HW#3

Eddie Perez

2/19/2022

```
library(tidyverse)
library(stringr)
```

#1. Using the 173 majors listed in fivethirtyeight.com's College Majors dataset [https://fivethirtyeight.com/features/the-economic-guide-to-picking-a-college-major/], provide code that identifies the majors that contain either "DATA" or "STATISTICS"

Loading the Data

```
file <- "https://raw.githubusercontent.com/fivethirtyeight/data/master/college-majors/majors-list.csv"
major_list <- read.csv(file, TRUE, ",")
head (major_list)
```

```
##   FOD1P                                Major                Major_Category
## 1  1100                   GENERAL AGRICULTURE Agriculture & Natural Resources
## 2  1101 AGRICULTURE PRODUCTION AND MANAGEMENT Agriculture & Natural Resources
## 3  1102                AGRICULTURAL ECONOMICS Agriculture & Natural Resources
## 4  1103                      ANIMAL SCIENCES Agriculture & Natural Resources
## 5  1104                         FOOD SCIENCE Agriculture & Natural Resources
## 6  1105            PLANT SCIENCE AND AGRONOMY Agriculture & Natural Resources
```

Searching for Data and Statistics Majors

searching and storing majors from Data and Statistics

```
Data_Majors <- str_subset(major_list$Major,"DATA")
Statistic_Majors <- str_subset(major_list$Major,"STATISTICS")

print("Below you will find the Data major\n")
```

```
## [1] "Below you will find the Data major\n"
```

```
Data_Majors
```

```
## [1] "COMPUTER PROGRAMMING AND DATA PROCESSING"
```

```
print("\nBelow you will find the Statistic major\n")
```

```
## [1] "\nBelow you will find the Statistic major\n"
```

```
Statistic_Majors
```

```
## [1] "MANAGEMENT INFORMATION SYSTEMS AND STATISTICS"
## [2] "STATISTICS AND DECISION SCIENCE"
```

#2 Write code that transforms the data below:

[1] "bell pepper" "bilberry" "blackberry" "blood orange" [5] "blueberry" "cantaloupe" "chili pepper" "cloudberry"
[9] "elderberry" "lime" "lychee" "mulberry"
[13] "olive" "salal berry" Into a format like this: c("bell pepper", "bilberry", "blackberry", "blood orange", "blueberry", "cantaloupe", "chili pepper", "cloudberry", "elderberry", "lime", "lychee", "mulberry", "olive", "salal berry") The two exercises below are taken from R for Data Science, 14.3.5.1 in the on-line version:

```
fruits <- '[1] "bell pepper"  "bilberry"     "blackberry"   "blood orange"
[5] "blueberry"    "cantaloupe"   "chili pepper" "cloudberry"
[9] "elderberry"   "lime"         "lychee"       "mulberry"
[13] "olive"        "salal berry"'

fruits <- str_extract_all(fruits,'[a-z]+\\s[a-z]+|[a-z]+')
unlist(fruits)
```

```
##  [1] "bell pepper"  "bilberry"     "blackberry"   "blood orange" "blueberry"
##  [6] "cantaloupe"   "chili pepper" "cloudberry"   "elderberry"   "lime"
## [11] "lychee"       "mulberry"     "olive"        "salal berry"
```

#3 Describe, in words, what these expressions will match:

(.)\1\1 **This will look for the first Character that doesn't start on a new line and see if it repeats twice thereafter**

"(.)(.)\2\1" **It will look at the first two letters of a word and see if something matches the inverse**

(..)\1 **This would match character grouping for example a set of words that repeat, for example church the ch would be a good example**

"(.).\1.\1" **a specific letter repeated once then another letter repeated once**

"(.)(.)(.).*\3\2\1" **This would match any 3 characters pair repeated 3times**

#4 Construct regular expressions to match words that:

Start and end with the same character. Contain a repeated pair of letters (e.g. "church" contains "ch" repeated twice.) Contain one letter repeated in at least three places (e.g. "eleven" contains three "e"s.)

```
# Setting up my data for following Exercise

x <- c("eleven","banana","Mississippi","hello","today","yesterday","mom","dad")
```

Start and end with the same character.

```
str_view(x, "^(.)(.*)\\1$")
```

eleven

banana

Mississippi

hello

today

yesterday

mom

dad

Contain a repeated pair of letters (e.g. "church" contains "ch" repeated twice.)

```
str_view(x, "([A-Za-z][A-Za-z]).*\\1")
```

eleven

banana

Mississippi

hello

today

yesterday

mom

dad

Contain one letter repeated in at least three places (e.g. "eleven" contains three "e"s.)

```
str_view(x, "([A-Za-z]).*\\1.*\\1")
```

eleven

banana

Mississippi

hello

today

yesterday

mom

dad