

Final Exam

Name:

For each question, change the code to produce the **correct** output. Each question contains only one mistake, but which may be repeated (e.g., using the wrong function multiple times)

The following libraries are loaded in the global environment when running each snippet of code:

```
library(tidyverse)
library(fivethirtyeight)
```

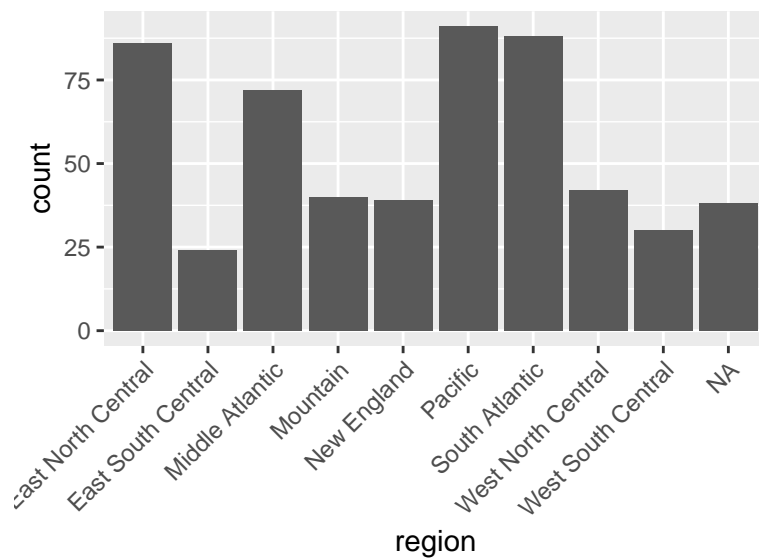
Data Visualization

1.

```
ggplot(data = steak_survey) +  
  geom_col(aes(x = region)) +  
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

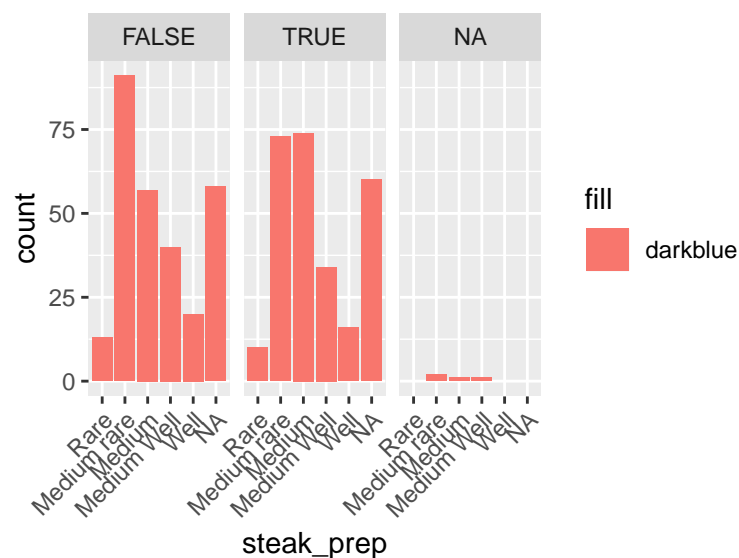
```
## Error in pmin(y, 0): object 'y' not found
```

correct:

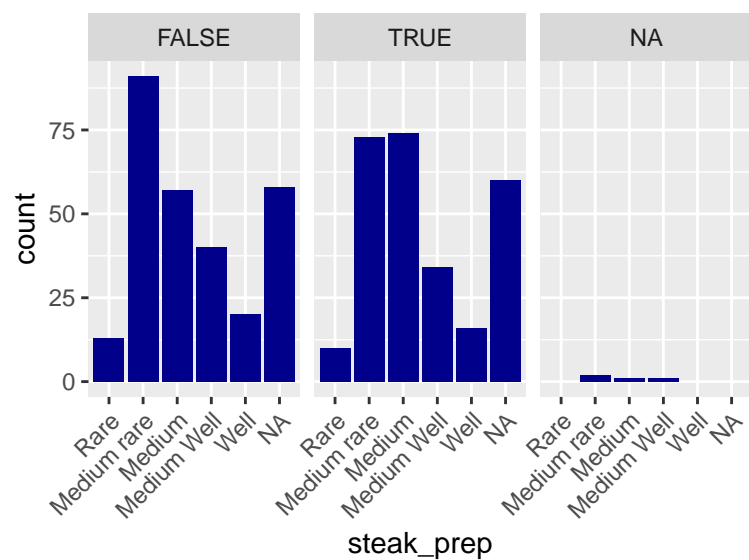


- Count of subjects in each level of steak preparation preference facetting by whether they played the lottery. Change the fill color of the bars to darkblue.

```
ggplot(data = steak_survey) +
  geom_bar(aes(x = steak_prep, fill = 'darkblue')) +
  facet_wrap(~ lottery_a) +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



correct:

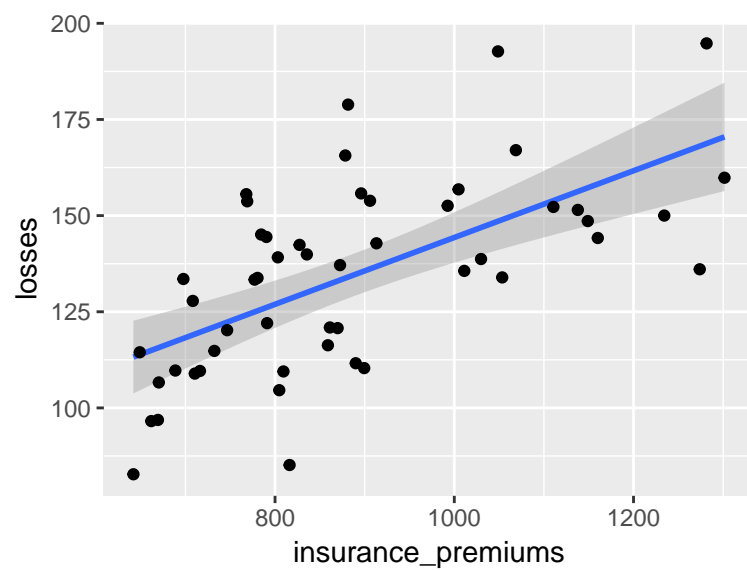


3. Chart of the correlation between `insurance_premiums` and `losses`

```
ggplot(bad_drivers) +  
  geom_smooth(aes(insurance_premiums, losses), method = 'lm') +  
  geom_point()
```

Error: geom_point requires the following missing aesthetics: x, y

correct:



Data Analysis

4. Total graduates by major_category (first 5 rows)

```
college_recent_grads %>%  
  group_by(major_category) %>%  
  summarise(Total = sum(total)) %>%  
  slice(1:5)
```

```
## # A tibble: 5 x 2  
##   major_category      Total  
##   <chr>            <int>  
## 1 Agriculture & Natural Resources    NA  
## 2 Arts                      357130  
## 3 Biology & Life Science    453862  
## 4 Business                 1302376  
## 5 Communications & Journalism    392601
```

correct:

```
## # A tibble: 5 x 2  
##   major_category      Total  
##   <chr>            <int>  
## 1 Agriculture & Natural Resources    75620  
## 2 Arts                      357130  
## 3 Biology & Life Science    453862  
## 4 Business                 1302376  
## 5 Communications & Journalism    392601
```

5. Total share of women by major_category (first 5 rows)

```
college_recent_grads %>%  
  group_by(major_category) %>%  
  summarise(percent = n(women)/n(total)) %>%  
  slice(1:5)
```

```
## Error in summarise_impl(.data, dots): `n()` does not take arguments
```

correct:

```
## # A tibble: 5 x 2  
##   major_category      percent  
##   <chr>            <dbl>  
## 1 Agriculture & Natural Resources    NA  
## 2 Arts                      0.624  
## 3 Biology & Life Science    0.593  
## 4 Business                 0.487  
## 5 Communications & Journalism    0.664
```

6. Extract the row for the major with the highest share of females

```
college_recent_grads %>%  
  filter(max(sharewomen, na.rm = T)) %>%  
  select(major_category:sharewomen)
```

```
## Error in filter_impl(.data, quo): Argument 2 filter condition does not evaluate to a logical vector
```

correct:

```
## # A tibble: 1 x 6  
##   major_category total sample_size   men women sharewomen  
##   <chr>          <int>      <int> <int> <int>      <dbl>  
## 1 Education      37589         342  1167 36422      0.969
```

7. Count of major_category for Engineering and Business only.

```
college_recent_grads %>%  
  filter(major_category == 'Engineering', major_category == 'Business') %>%  
  count(major_category)
```

```
## # A tibble: 0 x 2  
## # ... with 2 variables: major_category <chr>, n <int>
```

correct:

```
## # A tibble: 2 x 2  
##   major_category     n  
##   <chr>          <int>  
## 1 Business         13  
## 2 Engineering      29
```

8. Transform `shareswomen` in percent. Print only the first row and keep only column “sharewoman”

```
college_recent_grads %>%  
  slice(1) %>%  
  select(sharewomen) %>%  
  filter(sharewomen = scales::percent(sharewomen))
```

```
## Error: `sharewomen` (`sharewomen = scales::percent(sharewomen)`) must not be named, do you need `==`?
```

correct:

```
## # A tibble: 1 x 1  
##   sharewomen  
##   <chr>  
## 1 12.1%
```

9. Output a vector of means

```
map_dbl(mean, mtcars)
```

```
## Error: Can't convert a `data.frame` object to function
```

correct:

```
##      mpg      cyl      disp      hp      drat      wt  
## 20.090625  6.187500 230.721875 146.687500  3.596563  3.217250  
##      qsec      vs      am      gear      carb  
## 17.848750  0.437500  0.406250  3.687500  2.812500
```

Functions

10. Calculate the mean for each column of a dataframe containing numeric values:

```
columnwiseMean <- function(df) {  
  if (is.data.frame(df)) stop("`df` is not a dataframe")  
  map(df, mean)  
}  
columnwiseMean(cars)
```

```
## Error in columnwiseMean(cars): `df` is not a dataframe
```

correct:

```
## $speed  
## [1] 15.4  
##  
## $dist  
## [1] 42.98
```

11. Calculate a percentage given a total amount and a fraction of the total observations

```
prc <- function(total, obs, call. = FALSE) {  
  if (!is.double(total) | !is.double(obs)) {  
    stop("`total` and `obs` must be of type numeric")  
  }  
  (obs / total) * 100  
  x <- paste0(x, '%')  
  x  
}  
prc(400, 20)
```

```
## Error in paste0(x, "%"): object 'x' not found
```

correct:

```
prc(400, 20)
```

```
## [1] "5%"
```


#General R knowledge (TRUE/FALSE)

- == performs an assignment operation : F
- a NA is equivalent to 0 : F
- In dataframes, each column must have the same number of rows/cells: T
- The code snippets below retrieve the same output: T (vectorization)

```
c(1, 4, 3) + 1 # 1.
```

```
c(1, 4, 3) + c(1, 1, 1) # 2.
```

- The code snippets below retrieve the same output when x is the same: F

```
mean(is.na(x)) # 1.
```

```
x %>% mean() %>% is.na(x) # 2.
```

- The code snippets below retrieve the same output when x is the same: T

```
bad_drivers %>% filter(num_drivers > x) # 1.
```

```
bad_drivers %>% filter(!(num_drivers <= x)) # 2.
```

- The code snippets below retrieve the same output: F

```
steak_survey %>% count(lottery_a, smoke) # 1.
```

```
steak_survey %>% group_by(lottery_a, smoke) %>%  
  summarise(sum()) # 2.
```

- filter is for manipulating rows, while slice is for manipulating variables: F
- the output from summarise() has as many rows as the grouping levels: T
- datasets being plotted in ggplot() should be “tidy” (or in longitudinal form): T
- in ggplot, we can map multiple variables to the same aesthetic: F
- every time we query a database using dplyr the output is automatically collected locally on our machine: F