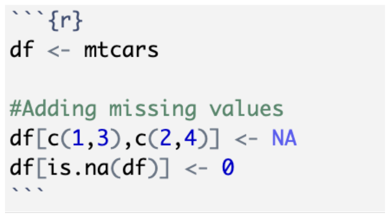
~~~~
7. In words, what does `df[is.na(df)] <- 0` attempt to do?

> df[is.na(df)] <- 0 sets all of the NA in the df data frame to 0.

How does it work if `df` is a numeric vector? Break the code down.

> It works in a similar way, it sets any NA values to 0:
> df[ extracts elements from the vector df
> is.na(df) finds all of the NA values in df
> df[is.na(df)] extracts the NA values in df
> df[is.na(df)] <- 0 sets NA values to 0

```{r}
df <- c(1:10,NA,11:20, NA)
df[is.na(df)] <- 0
df
```
~~~~



~~~~
```{r}
df <- mtcars

#Adding missing values
df[c(1,3),c(2,4)] <- NA
df[is.na(df)] <- 0
```
~~~~

3) Selecting single elements
    a) Data frames
        i) df[[postion (integer)]]

        ii) df[[“name”]] OR df$name
        iii) *data frames only need prefixes of names, tibbles need full names
    b) Lists
        i) Same for lists → [[and $ extract single pattern from a list

4) Map vs Walk
    a) Use map when you want the output, and walk when you don’t necessarily need to see the output