hw5

Problem 1

The CO2 data frame is comes with a standard R installation, and describes CO2 uptake (uptake column) for plants from different areas (Type column describing where they came from, Quebec or Mississippi) in different conditions (Treatment column). (See help(CO2) for more info.)

```
print(head(CO2))
```

```
##   Plant   Type  Treatment conc uptake
## 1   Qn1 Quebec nonchilled   95   16.0
## 2   Qn1 Quebec nonchilled  175   30.4
## 3   Qn1 Quebec nonchilled  250   34.8
## 4   Qn1 Quebec nonchilled  350   37.2
## 5   Qn1 Quebec nonchilled  500   35.3
## 6   Qn1 Quebec nonchilled  675   39.2
```

Write a function called chilled_vs_nonchilled that takes such a dataframe (or a subset of it), that will compare uptake values in the nonchilled treatment to uptake values in the chilled treatment using a t.test(). It should return a single-row dataframe with five columns, pval_treat_uptake, mean_nonchilled and mean_chilled, n_chilled and n_nonchilled. (Hint: if you extract the uptake entries where Treatment is chilled as a vector, then length() can be used to get the number of entries in that vector. Alternatively, if you select a subset of the whole dataframe (as a dataframe) where Treatment is chilled, nrow() can return the number of rows in that subset).

```
# given a dataframe with columns a column for 'Treatment' (with values of 'chilled'
# and 'nonchilled') and a column for 'uptake' (numeric), runs a t.test on uptake values
# for the two treatments, and returns a single-row data frame.
chilled_vs_nonchilled <- function(sub_df) {
  # your code here!
}



## uncomment to test:
# print(chilled_vs_nonchilled(CO2))
```

The test output from running the test on the entire dataset above should be

```
  pval_treat_uptake mean_nonchilled mean_chilled n_chilled n_nonchilled
1       0.003106937        30.64286     23.78333        42           42
```

Problem 2

Use group_by() and do() from the dplyr package to run the test for plants from different areas (Quebec and Mississippi) and print the result.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
# using %>% not required
# your code here
```

Here's what I got:

```
# A tibble: 2 x 6
# Groups:   Type [2]
  Type        pval_treat_uptake mean_nonchilled mean_chilled n_chilled n_nonchilled
  <fct>                   <dbl>           <dbl>        <dbl>     <int>        <int>
1 Quebec             0.235                 35.3         31.8        21           21
2 Mississippi        0.00000506            26.0         15.8        21           21
```

Problem 3

This experiment was also run by placing the plants in different ambient concentrations of Co2 (`conc` column), and we hypothesize that this may affect the results. Do the test again, but this type grouping by both `Type` and `conc`; the result should have 14 rows. Use `p.adjust()` with `method = "BY"` to add a column called `pval_adj_BY` of adjusted p-values.

```
library(dplyr)
# using %>% not required
# your code here!

# uncomment to test
# print(head(result))
```

The first six rows as printed above should be:

```
# A tibble: 6 x 8
# Groups:   Type, conc [6]
  Type    conc pval_treat_uptake mean_nonchilled mean_chilled n_chilled n_nonchilled result_adj_BY
  <fct>  <dbl>             <dbl>           <dbl>        <dbl>     <int>        <int>         <dbl>
1 Quebec    95            0.318             15.3         12.9         3            3         1
2 Quebec   175            0.0683            30.0         24.1         3            3         0.518
3 Quebec   250            0.356             37.4         34.5         3            3         1
4 Quebec   350            0.106             40.4         35.8         3            3         0.534
5 Quebec   500            0.393             39.6         36.7         3            3         1
6 Quebec   675            0.0938            41.5         37.5         3            3         0.534
```

Problem 4

How many entries are there where `result_adj_BY` is less than 0.2? What can we infer about these tests?

Your answer here.