Note: I collaborated with Malu and Amy.

Question 1

- Provide at least two methods to view the number of rows and the number of columns in the `mpg` dataset.

```
library(ggplot2)

#using dim function
dim(mpg)
```

```
## [1] 234  11
```

```
#using nrow and ncol function
nrow(mpg)
```

```
## [1] 234
```

```
ncol(mpg)
```

```
## [1] 11
```

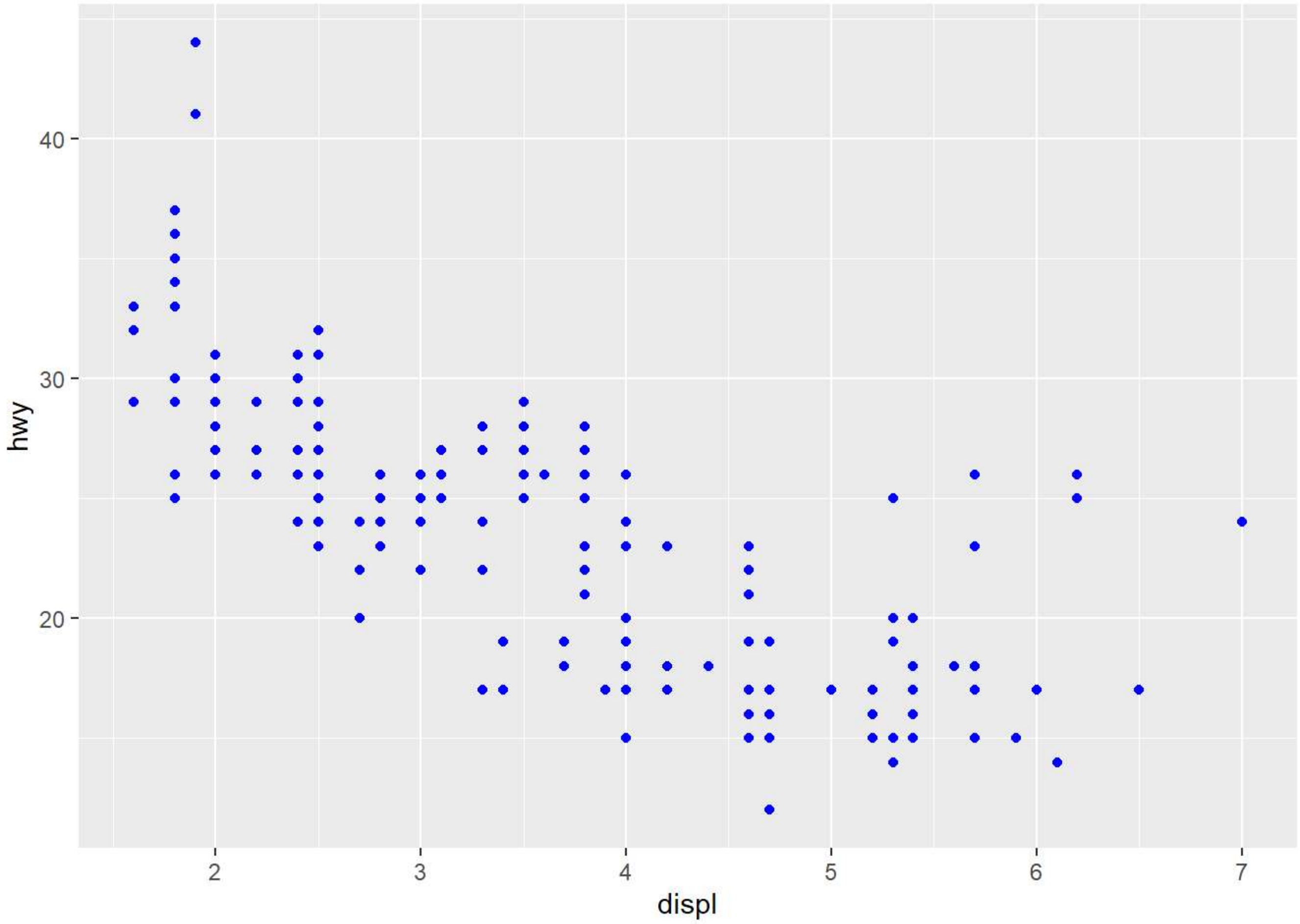
Question 2

- Execute the following code:

  ```
  ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ, y = hwy, color = "blue"))
  ```

- Identify the issue with the code.
- Correct it so that the plot displays as intended.

```
#identifying and correcting the issue in the code
ggplot(data = mpg) +
      geom_point(mapping = aes(x = displ, y = hwy), color = "blue")
```



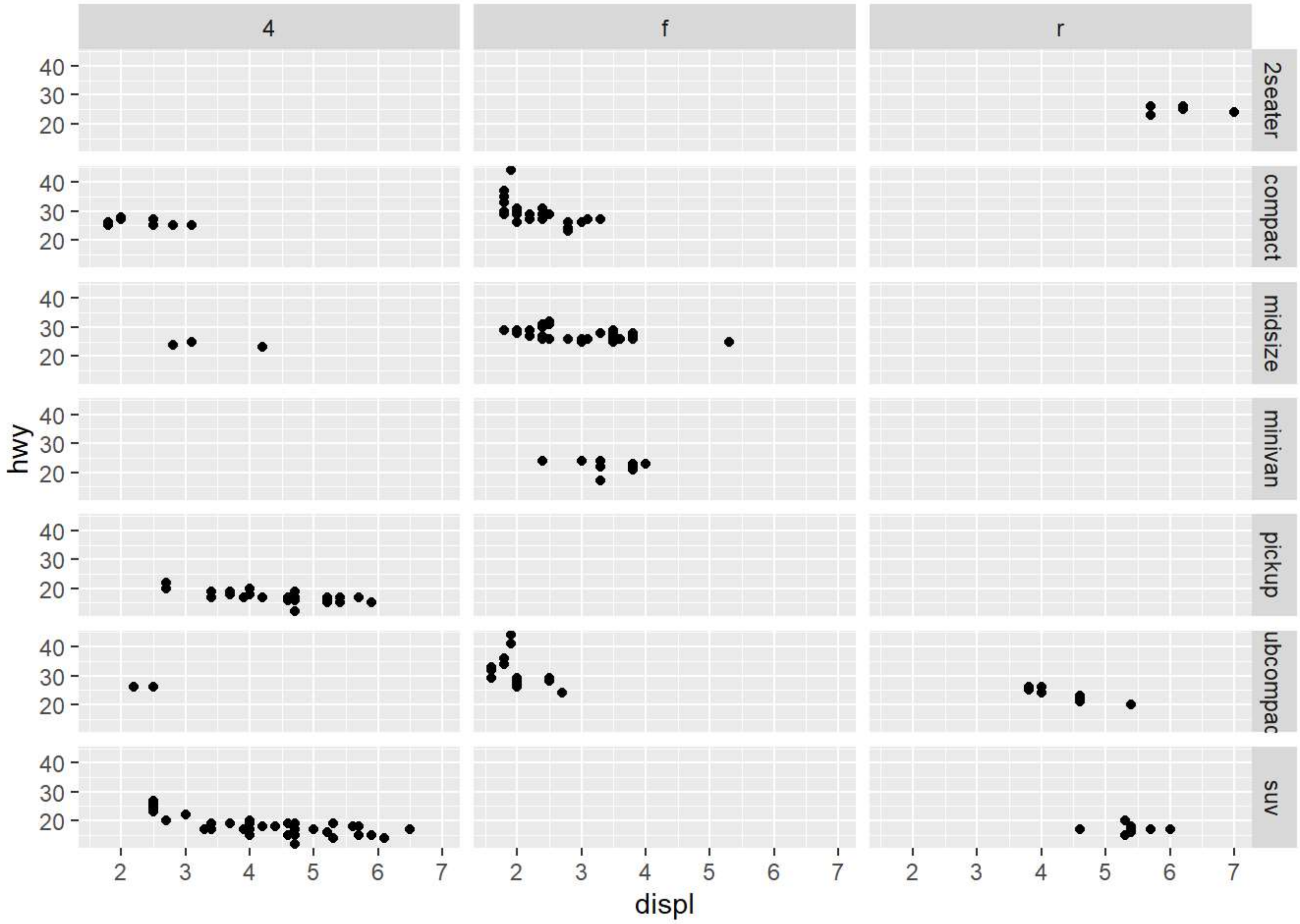
ISSUE: Color should have been outside of the “aes” argument in the “geom_point” function.

Question 3

Create a `facet grid` to explore how the relationships between `displ` and `hwy` differ among vehicle classes and drive configurations.

To create the facet grid, use the `facet_grid()` function in ggplot2. The tilde (~) symbol inside the function specifies which variable is a function of the other. For example, facet_grid(x ~ y) will generate subplots where x varies by row of the facet and y varies by column of the facet.

```
ggplot(data = mpg, aes(x = displ, y = hwy)) +
  geom_point() +
  facet_grid(class ~ drv)
```

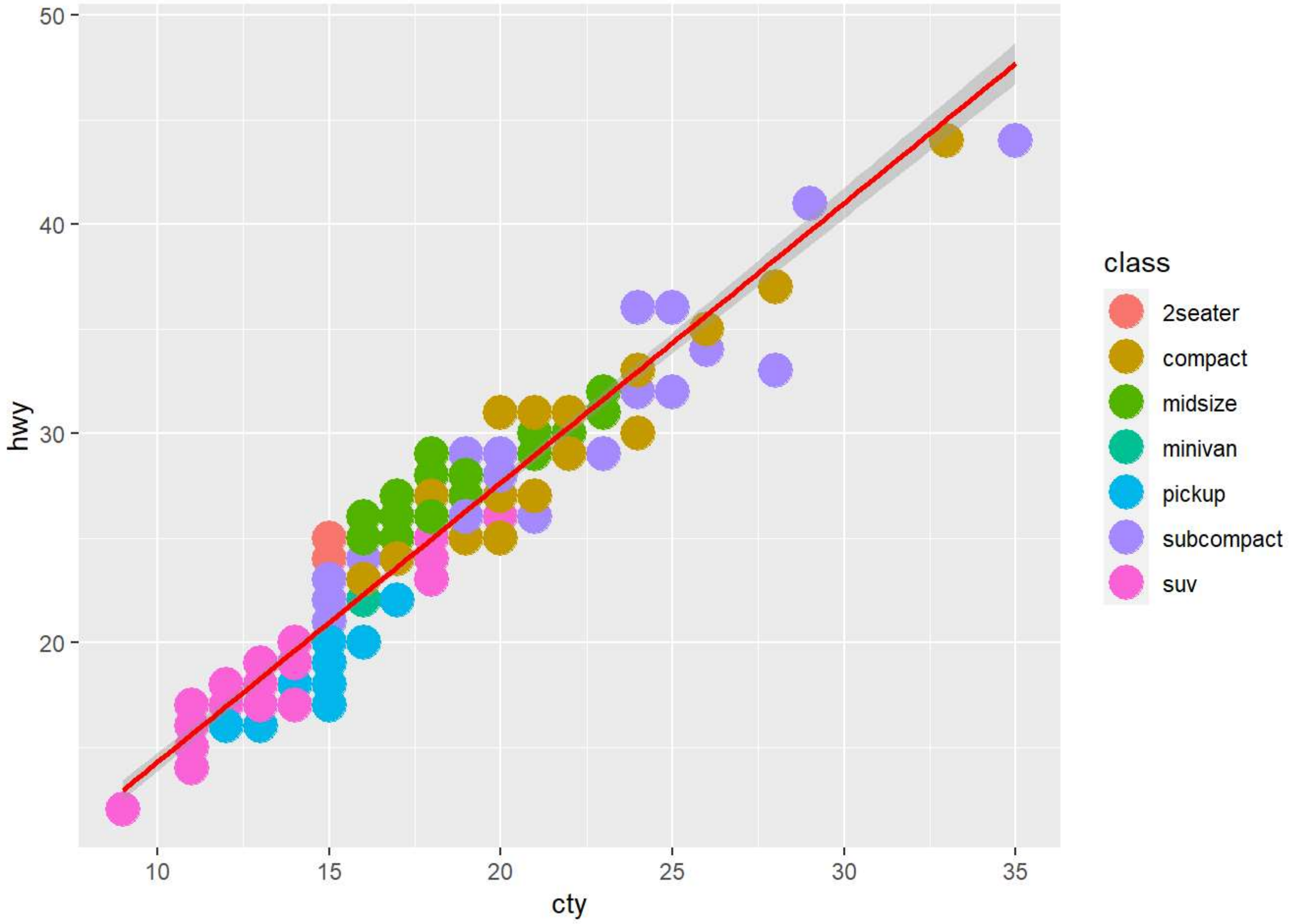


Question 4

- Create a scatter plot for the variables `cty` (x-axis) and `hwy` (y-axis) from the `mpg` dataset.
- Add a geom to show the linear regression between x and y.
    - Use a linear model (`lm`) instead of the default (`loess`) for computing the regression.
    - Set the line color to `red` .
    - Color the points by `class` .
    - Set the point size to 6.

```
ggplot(data = mpg, aes(x = cty, y = hwy)) +
    geom_point(aes(color = class), size = 6) +
    geom_smooth(method = "lm", se = TRUE, color = "red")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Question 5

- In the previous plot, you noticed subcompacts with a city fuel consumption of 15 miles per gallon.
- Extract only the manufacturer and model of these subcompacts.

```
compact <- subset(x = mpg, class == "compact")

man.mod <- compact[ ,c("manufacturer", "model")]

man.mod
```

```
## # A tibble: 47 × 2
##    manufacturer model
##    <chr>        <chr>
##  1 audi         a4
##  2 audi         a4
##  3 audi         a4
##  4 audi         a4
##  5 audi         a4
##  6 audi         a4
##  7 audi         a4
##  8 audi         a4 quattro
##  9 audi         a4 quattro
## 10 audi         a4 quattro
## # ℹ 37 more rows
```

Question 6

- The following code generates a warning:

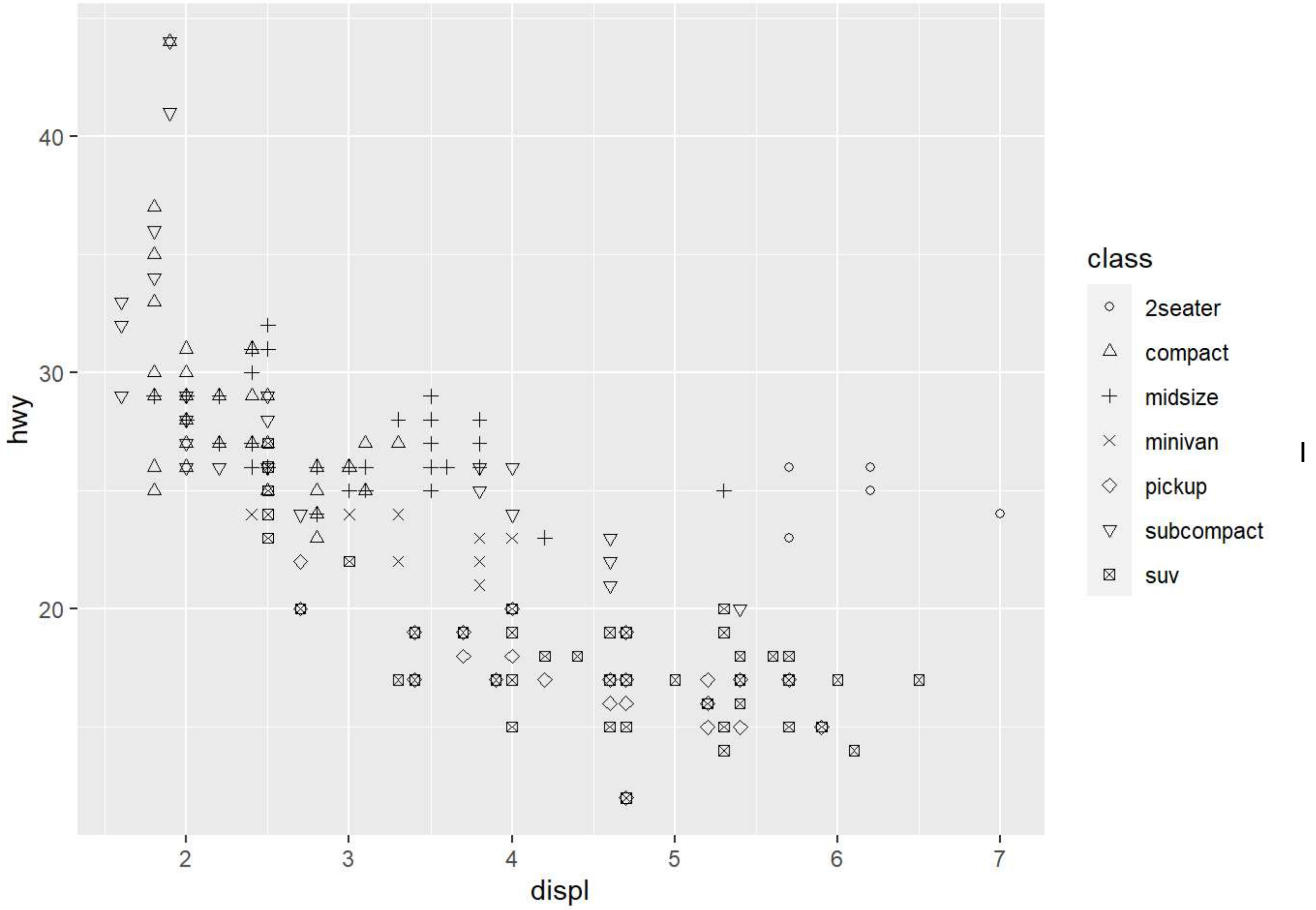
```
ggplot(data = mpg) +
  geom_point(mapping = aes(x = displ, y = hwy, shape = class))
```

- Correct the code so all classes are displayed.
  - Hint: You can manually specify shapes using the `scale_shape_manual` geom.

```
length(unique(mpg$class))
```

```
## [1] 7
```

```
ggplot(data = mpg) +
    geom_point(mapping = aes(x = displ, y = hwy, shape = class)) +
    scale_shape_manual(values = 1:7)
```



expanded the values to 7, the number of unique classes, so that all classes are displayed in the legend.

Question 7

- Given the following two vectors:

```
x <- c(3.385,0.48,1.35,465,36.33,27.66,14.83,1.04,4.19,0.425,0.101,0.92,1,0.005,0.06,3.5,2,1.7,2
547,0.023,187.1,521,0.785,10,3.3,0.2,1.41,529,207,85,0.75,62,6654,3.5,6.8,35,4.05,0.12,0.023,0.0
1,1.4,250,2.5,55.5,100,52.16,10.55,0.55,60,3.6,4.288,0.28,0.075,0.122,0.048,192,3,160,0.9,1.62,
0.104,4.235)
y <- c(44.5,15.5,8.1,423,119.5,115,98.2,5.5,58,6.4,4,5.7,6.6,0.14,1,10.8,12.3,6.3,4603,0.3,419,6
55,3.5,115,25.6,5,17.5,680,406,325,12.3,1320,5712,3.9,179,56,17,1,0.4,0.25,12.5,490,12.1,175,15
7,440,179.5,2.4,81,21,39.2,1.9,1.2,3,0.33,180,25,169,2.6,11.4,2.5,50.4)
```

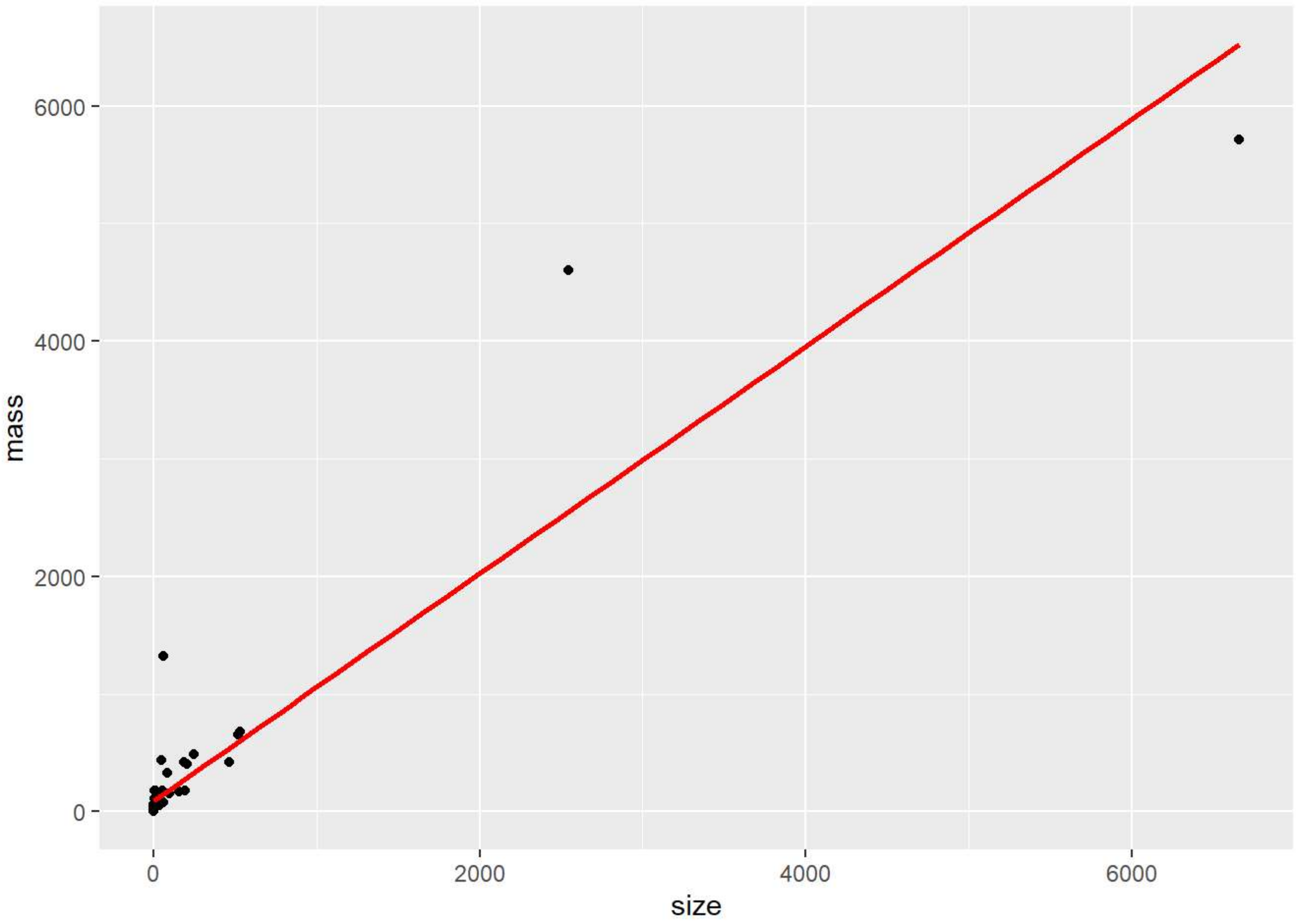
- Combine the two vectors into a data frame called `sizeMass` .
    - Name the columns as `size` and `mass` respectively.

```
#define x and y
x <- c(3.385,0.48,1.35,465,36.33,27.66,14.83,1.04,4.19,0.425,0.101,0.92,1,0.005,0.06,3.5,2,1.7,2
547,0.023,187.1,521,0.785,10,3.3,0.2,1.41,529,207,85,0.75,62,6654,3.5,6.8,35,4.05,0.12,0.023,0.0
1,1.4,250,2.5,55.5,100,52.16,10.55,0.55,60,3.6,4.288,0.28,0.075,0.122,0.048,192,3,160,0.9,1.62,
0.104,4.235)

y <- c(44.5,15.5,8.1,423,119.5,115,98.2,5.5,58,6.4,4,5.7,6.6,0.14,1,10.8,12.3,6.3,4603,0.3,419,6
55,3.5,115,25.6,5,17.5,680,406,325,12.3,1320,5712,3.9,179,56,17,1,0.4,0.25,12.5,490,12.1,175,15
7,440,179.5,2.4,81,21,39.2,1.9,1.2,3,0.33,180,25,169,2.6,11.4,2.5,50.4)

#creating dataframe
sizeMass <- data.frame(x,y)

colnames(sizeMass) <- c("size", "mass")

sizeMass
```

```
##        size    mass
## 1     3.385   44.50
## 2     0.480   15.50
## 3     1.350    8.10
## 4   465.000  423.00
## 5    36.330  119.50
## 6    27.660  115.00
## 7    14.830   98.20
## 8     1.040    5.50
## 9     4.190   58.00
## 10    0.425    6.40
## 11    0.101    4.00
## 12    0.920    5.70
## 13    1.000    6.60
## 14    0.005    0.14
## 15    0.060    1.00
## 16    3.500   10.80
## 17    2.000   12.30
## 18    1.700    6.30
## 19 2547.000 4603.00
## 20    0.023    0.30
## 21  187.100  419.00
## 22  521.000  655.00
## 23    0.785    3.50
## 24   10.000  115.00
## 25    3.300   25.60
## 26    0.200    5.00
## 27    1.410   17.50
## 28  529.000  680.00
## 29  207.000  406.00
## 30   85.000  325.00
## 31    0.750   12.30
## 32   62.000 1320.00
## 33 6654.000 5712.00
## 34    3.500    3.90
## 35    6.800  179.00
## 36   35.000   56.00
## 37    4.050   17.00
## 38    0.120    1.00
## 39    0.023    0.40
## 40    0.010    0.25
## 41    1.400   12.50
## 42  250.000  490.00
## 43    2.500   12.10
## 44   55.500  175.00
## 45  100.000  157.00
## 46   52.160  440.00
## 47   10.550  179.50
## 48    0.550    2.40
## 49   60.000   81.00
## 50    3.600   21.00
## 51    4.288   39.20
## 52    0.280    1.90
## 53    0.075    1.20
## 54    0.122    3.00
## 55    0.048    0.33
## 56  192.000  180.00
## 57    3.000   25.00
## 58  160.000  169.00
## 59    0.900    2.60
## 60    1.620   11.40
## 61    0.104    2.50
## 62    4.235   50.40
```

```
#plot data
ggplot(sizeMass, aes(size, mass)) +
    geom_point() +
    geom_smooth(method = "lm", color = "red", se = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



- Use the code below to plot the data.

```
ggplot(sizeMass, aes(size, mass)) +
  geom_point() +
  geom_smooth(method = "lm", color = "red", se = FALSE)
```

Question 8

- Transform the data in the plot to mitigate the “squashed” appearance.

1. Add two new columns named `size_log10` and `mass_log10` containing the log10 values of `size` and `mass` , respectively. Then use these new columns for plotting.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

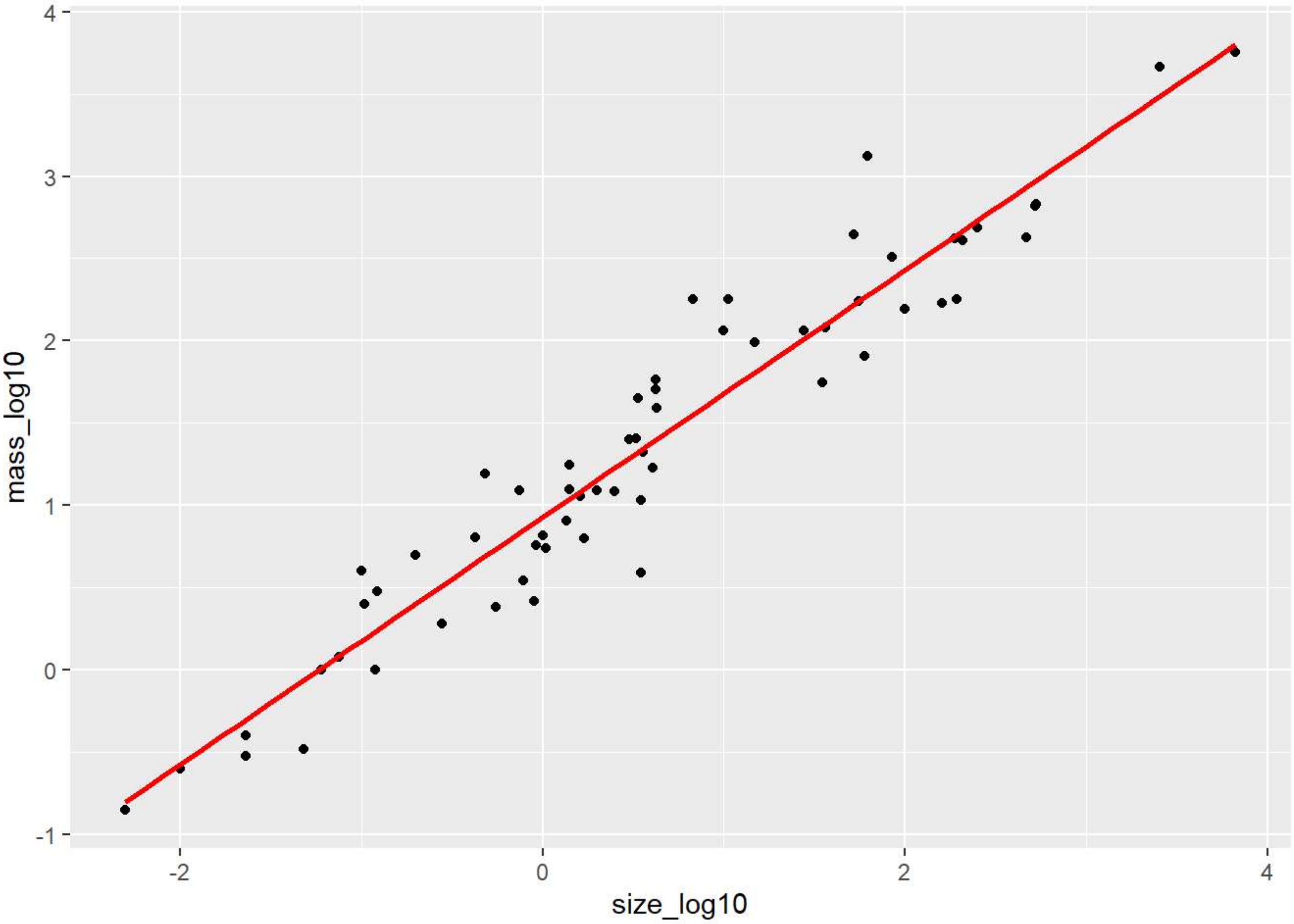
```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
#adding new columns
sizeMass <- sizeMass %>%
  mutate(size_log10 = log10(size))

sizeMass <- sizeMass %>%
  mutate(mass_log10 = log10(mass))

sizeMass
```

```
##        size    mass  size_log10  mass_log10
## 1     3.385   44.50  0.52955867  1.64836001
## 2     0.480   15.50 -0.31875876  1.19033170
## 3     1.350    8.10  0.13033377  0.90848502
## 4   465.000  423.00  2.66745295  2.62634037
## 5    36.330  119.50  1.56026540  2.07736791
## 6    27.660  115.00  1.44185218  2.06069784
## 7    14.830   98.20  1.17114115  1.99211149
## 8     1.040    5.50  0.01703334  0.74036269
## 9     4.190   58.00  0.62221402  1.76342799
## 10    0.425    6.40 -0.37161107  0.80617997
## 11    0.101    4.00 -0.99567863  0.60205999
## 12    0.920    5.70 -0.03621217  0.75587486
## 13    1.000    6.60  0.00000000  0.81954394
## 14    0.005    0.14 -2.30103000 -0.85387196
## 15    0.060    1.00 -1.22184875  0.00000000
## 16    3.500   10.80  0.54406804  1.03342376
## 17    2.000   12.30  0.30103000  1.08990511
## 18    1.700    6.30  0.23044892  0.79934055
## 19 2547.000 4603.00  3.40602894  3.66304097
## 20    0.023    0.30 -1.63827216 -0.52287875
## 21  187.100  419.00  2.27207379  2.62221402
## 22  521.000  655.00  2.71683772  2.81624130
## 23    0.785    3.50 -0.10513034  0.54406804
## 24   10.000  115.00  1.00000000  2.06069784
## 25    3.300   25.60  0.51851394  1.40823997
## 26    0.200    5.00 -0.69897000  0.69897000
## 27    1.410   17.50  0.14921911  1.24303805
## 28  529.000  680.00  2.72345567  2.83250891
## 29  207.000  406.00  2.31597035  2.60852603
## 30   85.000  325.00  1.92941893  2.51188336
## 31    0.750   12.30 -0.12493874  1.08990511
## 32   62.000 1320.00  1.79239169  3.12057393
## 33 6654.000 5712.00  3.82308280  3.75678820
## 34    3.500    3.90  0.54406804  0.59106461
## 35    6.800  179.00  0.83250891  2.25285303
## 36   35.000   56.00  1.54406804  1.74818803
## 37    4.050   17.00  0.60745502  1.23044892
## 38    0.120    1.00 -0.92081875  0.00000000
## 39    0.023    0.40 -1.63827216 -0.39794001
## 40    0.010    0.25 -2.00000000 -0.60205999
## 41    1.400   12.50  0.14612804  1.09691001
## 42  250.000  490.00  2.39794001  2.69019608
## 43    2.500   12.10  0.39794001  1.08278537
## 44   55.500  175.00  1.74429298  2.24303805
## 45  100.000  157.00  2.00000000  2.19589965
## 46   52.160  440.00  1.71733758  2.64345268
## 47   10.550  179.50  1.02325246  2.25406445
## 48    0.550    2.40 -0.25963731  0.38021124
## 49   60.000   81.00  1.77815125  1.90848502
## 50    3.600   21.00  0.55630250  1.32221929
## 51    4.288   39.20  0.63225478  1.59328607
## 52    0.280    1.90 -0.55284197  0.27875360
## 53    0.075    1.20 -1.12493874  0.07918125
## 54    0.122    3.00 -0.91364017  0.47712125
## 55    0.048    0.33 -1.31875876 -0.48148606
## 56  192.000  180.00  2.28330123  2.25527251
## 57    3.000   25.00  0.47712125  1.39794001
## 58  160.000  169.00  2.20411998  2.22788670
## 59    0.900    2.60 -0.04575749  0.41497335
## 60    1.620   11.40  0.20951501  1.05690485
## 61    0.104    2.50 -0.98296666  0.39794001
## 62    4.235   50.40  0.62685341  1.70243054
```

```
#plot data
ggplot(sizeMass, aes(size_log10, mass_log10)) +
    geom_point() +
    geom_smooth(method = "lm", color = "red", se = FALSE)
```

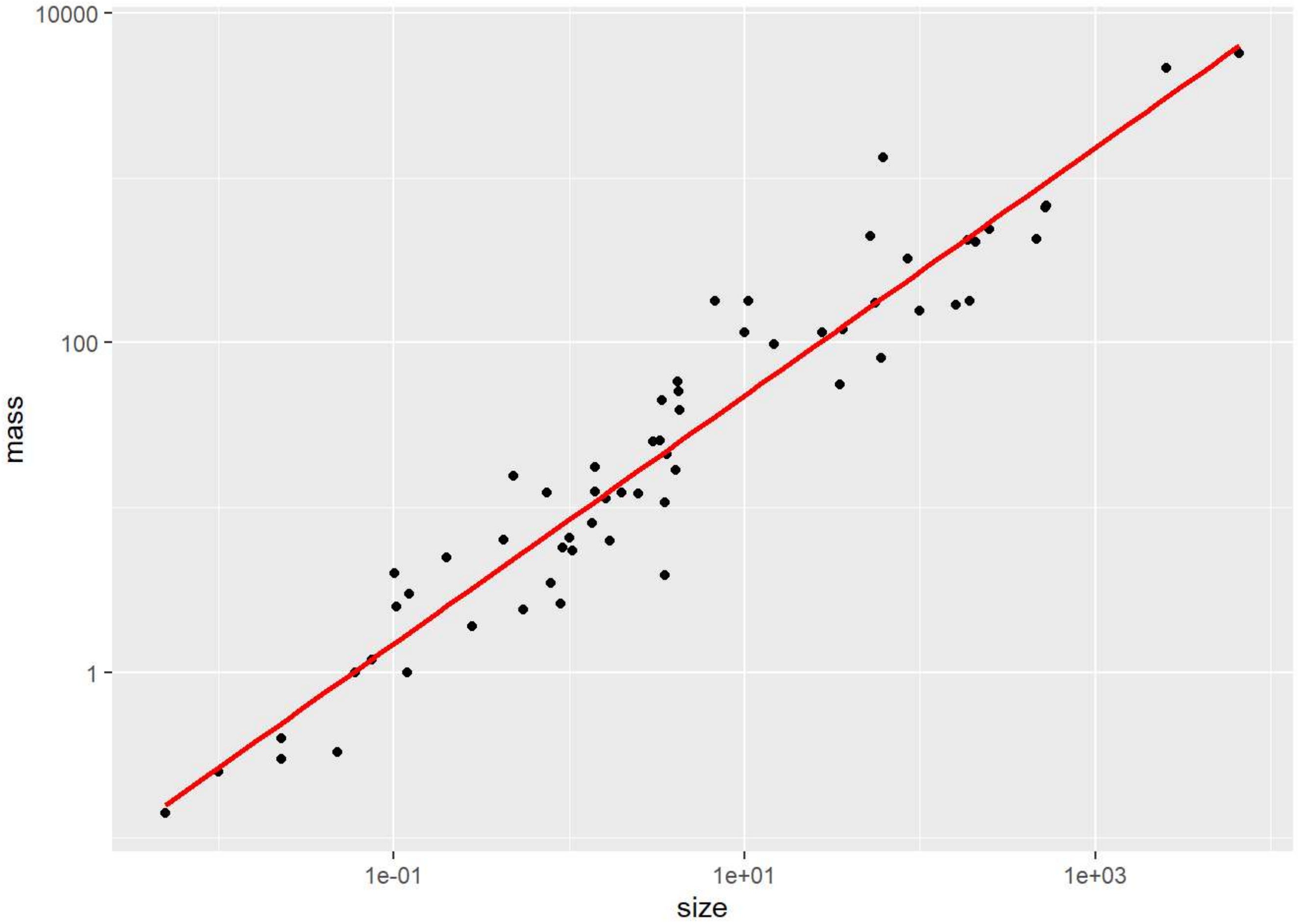
```
## `geom_smooth()` using formula = 'y ~ x'
```



2. Use `scale_x_log10()` and `scale_y_log10()` to set different scales for the x and y aesthetics.

```
ggplot(sizeMass, aes(size, mass)) +
  geom_point() +
  scale_x_log10() +
  scale_y_log10() +
  geom_smooth(method = "lm", color = "red", se = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```


mass
1e-01
1e+01
1e+03
size