
TITLE: "COMPSCIX 415.2 Homework 3" Author: "Ganesh Saravanan" Date: "6/25/2018" Output: pdf_document

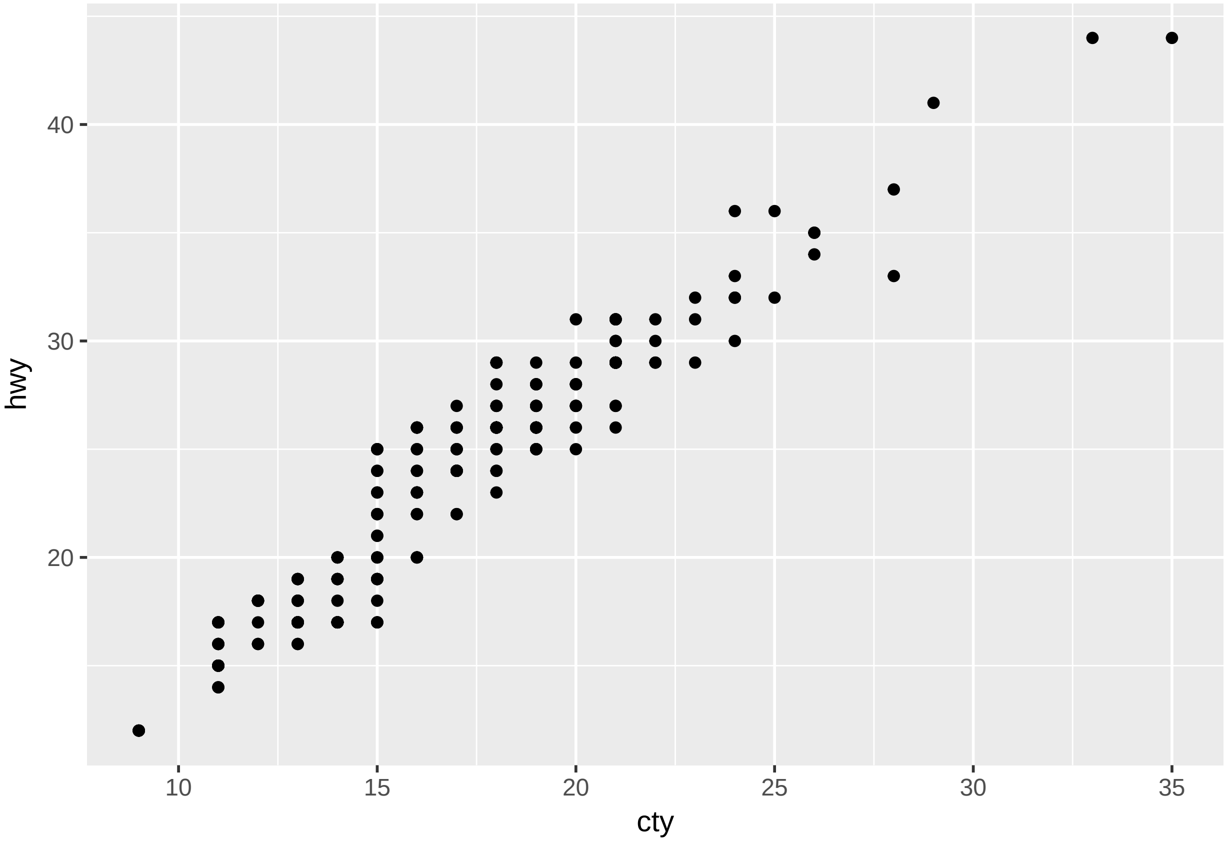
My Github repository for my assignments can be found at this URL:https://github.com/gsaravanan1/rstudiodemo.git

```
library(mdsr)
library(tidyverse)
library(ggplot2)
library(nycflights13)
```

3.8.1 Exercises

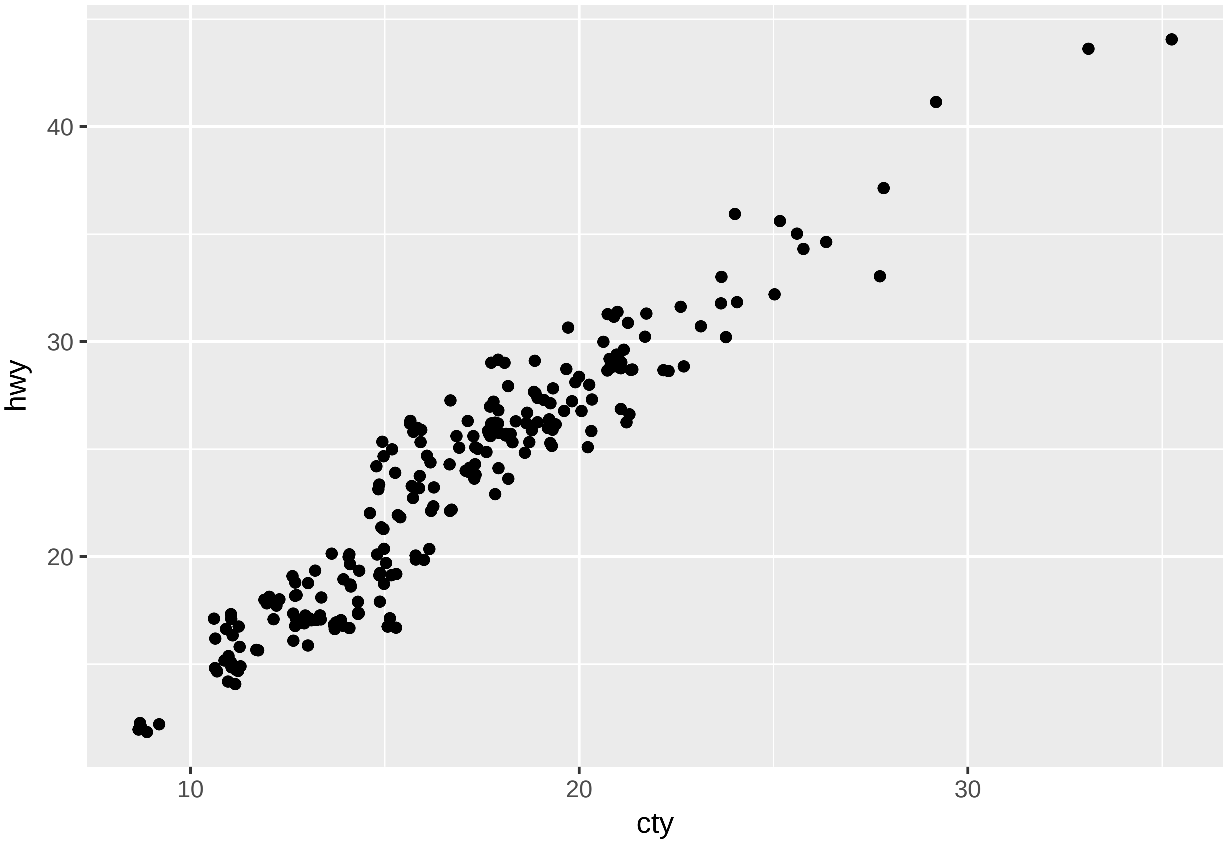
1 What is the problem with this plot? How could you improve it?

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
  geom_point()
```



ANSWER: Add a small amount of random variation

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
  geom_jitter()
```



2. What parameters to geom_jitter() control the amount of jittering?

ANSWER: width and height.

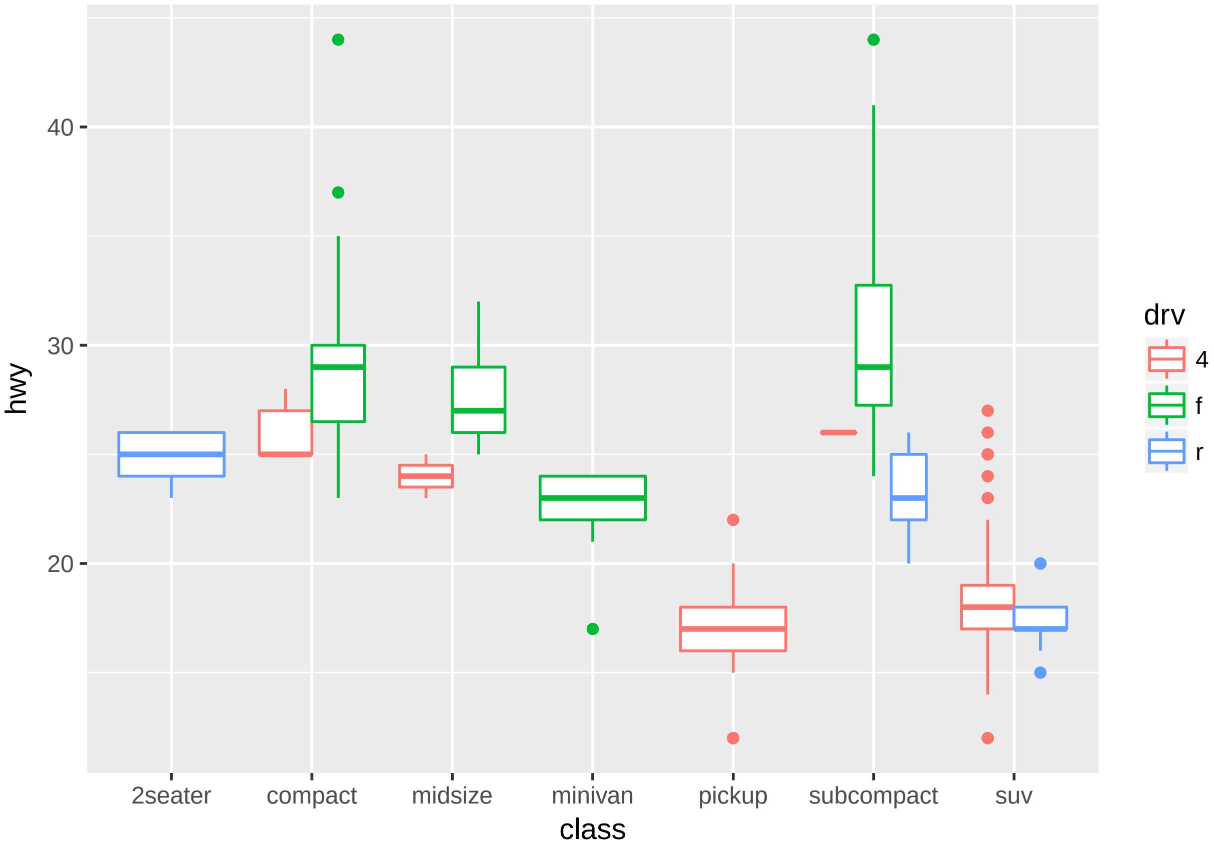
3. Compare and contrast geom_jitter() with geom_count().

ANSWER: Jittering is adding a small amount of random noise to data. It is often used to spread out points that would otherwise be overplotted. It is only effective in the non-continuous data case where overplotted points typically are surrounded by whitespace - jittering the data into the whitespace allows the individual points to be seen. It effectively un-discretizes the discrete data.

4. What's the default position adjustment for geom_boxplot()? Create a visualisation of the mpg dataset that demonstrates it.

ANSWER: The default position adjustment is position_dodge().

```r
ggplot(data = mpg, mapping = aes(x = class, y = hwy, color = drv)) +
  geom_boxplot(position = "dodge")
```



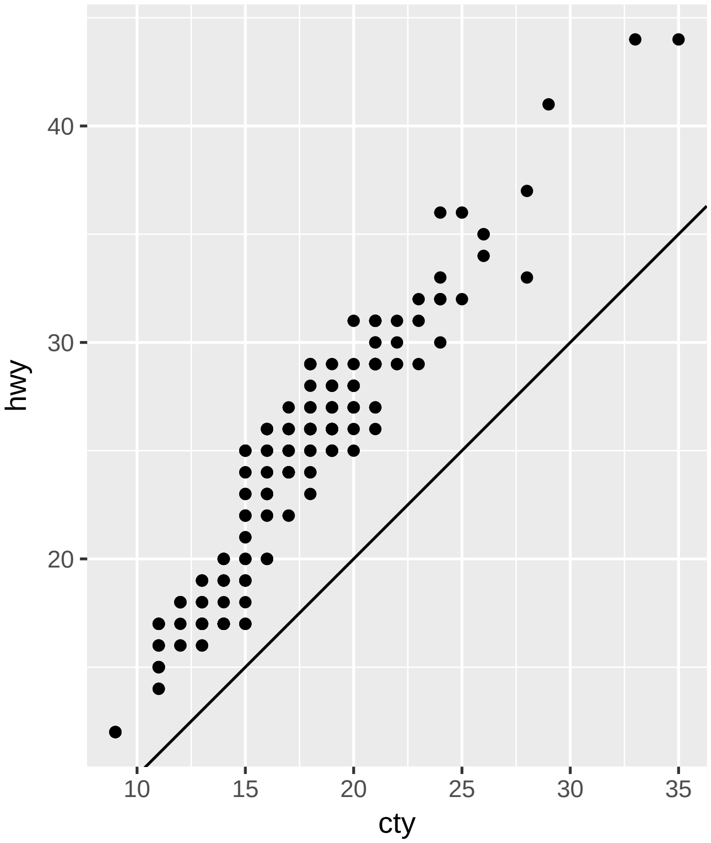
Section 3.9.1: #2 and #4 only

2 What does labs() do? Read the documentation.

ANSWER: labs() adds labels to the graph. You can add a title, subtitle, and a label for the x and y axes, as well as a caption.

4 What does the plot below tell you about the relationship between city and highway mpg? Why is coord_fixed() important? What does geom_abline() do?

```
ggplot(data = mpg, mapping = aes(x = cty, y = hwy)) +
  geom_point() +
  geom_abline() +
  coord_fixed()
```



ANSWER: Highway MPG is always (mostly) better than city MPG. coord_fixed() forces a specified ratio between the physical representation of data units on the axes. geom_abline() draws a line that, by default, has an intercept of 0 and slope of 1.

Section 4.4: #1 and #2 only

1 Why does this code not work?

```
my_variable <- 10
#my_varıable
```

ANSWER: Typo

2 Tweak each of the following R commands so that they run correctly:

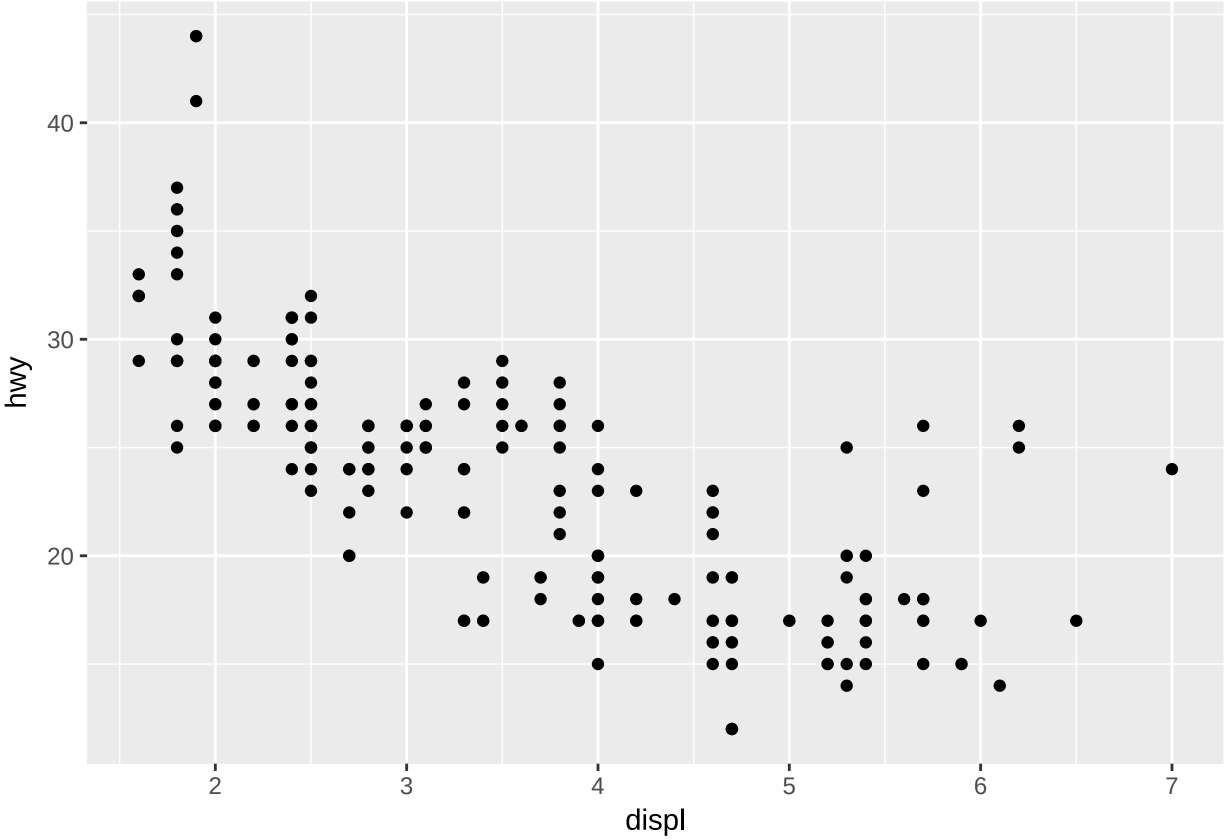
```
library(tidyverse)
```

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

fliter(mpg, cyl = 8)
filter(diamond, carat > 3)
```

ANSWER:

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



```
  filter(mpg, cyl == 8)
```

```
## # A tibble: 70 x 11
##    manufacturer model      displ  year   cyl trans  drv     cty   hwy fl
##    <chr>        <chr>      <dbl> <int> <int> <chr>  <chr> <int> <int> <chr>
##  1 audi         a6 quatt~    4.2  2008     8 auto(~ 4        16    23 p
##  2 chevrolet    c1500 su~    5.3  2008     8 auto(~ r        14    20 r
##  3 chevrolet    c1500 su~    5.3  2008     8 auto(~ r        11    15 e
##  4 chevrolet    c1500 su~    5.3  2008     8 auto(~ r        14    20 r
##  5 chevrolet    c1500 su~    5.7  1999     8 auto(~ r        13    17 r
##  6 chevrolet    c1500 su~    6     2008     8 auto(~ r        12    17 r
##  7 chevrolet    corvette     5.7  1999     8 manua~ r        16    26 p
##  8 chevrolet    corvette     5.7  1999     8 auto(~ r        15    23 p
##  9 chevrolet    corvette    6.2  2008     8 manua~ r        16    26 p
## 10 chevrolet    corvette    6.2  2008     8 auto(~ r        15    25 p
## # ... with 60 more rows, and 1 more variable: class <chr>
```

```
filter(diamonds, carat > 3)
```

```
## # A tibble: 32 x 10
##    carat cut     color clarity depth table price     x     y     z
##    <dbl> <ord>   <ord> <ord>   <dbl> <dbl> <int> <dbl> <dbl> <dbl>
##  1  3.01 Premium I     I1       62.7    58  8040  9.1   8.97  5.67
##  2  3.11 Fair    J     I1       65.9    57  9823  9.15  9.02  5.98
##  3  3.01 Premium F     I1       62.2    56  9925  9.24  9.13  5.73
##  4  3.05 Premium E     I1       60.9    58 10453  9.26  9.25  5.66
##  5  3.02 Fair    I     I1       65.2    56 10577  9.11  9.02  5.91
##  6  3.01 Fair    H     I1       56.1    62 10761  9.54  9.38  5.31
##  7  3.65 Fair    H     I1       67.1    53 11668  9.53  9.48  6.38
##  8  3.24 Premium H     I1       62.1    58 12300  9.44  9.4   5.85
##  9  3.22 Ideal   I     I1       62.6    55 12545  9.49  9.42  5.92
## 10  3.5  Ideal   H     I1       62.8    57 12587  9.65  9.59  6.03
## # ... with 22 more rows
```

Section 5.2.4: #1, #3 and #4 only.

1. Find all flights that

1.1. Had an arrival delay of two or more hours.

filter(flights, arr_delay>=120)

1.2. Flew to Houston (IAH or HOU)

filter(flights, dest == 'IAH' | dest == 'HOU')

1.3. Were operated by United, American, or Delta

filter(flights, carrier == 'UA' | carrier == 'AA' | carrier == 'DL')

1.4. Departed in summer (July, August, and September)

filter(flights, month >= 7 & month <= 9)

1.5. Arrived more than two hours late, but didn't leave late

filter(flights, arr_delay > 120, dep_delay <= 0)

1.6. Were delayed by at least an hour, but made up over 30 minutes in flight

filter(flights, dep_delay >= 60, dep_delay-arr_delay > 30)

1.7. Departed between midnight and 6am (inclusive)

filter(flights, dep_time <=600 | dep_time == 2400)

3 How many flights have a missing dep_time? What other variables are missing? What might these rows represent?

```r
summary(flights)
```

```
##       year          month             day           dep_time
##  Min.   :2013   Min.   : 1.000   Min.   : 1.00   Min.   :   1
##  1st Qu.:2013   1st Qu.: 4.000   1st Qu.: 8.00   1st Qu.: 907
##  Median :2013   Median : 7.000   Median :16.00   Median :1401
##  Mean   :2013   Mean   : 6.549   Mean   :15.71   Mean   :1349
##  3rd Qu.:2013   3rd Qu.:10.000   3rd Qu.:23.00   3rd Qu.:1744
##  Max.   :2013   Max.   :12.000   Max.   :31.00   Max.   :2400
##                                                  NA's   :8255
##  sched_dep_time   dep_delay          arr_time    sched_arr_time
##  Min.   : 106   Min.   : -43.00   Min.   :   1   Min.   :   1
##  1st Qu.: 906   1st Qu.:  -5.00   1st Qu.:1104   1st Qu.:1124
##  Median :1359   Median :  -2.00   Median :1535   Median :1556
##  Mean   :1344   Mean   :  12.64   Mean   :1502   Mean   :1536
##  3rd Qu.:1729   3rd Qu.:  11.00   3rd Qu.:1940   3rd Qu.:1945
##  Max.   :2359   Max.   :1301.00   Max.   :2400   Max.   :2359
##                 NA's   :8255      NA's   :8713
##    arr_delay          carrier              flight       tailnum
##  Min.   : -86.000   Length:336776      Min.   :   1   Length:336776
##  1st Qu.: -17.000   Class :character   1st Qu.: 553   Class :character
##  Median :  -5.000   Mode  :character   Median :1496   Mode  :character
##  Mean   :   6.895                      Mean   :1972
##  3rd Qu.:  14.000                      3rd Qu.:3465
##  Max.   :1272.000                      Max.   :8500
##  NA's   :9430
##     origin              dest              air_time        distance
##  Length:336776      Length:336776      Min.   : 20.0   Min.   :  17
##  Class :character   Class :character   1st Qu.: 82.0   1st Qu.: 502
##  Mode  :character   Mode  :character   Median :129.0   Median : 872
##                                        Mean   :150.7   Mean   :1040
##                                        3rd Qu.:192.0   3rd Qu.:1389
##                                        Max.   :695.0   Max.   :4983
##                                        NA's   :9430
##       hour           minute        time_hour
##  Min.   : 1.00   Min.   : 0.00   Min.   :2013-01-01 05:00:00
##  1st Qu.: 9.00   1st Qu.: 8.00   1st Qu.:2013-04-04 13:00:00
##  Median :13.00   Median :29.00   Median :2013-07-03 10:00:00
##  Mean   :13.18   Mean   :26.23   Mean   :2013-07-03 05:02:36
##  3rd Qu.:17.00   3rd Qu.:44.00   3rd Qu.:2013-10-01 07:00:00
##  Max.   :23.00   Max.   :59.00   Max.   :2013-12-31 23:00:00
##
```

ANSWER :

8255 flights have a missing dep_time, 8255 have a missing dep_delay, 8713 have a missing arr_time, 9430 have a missing arr_delay, and 9430 have a missing air_time. We can speculate that these are flights that failed to depart or arrive, since a flight that departs normally but is then rerouted will probably have a normally recorded departure but no similar record for it's arrival. However, these could also just be lost data about perfectly normal flights.

4 Why is NA ^ 0 not missing? Why is NA | TRUE not missing? Why is FALSE & NA not missing? Can you figure out the general rule? (NA * 0 is a tricky counterexample!)

ANSWER: NA ^ 0 evaluates to 1 because anything to the power of 0 is 1, so although we didn't know the original value, we know it's being taken to the zeroth power.

With NA | TRUE, since the | operator returns TRUE if either of the terms are true, the whole expression returns true because the right half returns true. This is easier to see in an expression like NA | 5<10 (since 5 is indeed less than 10).

For the next example, we know that & returns TRUE when both terms are true. So, for example, TRUE & TRUE evaluates to TRUE. In FALSE & NA, one of the terms is false, so the expression evaluates to FALSE. As does something like FALSE & TRUE.

NA * 0 could be argued to be because the NA could represent Inf, and Inf * 0 is NaN (Not a Number), rather than NA. However, I suspect that these results are dictated as much by what answer is natural, quick and sensible in C as by mathematical edge cases.

Section 5.4.1: #1 and #3 only

1 Brainstorm as many ways as possible to select dep_time, dep_delay, arr_time, and arr_delay from flights.

ANSWER:

```
select(flights, dep_time, dep_delay, arr_time, arr_delay)
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517         2      830        11
##  2      533         4      850        20
##  3      542         2      923        33
##  4      544        -1     1004       -18
##  5      554        -6      812       -25
##  6      554        -4      740        12
##  7      555        -5      913        19
##  8      557        -3      709       -14
##  9      557        -3      838        -8
## 10      558        -2      753         8
## # ... with 336,766 more rows
```

```
select(flights, dep_time,  dep_delay, arr_time, arr_delay)
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517         2      830        11
##  2      533         4      850        20
##  3      542         2      923        33
##  4      544        -1     1004       -18
##  5      554        -6      812       -25
##  6      554        -4      740        12
##  7      555        -5      913        19
##  8      557        -3      709       -14
##  9      557        -3      838        -8
## 10      558        -2      753         8
## # ... with 336,766 more rows
```

```
select(flights, c(dep_time,  dep_delay, arr_time, arr_delay))
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517         2      830        11
##  2      533         4      850        20
##  3      542         2      923        33
##  4      544        -1     1004       -18
##  5      554        -6      812       -25
##  6      554        -4      740        12
##  7      555        -5      913        19
##  8      557        -3      709       -14
##  9      557        -3      838        -8
## 10      558        -2      753         8
## # ... with 336,766 more rows
```

```
flights %>% select(dep_time,  dep_delay, arr_time, arr_delay)
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517         2      830        11
##  2      533         4      850        20
##  3      542         2      923        33
##  4      544        -1     1004       -18
##  5      554         -6      812        -25
##  6      554         -4      740         12
##  7      555         -5      913         19
##  8      557         -3      709        -14
##  9      557         -3      838         -8
## 10      558         -2      753          8
## # ... with 336,766 more rows
```

```
flights %>% select_("dep_time",  "dep_delay", "arr_time", "arr_delay")
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517          2      830         11
##  2      533          4      850         20
##  3      542          2      923         33
##  4      544         -1     1004        -18
##  5      554         -6      812        -25
##  6      554         -4      740         12
##  7      555         -5      913         19
##  8      557         -3      709        -14
##  9      557         -3      838         -8
## 10      558         -2      753          8
## # ... with 336,766 more rows
```

```
flights %>% select_(.dots=c("dep_time",  "dep_delay", "arr_time", "arr_delay"))
```

```
## # A tibble: 336,776 x 4
##    dep_time dep_delay arr_time arr_delay
##       <int>     <dbl>    <int>     <dbl>
##  1      517          2      830         11
##  2      533          4      850         20
##  3      542          2      923         33
##  4      544         -1     1004        -18
##  5      554         -6      812        -25
##  6      554         -4      740         12
##  7      555         -5      913         19
##  8      557         -3      709        -14
##  9      557         -3      838         -8
## 10      558         -2      753          8
## # ... with 336,766 more rows
```

3 What does the one_of() function do? Why might it be helpful in conjunction with this vector?

ANSWER : one_of() allows for subset-matching

```
vars <- c("year", "month", "day", "dep_delay", "arr_delay")
flights %>% select(one_of(vars))
```

```
## # A tibble: 336,776 x 5
##     year month   day dep_delay arr_delay
##    <int> <int> <int>     <dbl>     <dbl>
##  1  2013     1     1         2        11
##  2  2013     1     1         4        20
##  3  2013     1     1         2        33
##  4  2013     1     1        -1       -18
##  5  2013     1     1        -6       -25
##  6  2013     1     1        -4        12
##  7  2013     1     1        -5        19
##  8  2013     1     1        -3       -14
##  9  2013     1     1        -3        -8
## 10  2013     1     1        -2         8
## # ... with 336,766 more rows
```