Homework 3

[Ra-Zakee Muhammad]

Due 9/21/2021

Classmates/other resources consulted: [type answer here]

```
library(ggplot2)
library(dplyr)
library(nycflights13)
```

Question 1 (3 points)

Explain why the following results return FALSE, and how you should compare these values instead. You can give one explanation for both, you do not need to give a separate explanation for each. (note: every computer is different, and while they both return FALSE on my computer, they may not both return FALSE on your computer. Regardless, discuss why FALSE might show up as an output and what you should do instead).

```
?near()
```

```
sqrt(3)^2 == 3
```

```
## [1] FALSE
```

```
1.45 - 0.55 == 2.45 - 1.55
```

```
## [1] FALSE
```

These are different data types, one is an integer another is a dbl therefore in order to compare the values one needs to use the near function.

```
near(sqrt(3)^2, 3)
```

```
## [1] TRUE
```

```
near(1.45 - 0.55 , 2.45 - 1.55)
```

```
## [1] TRUE
```

Question 2 (12 points)

Using the flights data set, output a tibble consisting of flights meeting the following criteria. For each, indicate how many flights there are.

a. **flights with an arrival delay of more than two hours**

```
flights %>% filter(!is.na(arr_delay), arr_delay > 2)
```

```
## # A tibble: 123,096 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      554            558        -4      740            728
##  5  2013     1     1      555            600        -5      913            854
##  6  2013     1     1      558            600        -2      753            745
##  7  2013     1     1      558            600        -2      924            917
##  8  2013     1     1      559            600        -1      941            910
##  9  2013     1     1      600            600         0      837            825
## 10  2013     1     1      602            605        -3      821            805
## # ... with 123,086 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

b. **flights operated by United (UA), American (AA), or Delta (DL) (whether or not they took off)**

```
flights %>%
  filter( carrier == "UA" | carrier == "AA" | carrier == "DL" )
```

```
## # A tibble: 139,504 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      554            600        -6      812            837
##  5  2013     1     1      554            558        -4      740            728
##  6  2013     1     1      558            600        -2      753            745
##  7  2013     1     1      558            600        -2      924            917
##  8  2013     1     1      558            600        -2      923            937
##  9  2013     1     1      559            600        -1      941            910
## 10  2013     1     1      559            600        -1      854            902
## # ... with 139,494 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

c. **flights *not* operated by United, American, or Delta (whether or not they took off)**

```
flights %>%
  filter( carrier != "UA" , carrier != "AA" , carrier != "DL" )
```

```
## # A tibble: 197,272 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      544            545        -1     1004           1022
##  2  2013     1     1      555            600        -5      913            854
##  3  2013     1     1      557            600        -3      709            723
##  4  2013     1     1      557            600        -3      838            846
##  5  2013     1     1      558            600        -2      849            851
##  6  2013     1     1      558            600        -2      853            856
##  7  2013     1     1      559            559         0      702            706
##  8  2013     1     1      600            600         0      851            858
##  9  2013     1     1      600            600         0      837            825
## 10  2013     1     1      601            600         1      844            850
## # ... with 197,262 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

d. **flights that were scheduled to depart in February, March, May, July, September, or November (use %in%)**

```
flights %>%
  filter( month %in% c(2,3,5,7,9,11))
```

```
## # A tibble: 166,848 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013    11     1        5           2359         6      352            345
##  2  2013    11     1       35           2250       105      123           2356
##  3  2013    11     1      455            500        -5      641            651
##  4  2013    11     1      539            545        -6      856            827
##  5  2013    11     1      542            545        -3      831            855
##  6  2013    11     1      549            600       -11      912            923
##  7  2013    11     1      550            600       -10      705            659
##  8  2013    11     1      554            600        -6      659            701
##  9  2013    11     1      554            600        -6      826            827
## 10  2013    11     1      554            600        -6      749            751
## # ... with 166,838 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

e. **flights that departed between 7am and 8am, including both 7am and 8am**

```
flights %>%
  filter(dep_time >= 700 , dep_time <= 800) %>% arrange((dep_time))
```

```
## # A tibble: 22,009 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     2      700            630        30      917            840
##  2  2013     1     2      700            700         0      851            850
##  3  2013     1     2      700            700         0     1017           1015
##  4  2013     1     3      700            700         0      851            836
##  5  2013     1     4      700            700         0      941           1025
##  6  2013     1     4      700            700         0      956           1025
##  7  2013     1     5      700            700         0     1014           1025
##  8  2013     1     6      700            615        45     1001            921
##  9  2013     1     7      700            704        -4      912            932
## 10  2013     1     7      700            700         0      959           1045
## # ... with 21,999 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

Question 3 (6 points)

a. **Why does TRUE | NA not result in NA?**

because the command | only requires that one of the values on either side be true to return a true value

b. **Why does NA & FALSE not result in NA?**

Because when using the "&" or "," command if either side has a false value then the command returns false

c. **Why does NA^0 not result in NA?**

Probably because the r language reads any value numeric or logical raised to the 0th power as 1.

```
FALSE^0
```

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

```
TRUE^0
```

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

Question 4 (6 points)

Use arrange() to do the following

a. **Sort flights to find the flights with the shortest air time. Among the ten flights with the shortest airtime, where did they go?**

```
flights %>%
  filter(!is.na(air_time)) %>%
  arrange(air_time) %>%
  select(air_time, dest) %>%
  head(10) %>%
  mutate(n = row_number())
```

```
## # A tibble: 10 x 3
##    air_time dest      n
##       <dbl> <chr> <int>
##  1       20 BDL       1
##  2       20 BDL       2
##  3       21 BDL       3
##  4       21 PHL       4
##  5       21 BDL       5
##  6       21 PHL       6
##  7       21 BOS       7
##  8       21 PHL       8
##  9       21 BDL       9
## 10       21 BDL      10
```

“Bradley International Airport in Hartford County, Connecticut” ; “Philadelphia International Airport” ; “Boston Logan International Airport”

b. **Sort flights to find the flights that had the longest arrival delay. Among the five flights that had the longest arrival delays, what airport did four of them originate from?**

```
flights %>%
  filter(!is.na(arr_delay)) %>%
  arrange(desc(arr_delay))  %>%
  select(arr_delay, origin)%>%
  head(10) %>%
  mutate(n = row_number())
```

```
## # A tibble: 10 x 3
##    arr_delay origin     n
##        <dbl> <chr>  <int>
##  1      1272 JFK        1
##  2      1127 JFK        2
##  3      1109 EWR        3
##  4      1007 JFK        4
##  5       989 JFK        5
##  6       931 JFK        6
##  7       915 LGA        7
##  8       895 LGA        8
##  9       878 EWR        9
## 10       875 EWR       10
```

“John F. Kennedy International Airport in Queens, New York” ; “Newark Liberty International Airport”, “Drummond Twins International Airport in Cochrane, Chile”

Question 5 (9 points)

a. **In the flights data set, give a command that returns all columns/factors, in order, from carrier to destination.**

```
flights %>%
  select(carrier: dest)
```

```
## # A tibble: 336,776 x 5
##    carrier flight tailnum origin dest 
##    <chr>    <int> <chr>   <chr>  <chr>
##  1 UA        1545 N14228  EWR    IAH  
##  2 UA        1714 N24211  LGA    IAH  
##  3 AA        1141 N619AA  JFK    MIA  
##  4 B6         725 N804JB  JFK    BQN  
##  5 DL         461 N668DN  LGA    ATL  
##  6 UA        1696 N39463  EWR    ORD  
##  7 B6         507 N516JB  EWR    FLL  
##  8 EV        5708 N829AS  LGA    IAD  
##  9 B6          79 N593JB  JFK    MCO  
## 10 AA         301 N3ALAA  LGA    ORD  
## # ... with 336,766 more rows
```

b. **In the flights data set, give a command that uses “contains” to return the three columns/factors related to arrival time.**

```
flights %>%
  select(contains("arr_"))
```

```
## # A tibble: 336,776 x 3
##    arr_time sched_arr_time arr_delay
##       <int>          <int>     <dbl>
##  1      830            819        11
##  2      850            830        20
##  3      923            850        33
##  4     1004           1022       -18
##  5      812            837       -25
##  6      740            728        12
##  7      913            854        19
##  8      709            723       -14
##  9      838            846        -8
## 10      753            745         8
## # ... with 336,766 more rows
```

c. **In the flights data set, write a command that returns all columns/factors except for tailnum, hour, minute, and time_hour.**

```
flights %>%
  select(-tailnum , -hour , -minute, -time_hour)
```

```
## # A tibble: 336,776 x 15
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 336,766 more rows, and 7 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, origin <chr>, dest <chr>, air_time <dbl>,
## #   distance <dbl>
```

Question 6 (6 points)

a. **In the mpg data set, add a column/factor that gives the average of the city miles per gallon and the highway miles per gallon**

```
mpg %>%
  mutate(average_cty_hwy = (cty + hwy)/2)
```

```
## # A tibble: 234 x 12
##    manufacturer model      displ  year   cyl trans drv     cty   hwy fl    class
##    <chr>        <chr>      <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
##  1 audi         a4           1.8  1999     4 auto~ f        18    29 p     comp~
##  2 audi         a4           1.8  1999     4 manu~ f        21    29 p     comp~
##  3 audi         a4           2    2008     4 manu~ f        20    31 p     comp~
##  4 audi         a4           2    2008     4 auto~ f        21    30 p     comp~
##  5 audi         a4           2.8  1999     6 auto~ f        16    26 p     comp~
##  6 audi         a4           2.8  1999     6 manu~ f        18    26 p     comp~
##  7 audi         a4           3.1  2008     6 auto~ f        18    27 p     comp~
##  8 audi         a4 quattro   1.8  1999     4 manu~ 4        18    26 p     comp~
##  9 audi         a4 quattro   1.8  1999     4 auto~ 4        16    25 p     comp~
## 10 audi         a4 quattro   2    2008     4 manu~ 4        20    28 p     comp~
## # ... with 224 more rows, and 1 more variable: average_cty_hwy <dbl>
```

b. **In the flights data set, add a column/factor that is the larger of the departure delay and the arrival delay.**

```
flights %>%
  mutate(max_departure_arrival_delay = pmax(dep_delay, arr_delay))
```

```
## # A tibble: 336,776 x 20
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 336,766 more rows, and 12 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
## #   max_departure_arrival_delay <dbl>
```

Question 7 (12 points)

In this question, you'll need to determine the correct transformation(s) to do on the data set flights to produce the desired tibbles and answer the question. *In each part, use pipes.*

a. **Give a command that returns all the data in flights for factors that contain the letter "d" in the factor name, and also includes the month and day of each flight. What are the dimensions of your new tibble?**

```
flights %>%
  select(contains("d")| month)
```

```
## # A tibble: 336,776 x 9
##      day dep_time sched_dep_time dep_delay sched_arr_time arr_delay dest
##    <int>    <int>          <int>     <dbl>          <int>     <dbl> <chr>
##  1     1      517            515         2            819        11 IAH
##  2     1      533            529         4            830        20 IAH
##  3     1      542            540         2            850        33 MIA
##  4     1      544            545        -1           1022       -18 BQN
##  5     1      554            600        -6            837       -25 ATL
##  6     1      554            558        -4            728        12 ORD
##  7     1      555            600        -5            854        19 FLL
##  8     1      557            600        -3            723       -14 IAD
##  9     1      557            600        -3            846        -8 MCO
## 10     1      558            600        -2            745         8 ORD
## # ... with 336,766 more rows, and 2 more variables: distance <dbl>, month <int>
```

b. **Recall speed in miles per hour is distance / air_time * 60. What two airlines had all six of the fastest flights?**

```
flights %>%
  mutate(speed = distance / air_time * 60) %>%
  arrange(desc(speed))
```

```
## # A tibble: 336,776 x 20
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     5    25     1709           1700         9     1923           1937
##  2  2013     7     2     1558           1513        45     1745           1719
##  3  2013     5    13     2040           2025        15     2225           2226
##  4  2013     3    23     1914           1910         4     2045           2043
##  5  2013     1    12     1559           1600        -1     1849           1917
##  6  2013    11    17      650            655        -5     1059           1150
##  7  2013     2    21     2355           2358        -3      412            438
##  8  2013    11    17      759            800        -1     1212           1255
##  9  2013    11    16     2003           1925        38       17             36
## 10  2013    11    16     2349           2359       -10      402            440
## # ... with 336,766 more rows, and 12 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
## #   speed <dbl>
```

DL (delta) and EV(express jet)

c. **Consider the flights that arrived more than an hour late but didn’t leave late. How many of these were there in September?**

```
flights %>%
  filter(arr_delay > 60, dep_delay <= 0, month == 9) %>%
  mutate(n = row_number()) %>%
  summarise(max(n))
```

```
## # A tibble: 1 x 1
##   ‘max(n)‘
##      <int>
## 1       52
```

d. **The information about flights suggests that it should always be the case that sched_dep_time + dep_delay = dep_time. Create a new tibble with a column whose value is sched_dep_time + dep_delay and see for how many rows this is different from dep_delay. Hypothesize about what might be occurring here. (Hint: use multiple transformations)**

```
flights %>%
  mutate(dep_time_1 = sched_dep_time + dep_delay) %>%
  mutate(diff_dep_time = dep_time_1 - dep_time) %>%
  filter(diff_dep_time != 0)%>%
  mutate(n = row_number())
```

```
## # A tibble: 99,777 x 22
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      554            600        -6      812            837
##  2  2013     1     1      555            600        -5      913            854
##  3  2013     1     1      557            600        -3      709            723
##  4  2013     1     1      557            600        -3      838            846
##  5  2013     1     1      558            600        -2      753            745
##  6  2013     1     1      558            600        -2      849            851
##  7  2013     1     1      558            600        -2      853            856
##  8  2013     1     1      558            600        -2      924            917
##  9  2013     1     1      558            600        -2      923            937
## 10  2013     1     1      559            600        -1      941            910
## # ... with 99,767 more rows, and 14 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>,
## #   dep_time_1 <dbl>, diff_dep_time <dbl>, n <int>
```

The problem is that r isn't automatically thinking about addition and subtraction in terms of mod 60 (which is the addition that is necessary to maintain the hour structure) while it is helpful that scheduled departure time is in military time, the data type of the column is still integers which means that if we are representing 6:00 as 600 then adding a departure delay of -6 minutes will give us 594 as opposed to 5:54. This explains why many of the non-zero values for dep_time_1 are 40 because 50 + 40 is 90.

Question 8 (3 points)

Explain why the comparison x == y in the following code doesn't produce FALSE, since x and y are different vectors.

```
x <- c(5,2,9,4)
y <- c(5,2,11,6)
x == y
```

```
## [1]  TRUE  TRUE FALSE FALSE
```

Because in r when you compare vectors using equality or inequalities, the language compares by common indexed entries as opposed to the entire vectors. this way, you get dimension number of logical values if in fact the two vectors share dimension.

Question 9 (3 points)

> **Summarize the starwars dataset: find the minimum mass, maximum height, and median birth year of all characters in the data set (for whom these values are defined). Your answer should be a tibble that only contains these three values.**

```
starwars %>%
  arrange(mass)
```

```
## # A tibble: 87 x 14
##    name     height  mass hair_color skin_color eye_color birth_year sex   gender
##    <chr>     <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
##  1 Ratts T~     79    15 none       grey, blue unknown           NA male  mascu~
##  2 Yoda         66    17 white      green      brown            896 male  mascu~
##  3 Wicket ~     88    20 brown      brown      brown              8 male  mascu~
##  4 R2-D2        96    32 <NA>       white, bl~ red             33 none  mascu~
##  5 R5-D4        97    32 <NA>       white, red red             NA none  mascu~
##  6 Sebulba     112    40 none       grey, red  orange          NA male  mascu~
##  7 Dud Bolt     94    45 none       blue, grey yellow          NA male  mascu~
##  8 Padmé A~    165    45 brown      light      brown           46 fema~ femin~
##  9 Wat Tam~    193    48 none       green, gr~ unknown         NA male  mascu~
## 10 Sly Moo~    178    48 none       pale       white           NA <NA>  <NA>
## # ... with 77 more rows, and 5 more variables: homeworld <chr>, species <chr>,
## #   films <list>, vehicles <list>, starships <list>
```

```
starwars %>%
  filter(!is.na(mass), !is.na(height), !is.na(birth_year)) %>%
  summarise(min(mass), max(height), median(birth_year))
```

```
## # A tibble: 1 x 3
##   ‘min(mass)‘ ‘max(height)‘ ‘median(birth_year)‘
##         <dbl>         <int>                <dbl>
## 1          17           228                 46.5
```

Question 10 (15 points)

In the flights data set, tailnum identifies the particular plane that was used in each flight. Throughout this question, use pipes.

a. **First, modify the flights data set so that it only contains flights that are not cancelled. Throughout this question, we'll only want to consider flights that were not cancelled; you can either store these non-cancelled flights in a new tibble, or do this same modification in each part below.**

```
flights %>%
  filter(!is.na(dep_time), !is.na(arr_time))
```

```
## # A tibble: 328,063 x 19
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 328,053 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

b. **Group the flights data by tailnum. How many different tail numbers are there?**

```r
flights %>%
  filter(!is.na(dep_time), !is.na(arr_time)) %>%
  group_by(tailnum)
```

```
## # A tibble: 328,063 x 19
## # Groups:   tailnum [4,037]
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      542            540         2      923            850
##  4  2013     1     1      544            545        -1     1004           1022
##  5  2013     1     1      554            600        -6      812            837
##  6  2013     1     1      554            558        -4      740            728
##  7  2013     1     1      555            600        -5      913            854
##  8  2013     1     1      557            600        -3      709            723
##  9  2013     1     1      557            600        -3      838            846
## 10  2013     1     1      558            600        -2      753            745
## # ... with 328,053 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

4037 groups

c. **Make a summary tibble that shows, for each tailnum, how many times that plane flew out of new york and what the average departure delay of those flights was.**

```r
flights %>%
  filter(!is.na(dep_time), !is.na(arr_time)) %>%
  filter(origin == "JFK"| origin == "LGA"| origin == "EWR") %>%
  group_by(tailnum) %>%
  summarise(n_rows = n(), avg_dep = mean(dep_delay))
```

```
## # A tibble: 4,037 x 3
##    tailnum n_rows avg_dep
##    <chr>    <int>   <dbl>
##  1 D942DN       4  31.5  
##  2 N0EGMQ     354   8.49 
##  3 N10156     146  17.8  
##  4 N102UW      48   8    
##  5 N103US      46  -3.20 
##  6 N104UW      46  10.1  
##  7 N10575     271  22.3  
##  8 N105UW      45   2.58 
##  9 N107US      41  -0.463
## 10 N108UW      60   4.22 
## # ... with 4,027 more rows
```

d. **What tailnum made the most flights out of new york in 2013?**

```
flights %>%
  filter(!is.na(dep_time), !is.na(arr_time)) %>%
  filter(year == 2013)%>% filter( origin == "JFK"| origin == "LGA"| origin == "EWR") %>% group_by(tailnu
```

```
## # A tibble: 1 x 3
##   tailnum n_rows avg_dep
##   <chr>    <int>   <dbl>
## 1 N725MQ     546    6.87
```

e. **Filter your tibble from (a) so that it only contains tail numbers that made at least 100 flights out of New York.**

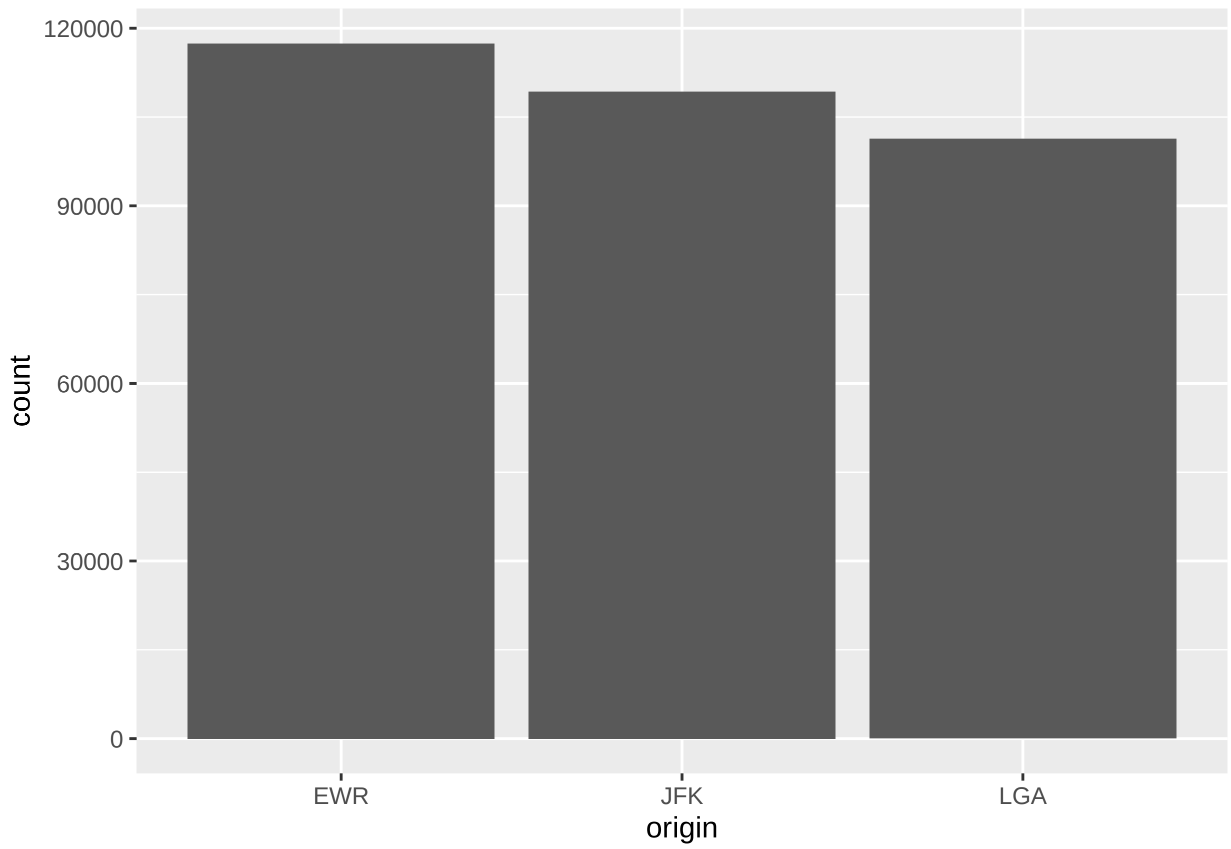
```
flights %>%
  filter(!is.na(dep_time), !is.na(arr_time)) %>%
  filter(origin == "JFK"| origin == "LGA" | origin == "EWR") %>%
  group_by(tailnum) %>% filter(n() >= 100)
```

```
## # A tibble: 222,870 x 19
## # Groups:   tailnum [1,210]
##     year month   day dep_time sched_dep_time dep_delay arr_time sched_arr_time
##    <int> <int> <int>    <int>          <int>     <dbl>    <int>          <int>
##  1  2013     1     1      517            515         2      830            819
##  2  2013     1     1      533            529         4      850            830
##  3  2013     1     1      544            545        -1     1004           1022
##  4  2013     1     1      554            558        -4      740            728
##  5  2013     1     1      555            600        -5      913            854
##  6  2013     1     1      557            600        -3      709            723
##  7  2013     1     1      557            600        -3      838            846
##  8  2013     1     1      558            600        -2      849            851
##  9  2013     1     1      558            600        -2      853            856
## 10  2013     1     1      558            600        -2      923            937
## # ... with 222,860 more rows, and 11 more variables: arr_delay <dbl>,
## #   carrier <chr>, flight <int>, tailnum <chr>, origin <chr>, dest <chr>,
## #   air_time <dbl>, distance <dbl>, hour <dbl>, minute <dbl>, time_hour <dttm>
```

Question 11 (25 points)

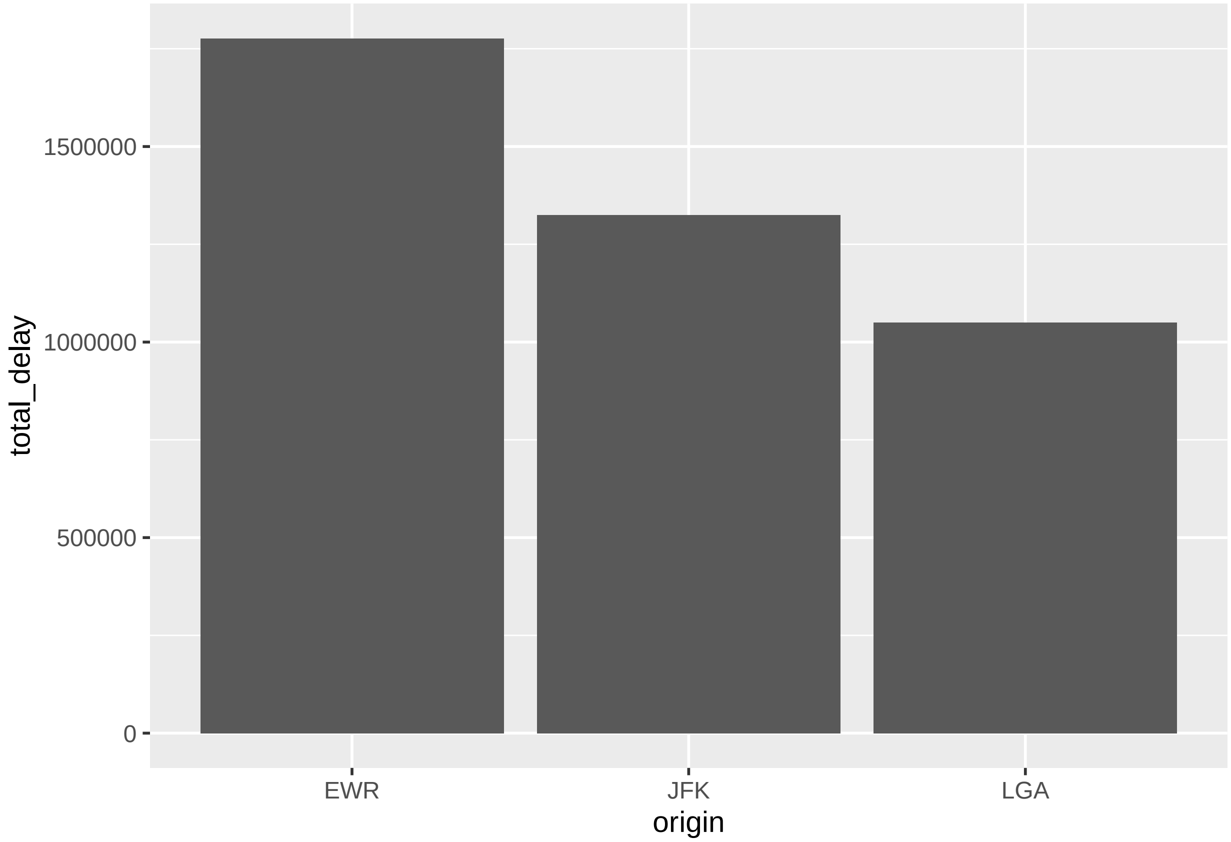
a. **(3 points) Make a bar chart showing how many flights in this data set departed from each of the three origin airports. (Hint: the easiest way to do this does not involve any data transformations)**

```
flights %>%
  filter(!is.na(dep_time), !is.na(arr_time)) %>%
  filter(origin == "JFK"| origin == "LGA"| origin == "EWR") %>%
  ggplot(aes(x = origin)) + geom_bar()
```



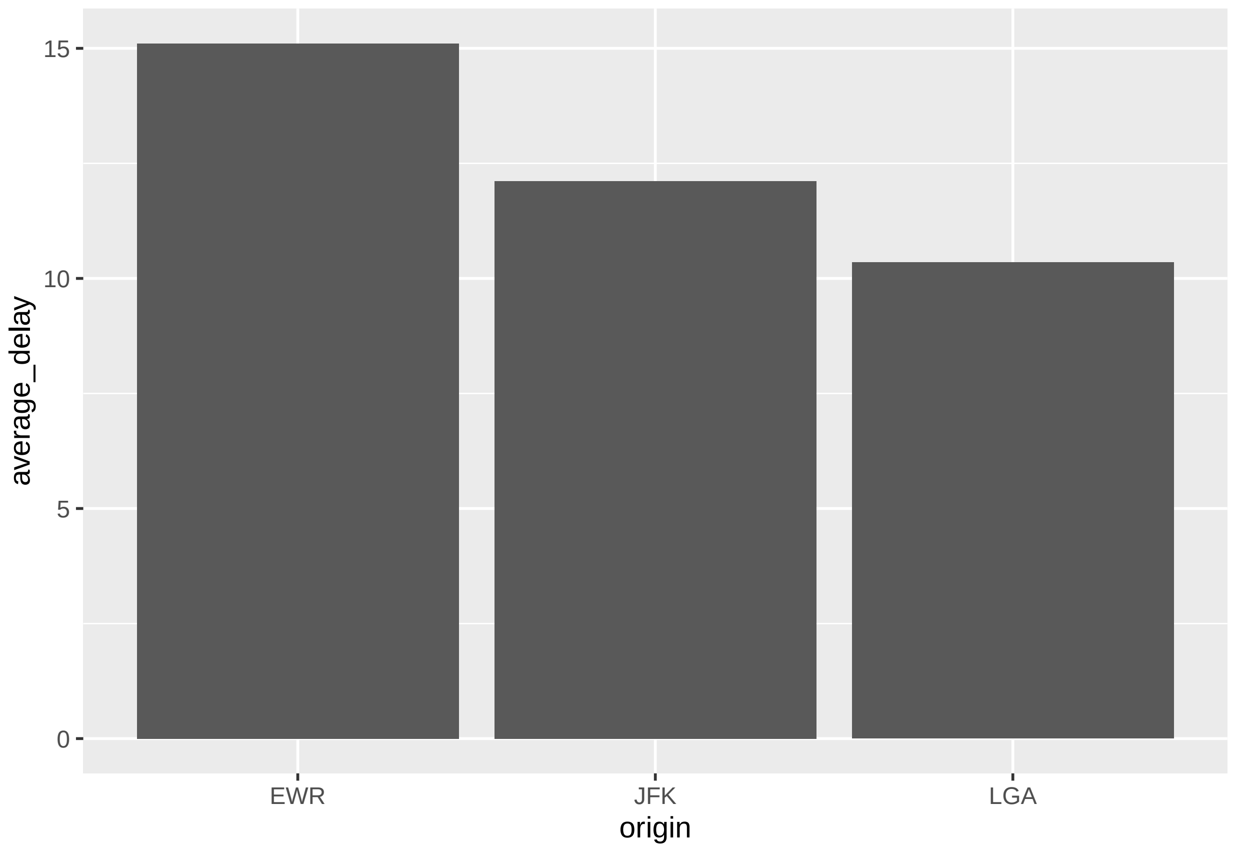
b. **(6 points) Make a bar chart showing how many total minutes of departure delay there were for each of the three origin airports.**

```
flights %>% filter(!is.na(dep_delay)) %>%
  filter(origin == "JFK"| origin == "LGA"| origin == "EWR") %>%
  group_by(origin)%>% summarize(total_delay = sum(dep_delay)) %>%
  ggplot(mapping = aes(x = origin, y = total_delay)) + geom_col()
```



c. **(8 points) For each origin airport, compute the average delay of all flights leaving that airport. Show these average delays in a bar chart. Be sure you are removing any NA values *before* computing your averages.**

```
flights %>%
  filter(!is.na(dep_delay)) %>%
  filter(origin == "JFK"| origin == "LGA"| origin == "EWR") %>%
  group_by(origin)%>% summarize(average_delay = mean(dep_delay)) %>%
  ggplot(mapping = aes(x = origin, y = average_delay)) + geom_col()
```



d. **(6 points) For each origin airport, compute the total number of flights, the number of flights that are cancelled, and the fraction of flights that are cancelled. A useful command is sum(is.na(dep_time)), which counts how many flights have a dep_time that is NA, that is, how many flights were cancelled. (You can visualize these in a bar chart if you want but it's not required)**

```
flights  %>%
  filter(origin == "JFK"| origin == "LGA"| origin == "EWR") %>%
  group_by(origin) %>%
  summarize(number_Cancelled_flights = sum(is.na(dep_time)),
            number_good_flights = sum(!is.na(dep_time)),
            number_flights = n(),
            Percent_cancelled = 100*(number_Cancelled_flights/number_flights))
```

```
## # A tibble: 3 x 5
##   origin number_Cancelled_fl~ number_good_fligh~ number_flights Percent_cancell~
##   <chr>                 <int>              <int>          <int>            <dbl>
## 1 EWR                    3239             117596         120835             2.68
## 2 JFK                    1863             109416         111279             1.67
## 3 LGA                    3153             101509         104662             3.01
```

e. **(2 points) Based on your answers to the previous parts, if you want to minimize your flight delays, which new york city area airport should you fly out of? If you want to minimize the chance that your flight is cancelled, what airport should you fly out of?**

To minimize delays one should use the laguardia airport, and to minimize likelihood of cancellation, one should use John f. Kennedy airport.