In [33]:
```
install.packages("nycflights13")
library(nycflights13)
options(repr.plot.width=5, repr.plot.height=4)
```

```
Installing package into ‘/usr/local/lib/R/site-library’
(as ‘lib’ is unspecified)
```

STATS 306

Homework 1: Plotting and data manipulation

- Each problem is worth one or two points for a total of 10.
- For each problem, enter the R code in the cell provided marked "YOUR SOLUTION HERE".

Problem 1: Recap of Lecture 1 (2 pts)

(a) Write the command to install the package `tidyverse` . For this problem, you can comment out the command using # so that you do not need to reinstall your `tidyverse` package. Then, load the `tidyverse` package into your current environment. *1/2 point*

In [34]:
```
# Your solution here
install.packages("tidyverse")
```

```
Installing package into '/usr/local/lib/R/site-library'
(as 'lib' is unspecified)
```

In [35]:
```
library(tidyverse)
```

(b) Write the command to get more information on the `trees` data set. What is the `trees` dataset about? Output the `trees` first 6 rows of the data set. *1 point*

In [36]:
```
# Your solution here
help(trees)
#As shown in the documentation, the trees data set provides measuremnts of the diameter,
#height, and volume of timber in 31 fellef black cherry trees.
```

In [37]:
```
head(trees)
```

A data.frame: 6 × 3

	Girth	Height	Volume
	<dbl>	<dbl>	<dbl>
1	8.3	70	10.3
2	8.6	65	10.3
3	8.8	63	10.2
4	10.5	72	16.4
5	10.7	81	18.8
6	10.8	83	19.7

(c) How many rows and columns does `trees` have? *1/2 point*

In [38]:
```
# Your solution here
nrow(trees)
#31 rows
```

31

In [39]:
```
ncol(trees)
#3 columns
```

3

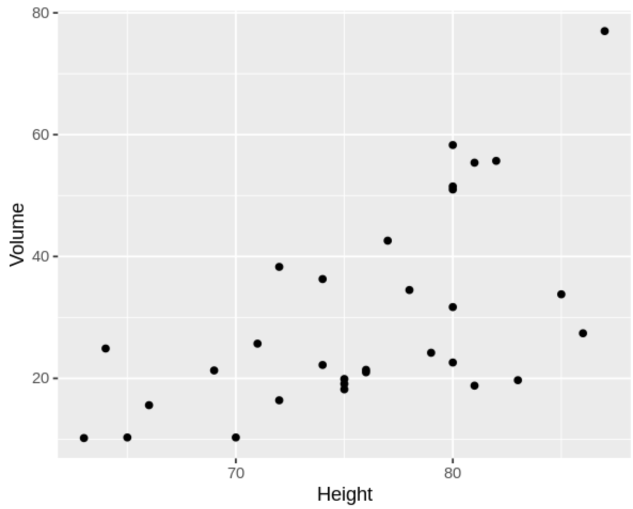
(d) To create a scatterplot using `Height` and `Volume` variables in the `trees` data frame, Professor Terhorst uses the following command

```
ggplot() +
    geom_point(mapping = (x = Height, y = Volume))
```

and R produces an error.

Fix Professor Terhorst's code and type it below to produce a scatterplot. Output your results. *1 point*

In [40]:
```
# Your solution here
ggplot(data = trees) +
    geom_point(mapping = aes(x = Height, y = Volume))
```



Problem 2: Animals (3 pts)

Problems 2 is based on the `animals` table, which is defined for you in the next cell.

In [41]:
```
library(MASS)
animals <- as_tibble(MASS::Animals) %>% mutate(species = rownames(MASS::Animals))
```

In [42]:
```
head(animals)
```

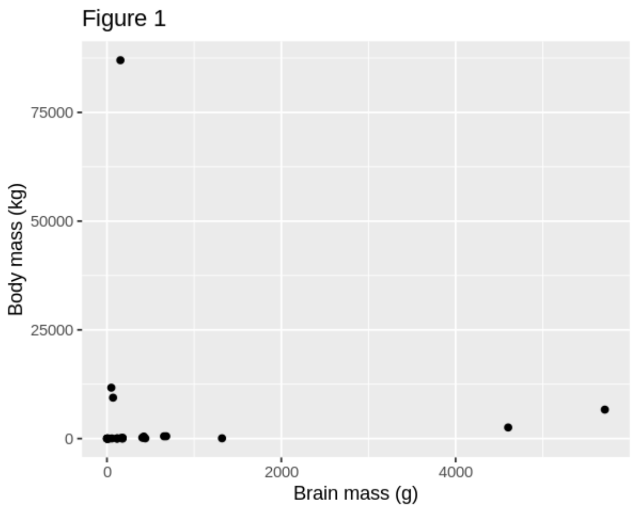
A tibble: 6 × 3

body	brain	species
<dbl>	<dbl>	<chr>
1.35	8.1	Mountain beaver
465.00	423.0	Cow
36.33	119.5	Grey wolf
27.66	115.0	Goat
1.04	5.5	Guinea pig
11700.00	50.0	Dipliodocus

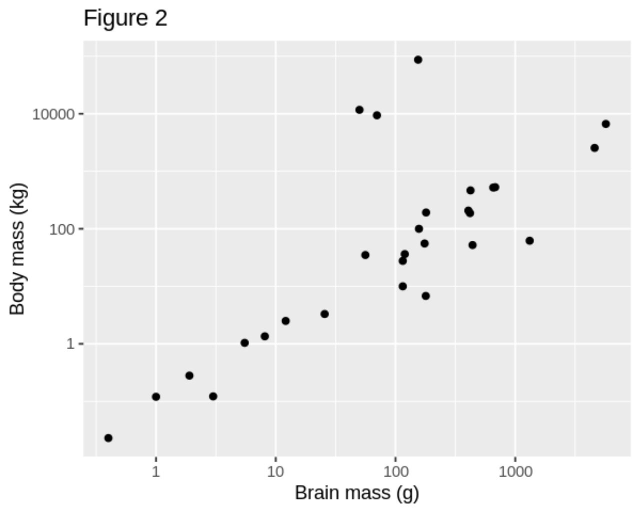
(a) How are brain mass and body mass related? Following the examples we saw in Lecture 1, reproduce the following two plots: *2 points*

In [43]:
```
# Your solution here
ggplot(data = animals) +
    geom_point(mapping = aes(x = brain, y = body)) +
    ggtitle("Figure 1") +
    xlab("Brain mass (g)") +
    ylab("Body mass (kg)")

#From the graph below, it is hard to see if there is any correlation between
# brain mass and body mass given the large range of x and y values on the graph.
```



In [44]:
```
ggplot(data = animals) +
    geom_point(mapping = aes(x = brain, y = body)) +
    ggtitle("Figure 2") +
    xlab("Brain mass (g)") +
    ylab("Body mass (kg)") +
    scale_y_continuous(trans='log10') +
    scale_x_continuous(trans='log10')
```



(b) The preceding plot indicates that the log of brain mass is linearly related to the log of body mass. Check this by computing the correlation of brain mass with body mass, and compare that with the correlation between the logs of those two quantities. (*Hint*: R contains a built-in function for computing the correlation of two vectors.) *1 point*

In [45]:
```
# Your solution here
cor(animals$brain, animals$body)
```

-0.00534116256125113

In [46]:
```
vec1 = log(animals$brain)
vec2 = log(animals$body)
cor(vec1, vec2)
```

0.779493496728511

In [47]:
```
#As we can see from the two calculates above, the linear correlation between brain mass and
#body mass is fairy weak negative correlation as it is nearly 0 (-0.005).
#However, the linear correlaiton between the log of brain mass and the log of body mass
#is a pretty strong positive correlation as it is closer to 1 (0.779)
```

Problem 3: Flights (3 pts)

This problem looks at the `flights` dataset, which was already loaded at the begginning of this notebook.

(a) The following command lists the top six destinations in this dataset:

In [48]:
```
dest_top6 <- count(flights, dest) %>% top_n(6) %>% print
```

```
Selecting by n
```

```
# A tibble: 6 × 2
  dest      n
  <chr> <int>
1 ATL   17215
2 BOS   15508
3 CLT   14064
4 LAX   16174
5 MCO   14082
6 ORD   17283
```

Use `filter()` to subset the flights table down to only those for the top six destinations. You should end up with a table that has 94,326 rows, the first ten of which look like: *1/2 point*

```
# A tibble: 94,326 × 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      554            600        -6      812            837
 2  2013     1     1      554            558        -4      740            728
 3  2013     1     1      557            600        -3      838            846
 4  2013     1     1      558            600        -2      753            745
 5  2013     1     1      558            600        -2      924            917
 6  2013     1     1      559            559         0      702            706
 7  2013     1     1      600            600         0      837            825
 8  2013     1     1      606            610        -4      837            845
 9  2013     1     1      608            600         8      807            735
10  2013     1     1      615            615         0      833            842
# … with 94,316 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>
```

In [49]:
```
# Your solution here
top6 <- c("ATL", "BOS", "CLT", "LAX", "MCO", "ORD")
filter(flights, dest %in% top6) %>% print
```

```
# A tibble: 94,326 × 19
    year month   day dep_time sched_de…¹ dep_d…² arr_t…³ sched…⁴ arr_d…⁵ carrier
   <int> <int> <int>    <int>      <int>   <dbl>   <int>   <int>   <dbl> <chr>
 1  2013     1     1      554        600      -6     812     837     -25 DL
 2  2013     1     1      554        558      -4     740     728      12 UA
 3  2013     1     1      557        600      -3     838     846      -8 B6
 4  2013     1     1      558        600      -2     753     745       8 AA
 5  2013     1     1      558        600      -2     924     917       7 UA
 6  2013     1     1      559        559       0     702     706      -4 B6
 7  2013     1     1      600        600       0     837     825      12 MQ
 8  2013     1     1      606        610      -4     837     845      -8 DL
 9  2013     1     1      608        600       8     807     735      32 MQ
10  2013     1     1      615        615       0     833     842      -9 DL
# … with 94,316 more rows, 9 more variables: flight <int>, tailnum <chr>,
#   origin <chr>, dest <chr>, air_time <dbl>, distance <dbl>, hour <dbl>,
#   minute <dbl>, time_hour <dttm>, and abbreviated variable names
#   ¹sched_dep_time, ²dep_delay, ³arr_time, ⁴sched_arr_time, ⁵arr_delay
```

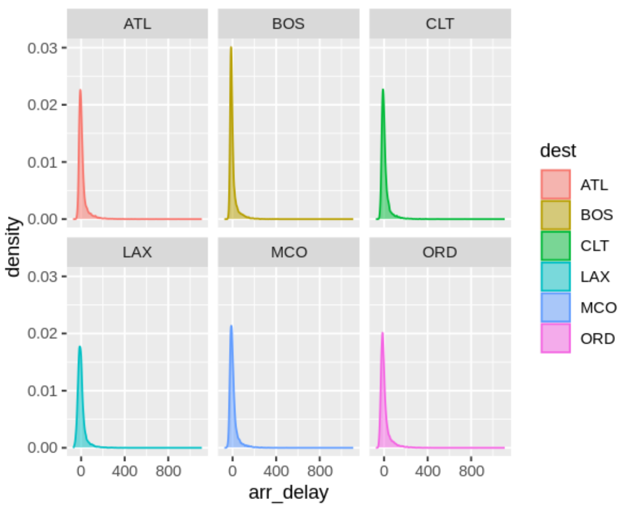
In [ ]:
```
newdf <- filter(flights, dest %in% top6)
```

(b) Using the table you created in part (a), recreate the following plot showing the density of arrival delays for the top six destinations: *2 points*

In [58]:
```
# Your solution here

ggplot(data = newdf, aes(arr_delay, color = dest, fill = dest)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~dest)
```

```
Warning message:
“Removed 2234 rows containing non-finite values (`stat_density()`).”
```



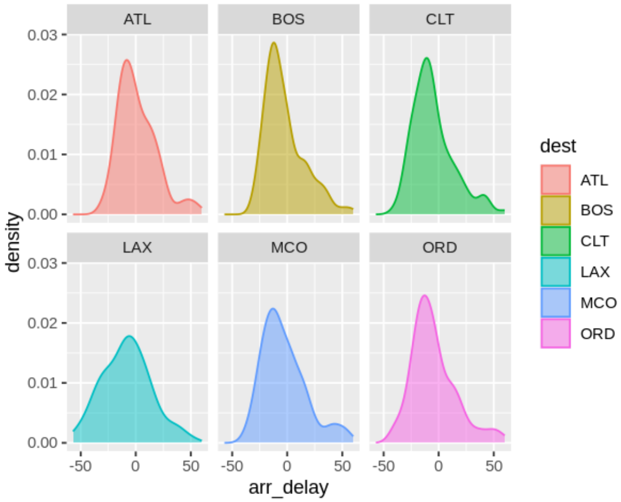
(c) Adjust the plot from part (b) to recreate the following plot, which only shows flights which had an arrival delay of $\pm$ 1 hour: *1/2 point*

In [73]:
```
help(flights)
```

In [77]:
```
# Your solution here
specific_delays <- filter(newdf, arr_delay == c(-60:60))
```

```
Warning message in arr_delay == c(-60:60):
“longer object length is not a multiple of shorter object length”
```

In [78]:
```
ggplot(data = specific_delays, aes(arr_delay, color = dest, fill = dest)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~dest) +
  scale_x_continuous(breaks=c(-50,0,50))
```



Problem 4: Challenge Problem (2 pts)

Each problem set will feature one or two questions that go a bit beyond what we have covered in lab and lecture. The goal of these is for you learn how to use online resources (R's help, Google, Stack Overflow, etc.) to solve programming challenges that you have not encountered before. This is an important skill which you will use constantly as data scientists in the real world.

Load the `diamonds` data set, and use it to reproduce the following box-and-whisker plot which shows the distribution of diamond price as a function of the quality of its `cut` .

In [92]:
```
# Your solution here
head(diamonds)
```

A tibble: 6 × 10

carat	cut	color	clarity	depth	table	price	x	y	z
<dbl>	<ord>	<ord>	<ord>	<dbl>	<dbl>	<int>	<dbl>	<dbl>	<dbl>
0.23	Ideal	E	SI2	61.5	55	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58	335	4.34	4.35	2.75
0.24	Very Good	J	VVS2	62.8	57	336	3.94	3.96	2.48

In [106]:
```
ggplot(diamonds, aes(x=cut, y=price)) +
  geom_boxplot(fill="steelblue") +
  coord_flip()+
  xlab("Quality of Cut")+
  ylab("Price in USD")+
  scale_x_discrete(limits=c("Ideal","Very Good","Good", "Premium", "Fair"))
```