Question 1

```
library(tidyverse)
```

It is an chance for you to practice your skills on regular expressions.

a. Consider the `phones.txt` files, extract all the telephone/fax numbers and format them as `(xxx)-xxx-xxxx`. (Hint: there should be 24 of them.)

```
# File to search for the 24 phone numbers
txt <- read_file("phones.txt")

# Pattern to match the phone numbers
phone <- "\\(?([2-9][0-9]{2})\\)?.([0-9]{3}).([0-9]{4})"

# Replace all instances of phone numbers to fit our desired pattern
str_replace_all(txt, phone, "(\\1)-\\2-\\3") %>%
# Extract all phone numbers matching our pattern
str_extract_all(phone)
```

```
## [[1]]
##  [1] "(734)-647-7087" "(734)-647-1848" "(734)-764-9586" "(734)-764-4123"
##  [5] "(734)-763-9272" "(734)-936-9358" "(734)-647-9069" "(734)-763-6577"
##  [9] "(734)-763-6276" "(734)-936-2598" "(800)-862-7284" "(800)-537-7284"
## [13] "(734)-936-9355" "(734)-764-3883" "(734)-763-5611" "(734)-764-0351"
## [17] "(734)-763-4765" "(734)-763-8046" "(734)-764-0400" "(734)-763-3257"
## [21] "(734)-763-7543" "(734)-764-7453" "(734)-936-4000" "(734)-936-4000"
```

b. Given the two vectors,

```
strings <- c(
    "http://www.foufos.gr",
    "https://www.foufos.gr",
    "http://foufos.gr",
    "http://www.foufos.gr/kino",
    "http://werer.gr",
    "www.foufos.gr",
    "www.mp3.com",
    "www.t.co",
    "http://t.co",
    "http://www.t.co",
    "https://www.t.co",
    "www.aa.com",
    "http://aa.com",
    "http://www.aa.com",
    "https://www.aa.com",
    "https://www.aa.com"
)
strings2 <- c(
    "www.foufos",
    "www.foufos-.gr",
    "www.-foufos.gr",
    "foufos.gr",
    "http://www.foufos",
    "http://foufos",
    "www.mp3#.com"
)
```

Write a regular expression `regex` that will match all of `strings` and none of the `strings2`.

```
# Our regular expression looks a little complicated so let's break it down
# All instances in strings begin with either http[s] or www so we need to this at least
 one hence the +
# Our https[s] or www will be followed by either a [.:]?
# If it's an "http[s]:" it will be followed by two // hence (\\/{2})?
# Then the url may be followed by a www. so we use (www.)?
# ([a-zA-Z0-9@:%._\\+~=]{1,256}\\.) looks for between 1-256 characters and will end the
 string
# We account for the url endings with (gr|com|co)
# and the special case of "/kino" using (/kino)?
regex <- "(http[s]?|www)+[.:]?(\\/{2})?(www.)?([a-zA-Z0-9@:%._\\+~=]{1,256}\\.)(gr|com|c
o)(/kino)?"
all(str_detect(strings, regex))
```

```
## [1] TRUE
```

```
all(str_detect(strings2, regex, negate = TRUE))
```

```
## [1] TRUE
```

c. Write a regular expression that matches the answers to the question

What are the colours of the French flag (in any order)?

Your regex should be specific, for example, it should not match “green, white, red” and “blue, white or red”.

It is optional to handle repetitions such as “blue, blue, blue” or “blue, white and blue”.

```r
answers <- c(
   "blue, white, red",
   "blue, white and red",
   "blue, red, white",
   "blue, red and white",
   "white, red, blue",
   "white, red and blue",
   "white, blue, red",
   "white, blue and red",
   "red, blue, white",
   "red, blue and white",
   "red, white, blue",
   "red, white and blue"
)
# Our regex pattern will find the colors in any order with either commas or "and"
# Does not account for repitition.
regex <- "(blue|white|red), (blue|white|red)[, ]{1}(and)? (blue|white|red)"
str_detect(answers, regex)
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

q2.Rmd

```
library(tidyverse)
library(tidytext)
library(proxy)
```

```
simpsons <- read_csv("simpsons_dataset.csv")
simpsons %>% glimpse()
```

```
## Rows: 158,314
## Columns: 2
## $ role        <chr> "Miss Hoover", "Lisa Simpson", "Miss Hoover", "Lisa Simp…
## $ spoken_words <chr> "No, actually, it was a little of both. Sometimes when a…
```

This dataset contains a number of scripts played by different Simpson characters. We hope to learn some similarity and differences between different characters.

(a) Discard the characters which have less than 50 scripts (rows). Use the same dataset for the following questions.

```
# Create a filtered dataframe for roles wtih 50 or more scripts (rows)
filtered_df <- simpsons %>%
  group_by(role) %>%
  count() %>%
  filter(n >= 50)
# Filter the original dataframe based on if the "role" is in our filtered dataframe
# and drop_na()
common_chars <- simpsons %>%
                filter(role %in% filtered_df$role) %>%
                drop_na()
common_chars
```

```
## # A tibble: 107,059 x 2
##    role                spoken_words
##    <chr>               <chr>
##  1 Miss Hoover         No, actually, it was a little of both. Sometimes when a d…
##  2 Lisa Simpson        Where's Mr. Bergstrom?
##  3 Miss Hoover         I don't know. Although I'd sure like to talk to him. He d…
##  4 Lisa Simpson        That life is worth living.
##  5 Edna Krabappel-Fl… The polls will be open from now until the end of recess. …
##  6 Martin Prince       I don't think there's anything left to say.
##  7 Edna Krabappel-Fl… Bart?
##  8 Bart Simpson        Victory party under the slide!
##  9 Lisa Simpson        Mr. Bergstrom! Mr. Bergstrom!
## 10 Lisa Simpson        Do you know where I could find him?
## # … with 107,049 more rows
```

(b) What's Homer Simpson most spoken non stop word?

```
# Filter down to "Homer Simpson"
# split the spoken_words into tokens
# Using anti_join to remove stop words
# Use group_by and count() to get a count of individual words
# Sort by `n` and display the first row
common_chars %>%
  filter(role == "Homer Simpson") %>%
  unnest_tokens(word, spoken_words) %>%
  anti_join(stop_words) %>%
  group_by(word) %>%
  count() %>%
  arrange(-n) %>%
  head(1)
```

```
## # A tibble: 1 x 2
## # Groups:   word [1]
##   word      n
##   <chr> <int>
## 1 marge  1796
```

(c) By using bigrams, what is the most spoken two-word pharse spoken by Homer Simpson?

Hint: remember to remove stop words.

```
# Filter down to "Homer Simpson"
# split the spoken_words into bigrams and create two seperate columns for the words
# Remove the stop words
# Group by the two words and count the amount of occurences
# Dispaly the top result
common_chars %>%
  filter(role == "Homer Simpson") %>%
  unnest_tokens(bigram, spoken_words, token = "ngrams", n = 2) %>%
  separate(bigram, c("word1", "word2"), sep = " ") %>%
  anti_join(stop_words, by = c("word1" = "word")) %>%
  anti_join(stop_words, by = c("word2" = "word")) %>%
  group_by(word1, word2) %>%
  count() %>%
  arrange(-n) %>%
  head(1)
```

```
## # A tibble: 1 x 3
## # Groups:   word1, word2 [1]
##   word1 word2     n
##   <chr> <chr> <int>
## 1 woo   hoo     312
```

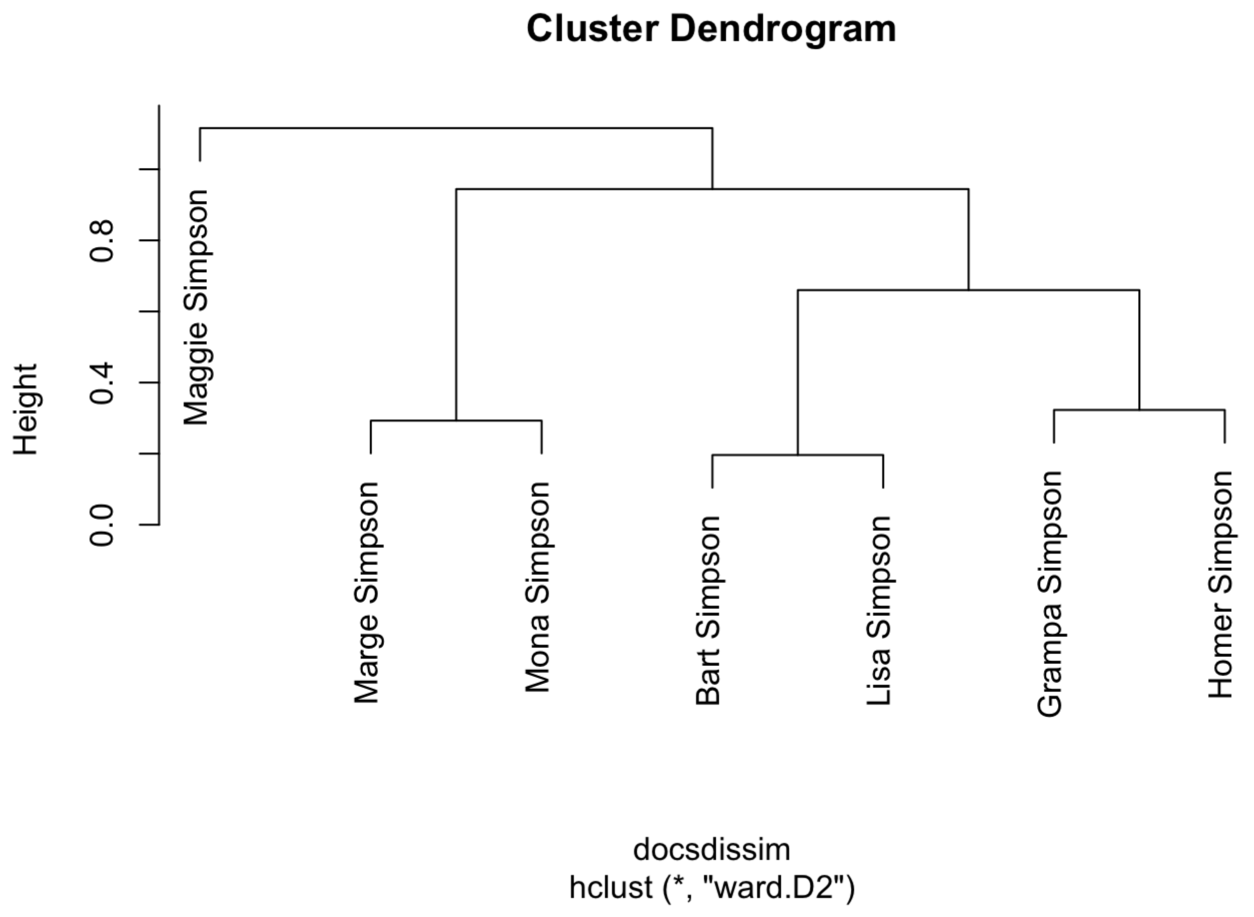
(d) What's the most significant word spoken by Homer Simpson in terms of `tf-idf` ?

```
# Create word tokens from dataset and remove stopwords
# Group by words and sort by the count of the words
char_tokens <- common_chars %>%
  unnest_tokens(word, spoken_words) %>%
  anti_join(stop_words) %>%
  group_by(role) %>%
  count(word, sort = TRUE) %>%
  arrange(role)

# Use bind_tf_idf to get our tf-idf values
# Filter to "Homer Simpson" and display the top row
char_tokens %>%
  bind_tf_idf(word, role, n) %>%
  filter(role == "Homer Simpson") %>%
  slice_max(tf_idf, n = 1)
```

```
## # A tibble: 1 x 6
## # Groups:   role [1]
##   role          word      n     tf   idf tf_idf
##   <chr>         <chr> <int>  <dbl> <dbl>  <dbl>
## 1 Homer Simpson marge  1796 0.0188 0.818 0.0154
```

(e) Use hierarchical clustering to classify the characters with Family name “Simpson”.

```
# Create a dataaframe of characters with Family name "Simpson" using str_detect
simpsons_df <- char_tokens %>%
                filter(str_detect(role, "Simpson"))

# Use document-term matrix to create a dendrogram
docsdissim <- dist(as.matrix(cast_dtm(simpsons_df, role, word, n)), method = "cosine")
h <- hclust(docsdissim, method = "ward.D2")
plot(h)
```


Cluster Dendrogram
Height
0.0
0.4
0.8
Maggie Simpson
Marge Simpson
Mona Simpson
Bart Simpson
Lisa Simpson
Grampa Simpson
Homer Simpson
docsdissim
hclust (*, "ward.D2")