Homework 5

Zahlen Zbinden

```
library(tidyverse)
```

Tasks that require an answer are bolded (inside ** in the .Rmd file). For any task that includes a question (i.e. it ends with "?"), you should also answer the question in sentence form.

Vectors and Vector Functions

1.

(1 pt)

The following three chunks are attempts to create vectors, but each one has a problem. Either the vector created is of the wrong type, or there is a syntax error. **Identify the problem, then fix the code in each chunk.**

This should be a logical vector of length 5:

```
typeof(as.logical(c("TRUE", "FALSE", "TRUE", "TRUE", "FALSE")))
```

```
[1] "logical"
```

```
typeof(c(T, F, T, T, F))
```

```
[1] "logical"
```

This should be a character vector of length 4:

```
c("potato", "carrot", "eggplant", "lettuce")
```

```
[1] "potato"   "carrot"   "eggplant" "lettuce"
```

This should be a double vector of length 4:

```
c(1.1, 6.4, 1.5, 0.9)
```

```
[1] 1.1 6.4 1.5 0.9
```

2.

(2 pts)

Consider the vector `x`:

```
x <- c("10", "100%", "$1000")
```

What type of vector is x?

x is a character vector

```
typeof(x)
```

```
[1] "character"
```

I mentioned the dangers of coercion with `as.numeric()`. readr, a tidyverse package, provides the function `parse_number()`. **Apply both `as.numeric()` and `parse_number()` to `x`, then in your own words describe the difference in their behaviour.**

as.numeric can take a string that has no character and turn them into type numeric. parse_number() is more sophisticated and finds characters in the objects of the vector and removes them to find the number.

```
as.numeric(x)
```

```
Warning: NAs introduced by coercion
```

```
[1] 10 NA NA
```

```
parse_number(x)
```

```
[1]   10  100 1000
```

3.

(1 pt)

Consider the following code and output:

```
x <- c(1, 2, 3, 4)
y <- c(TRUE, FALSE)
x * y
```

```
[1] 1 0 3 0
```

In your own words, describe how R arrives at the output. x is of length 4, y is length 2. R is taking the short vector and repeating it till it gets the same length or longer than the first vector in the operation requested. Then takes as many elements as the first vector from the second vector and does the element math.

4.

(2 pts)

Consider the `starwars` dataset from dplyr.

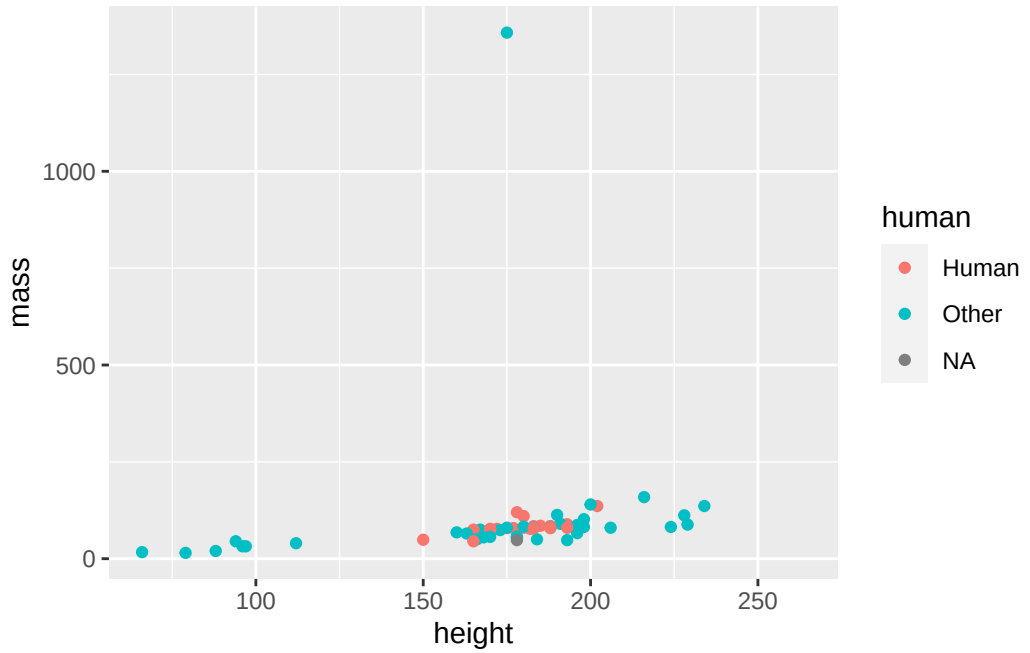
Add a column called `human` to `starwars` that takes the value `"Human"` if `species` is `"Human"` and `"Other"` otherwise.

```
starwars <- starwars %>%
  mutate(human = ifelse(species == "Human", "Human", "Other"))
```

Create a scatterplot of `height` versus `mass` with points colored by your new `human` column.

```
ggplot(starwars, aes(x = height, y = mass, color = human)) +
  geom_point()
```

```
Warning: Removed 28 rows containing missing values (`geom_point()`).
```



5.

(2 pts)

How many characters in `starwars` have more than one skin color?

Complete the following steps to answer the question.

One strategy to look for multiple skin colors, is to look to see if the value for `skin_color` contains a comma. E.g.

```
example_skin <- c("fair", "gold", "white, blue")
str_detect(example_skin, ",")
```

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

Create a new column in `starwars` called `many_cols` that contains `TRUE` if the characters `skin_color` contains a comma and `FALSE` otherwise.

```
starwars <- starwars %>%
  mutate(many_cols = str_detect(skin_color, ","))
```

```r
starwars
```

```
# A tibble: 87 x 16
   name     height  mass hair_color skin_color eye_color birth_year sex   gender
   <chr>     <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
 1 Luke Sk~    172    77 blond      fair       blue            19   male  mascu~
 2 C-3PO       167    75 <NA>       gold       yellow         112   none  mascu~
 3 R2-D2        96    32 <NA>       white, bl~ red             33   none  mascu~
 4 Darth V~    202   136 none       white      yellow          41.9 male  mascu~
 5 Leia Or~    150    49 brown      light      brown           19   fema~ femin~
 6 Owen La~    178   120 brown, gr~ light      blue            52   male  mascu~
 7 Beru Wh~    165    75 brown      light      blue            47   fema~ femin~
 8 R5-D4        97    32 <NA>       white, red red             NA   none  mascu~
 9 Biggs D~    183    84 black      light      brown           24   male  mascu~
10 Obi-Wan~    182    77 auburn, w~ fair       blue-gray       57   male  mascu~
# i 77 more rows
# i 7 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>, human <chr>, many_cols <lgl>
```

Filter starwars using the column many_cols.

```r
starwars %>% filter(many_cols == F)
```

```
# A tibble: 73 x 16
   name     height  mass hair_color skin_color eye_color birth_year sex   gender
   <chr>     <int> <dbl> <chr>      <chr>      <chr>          <dbl> <chr> <chr>
 1 Luke Sk~    172    77 blond      fair       blue            19   male  mascu~
 2 C-3PO       167    75 <NA>       gold       yellow         112   none  mascu~
 3 Darth V~    202   136 none       white      yellow          41.9 male  mascu~
 4 Leia Or~    150    49 brown      light      brown           19   fema~ femin~
 5 Owen La~    178   120 brown, gr~ light      blue            52   male  mascu~
 6 Beru Wh~    165    75 brown      light      blue            47   fema~ femin~
 7 Biggs D~    183    84 black      light      brown           24   male  mascu~
 8 Obi-Wan~    182    77 auburn, w~ fair       blue-gray       57   male  mascu~
 9 Anakin ~    188    84 blond      fair       blue            41.9 male  mascu~
10 Wilhuff~    180    NA auburn, g~ fair       blue            64   male  mascu~
# i 63 more rows
# i 7 more variables: homeworld <chr>, species <chr>, films <list>,
#   vehicles <list>, starships <list>, human <chr>, many_cols <lgl>
```

Using the result from above, answer the question, how many characters in `starwars` have more than one skin color?

```
nrow(starwars %>% filter(many_cols == F))
```

```
[1] 73
```

6.

Here's a small example of taking a vector that contains years and converting it to a character vector representing decades:

```
year <- c(1900, 1901, 1909, 1910, 1921, 1931, 2001)
floor(year / 10) |> paste0("0's")
```

```
[1] "1900's" "1900's" "1900's" "1910's" "1920's" "1930's" "2000's"
```

Which functions in the second line of code are vector functions?

Here's some randomly generated data relating to prices over time:

```
set.seed(2484) # so you all get the same "random" data
prices <- tibble(
  year = 1900:1950,
  price = rnorm(n = length(year), mean = year/10)
)
prices
```

```
# A tibble: 51 x 2
    year price
   <int> <dbl>
 1  1900  191.
 2  1901  190.
 3  1902  190.
 4  1903  191.
 5  1904  191.
 6  1905  190.
 7  1906  192.
 8  1907  187.
 9  1908  191.
```

```
10  1909  192.
# i 41 more rows
```

Add a column `decade` that is a character string representing the decade corresponding the `year`

```
prices <- prices %>%
  mutate(decade = floor(year / 10) %>% paste0("0's"))
```

Use your new `decade` column to produce a summary with the mean price per decade.

```
prices %>%
  group_by(decade) %>%
  summarise(mean(price))
```

```
# A tibble: 6 x 2
  decade `mean(price)`
  <chr>          <dbl>
1 1900's          190.
2 1910's          191.
3 1920's          192.
4 1930's          193.
5 1940's          195.
6 1950's          196.
```