````
---
title: "String Manipulation and Regular Expressions
Assignment"
author: "Vickie Gray"
date: "`r Sys.Date()`"
output: html_document
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)

```


##### Assignment Instructions

Complete all questions below. After completing the
assignment, knit your document, and download both your
.Rmd and knitted output. Upload your files for peer
review.

For each response, include comments detailing your
response and what each line does. Ensure you test your
functions with sufficient test cases to identify and
correct any potential bugs.

##### Required Libraries

Load the stringr library

```{r libraries}
library(tidyverse) # stringr is included
```


##### Question 1.

Use str_c to put `(` before the area codes followed by
`)` and a space followed by the phone number.

```{r question-1-response}
````
````

```
### Answer should be of the form "(703) 5551212" "(863) 1234567" "(404) 7891234" "(202) 4799747"

area_codes <- c(703, 863, 404, 202)
phone_nums <- c(5551212, 1234567, 7891234, 4799747)

str_c("(", area_codes, ") ", phone_nums)
```

Question 2.

Create a function that receives a single word as an input. Use str_length() and str_sub() to extract the middle character from the string. What will you do if the string has an even number of characters? Test your function on the strings “hamburger” and “hotdog”

```{r question-2-response}
# we can use the ceiling function to round up 1/2 the length if the string has an even number of characters

word_func <- function(new_word) {
  nw_length <- str_length(new_word)
  nw_length # the length of the string
  middle <- ceiling(nw_length / 2)
 # middle is the value at the middle of the string; using ceiling, negative numbers round down, positive number round up to the smallest integer larger than n
  str_sub(new_word, middle, middle) # subset starting and ending at the value stored in middle
}
word_func("hotdog")
word_func("hamburger")
word_func("able")
word_func("new day")
```

Question 3.

How would you match the sequence "’\? Note this is a

double quote, single quote, backslash and question mark. Build it up one piece at a time. Use it to identify that sequence contained in s2 below.

```{r question-3-response}
writeLines("\"\'\\?") # matches the sequence above.

s <- "\"'\\?"
s2 <- str_c("some stuff",s,"more!")

writeLines(s)

writeLines(s2)

str_view(s2, "\"'\\\\\\?", match = TRUE) # The trick is that the escape characters in s2 need to be escaped again in str_view.
# We need to create the pattern "\"'\\\\\?" to match the string "\"'\\?"
```

Question 4.

Using the words provided in stringr::words, create regular expressions that find all words that:

```{r question-4-response}

# End with "ing" or "ise"
str_view_all(words, "i(ng|se)", match = TRUE)

# Do not follow the rule "i before e except after c"
str_view_all(words, "([^c]ei|cie)", match = TRUE)

# Begin with at least two vowels and end with at least two consonants
str_view_all(words, "(^[aeiou][aeiou].*[^aeiou][^aeiou]$)", match = TRUE)

# Contain a repeated pair of letters (e.g. "church"
```

```
contains "ch" twice)
str_view_all(words, "(..).*\\1", match = TRUE)
```

```{r}
# Contain one letter other than e that is repeated in at least three places (e.g. “appropriate” contains three “p”s.)
str_view_all(words, "([^e]).*\\1.*\\1", match = TRUE)
# a caret inside square brackets negates e

```

Question 5.

Using the sentences provided in stringr::sentences, find all words that come after a “number” like “one”, “two”, … “twelve”. Pull out both the number and the word.

```{r question-5-response}
# create an object to hold the regex
num_then_word <-
"\\b(one|two|three|four|five|six|seven|eight|nine|ten) + (\\w+)"
# detect the sentences matching the regex object, then pipe the sentences to str_extract to pull out the number then word
sentences[str_detect(sentences, num_then_word)] %>%
  str_extract(num_then_word)
```

Question 6.

Using the sentences provided in stringr::sentences, view all sentences that contain the word “good” or the word “bad”. Get the sentence numbers where those words occur. Use str_replace_all() to replace the word “bad” with the word “horrible” and the word “good” with the word “great”. Look at the sentence numbers you found before to verify the words were replaced correctly.

```{r question-6-response}
# First find the sentences
df1 <- str_view_all(sentences, "good |bad ", match =
TRUE)
df1
# Then find the sentences
sent_num1 <- str_which(sentences, "good |bad ")
sent_num1 # sentence numbers for original words
# Now replace the words and save the sentences to a new
object
df2 <- str_replace_all(sentences, c("good" = "great",
"bad" = "horrible"))
# View the matches in the new object
str_view_all(df2, "great |horrible ", match = TRUE)

sent_num2 <- str_which(df2, "great |horrible ")
sent_num2 # sentence numbers for replaced words. Some
sentences already had the words "great" and "horrible"
in them, so the new list of sentences is longer than the
original. We can look for the intersect of the original
and changed sentences:
common_nums <- intersect(sent_num1, sent_num2)
common_nums # sentence numbers in common between
sentences with original and replaced words
intersect(common_nums, sent_num1) # test to see if the
in-common sentence numbers match the original sentence
numbers. They do. All the original words have been
changed.
```