# Class Activity 12

Arthur Viegas Eguia

April 19 2024

In this tutorial, we will learn about string manipulations using regular expressions and the `stringr` library in R. We will cover different examples and use cases to help you understand the concepts and functions related to string manipulation.

## Group Activity 1

```
x <- "My SSN is 593-29-9502 and my age is 55"
y <- "My phone number is 612-643-1539"
z <- "My old SSN number is 39532 9423."
out <- str_flatten(c(x,y,z), collapse = ". ")
```

### a. What characters in x will `str_view_all(x, "-..-")` find?

*answer:* It will extract 29

```
str_view_all(x, "-..-")
[1] | My SSN is 593<-29->9502 and my age is 55
```

### b. What pattern will `str_view_all(x, "-\\d{2}-")` find?

*answer:* This will do the same thing, it will extract 29

```
 str_view_all(x, "-\\d{2}-")
[1] | My SSN is 593<-29->9502 and my age is 55
```

### c. What pattern will `str_view_all(out, "\\d{2}\\.*")` find?

*answer:* It will extract every pair of numbers, so 59, 29, 95, 02, and 55. THen 61, 64, 15, 39, 39, 53,94 and 23. So all numbers and numbers followed by periods

```
str_view_all(out, "\\d{2}\\.*")
[1] | My SSN is <59>3-<29>-<95><02> and my age is <55.> My phone number is <61>2-<64>3-<15><39.> My old
```

### d. Use `str_view_all` to determine the correct regex pattern to identify all SSN in `out`

*answer:*

```
str_view_all(out, "[0-8]\\d{2}-\\d{2}-\\d{4}")
[1] | My SSN is <593-29-9502> and my age is 55. My phone number is 612-643-1539. My old SSN number is 3
```

This misses the oddly formatted SSN in the third entry. Rather than use a dash, we can specify the divider as `[-\\s]?` which allows either 0 or 1 occurrences of either a dash or space divider:

```
str_view_all(out, "[0-8]\\d{2}[-\\s]?\\d{2}[-\\s]?\\d{4}")
[1] | My SSN is <593-29-9502> and my age is 55. My phone number is 612-643-1539. My old SSN number is <
```

### e. Write a regular expression to extract dates in the format YYYY-MM-DD from a given text.

```
date_pattern <- "\\d{4}[-\\s]\\d{2}[-\\s]\\d{2}"
text <- "The event will take place on 2023-07-20 and end on 2023-07-22."
str_extract_all(text, date_pattern)
[[1]]
[1] "2023-07-20" "2023-07-22"
```

*answer:*

### f. Write a regular expression to extract all words that start with a capital letter in a given text.

*answer:*

```
capital_pattern <- "[A-Z][A-Za-z]*"
text <- "Alice and Bob went to the Market to buy some Groceries."
str_extract_all(text, capital_pattern)
[[1]]
[1] "Alice"     "Bob"       "Market"    "Groceries"
```

---

## Group Activity 2

Consider the following string:

```
string1 <- "100 dollars 100 pesos"
```

### a. Explain why the following matches the first 100 and not the second.

*answer:* This matches any digit followed by dollars. The second quantity is followed by pesos, and therefore it is not matched. So it matches the first 100 but not the second 100.

It will match one or more digits, that is followed by a space and dollars

```
str_view(string1, "\\d+(?= dollars)")
[1] │ <100> dollars 100 pesos
```

### b. Explain why the following matches the second 100 and not the first.

*answer:*

This is ?!. This will match one or more digits that is not followed by either a digit, or a space followed by "dollars"

```
str_view(string1, "\\d+(?!\\d| dollars)")
[1] │ 100 dollars <100> pesos
```

Please take a look at `string2`:

```
string2 <- "USD100 PESO100"
```

### c. Explain why the following matches the first 100 and not the second.

*answer:* This is look behind. This is looking for digits that are exactly of length 3 and have USD right behind it

```
str_view(string2, "(?<=USD)\\d{3}")
[1] | USD<100> PESO100
```

### d. Explain why the following matches the second 100 and not the first.

*answer:* In this case you have a not. So it matches exactly three digits that are not followed by. USD

```
str_view(string2, "(?<!USD)\\d{3}")
[1] | USD100 PESO<100>
```

---

## Group Activity 3

Now, we will use `TrumpTweetData.csv`, a dataset of tweets to demonstrate common tasks such as detecting patterns, filtering text, and summarizing string properties.

```
tweets<- read_csv("https://raw.githubusercontent.com/deepbas/statdatasets/main/TrumpTweetData.csv")
```

### a. What proportion of tweets (text) mention "America"?

```
tweets %>%
  summarize(prop = mean(str_detect(str_to_title(text), "America"))) ->foo
foo
# A tibble: 1 x 1
    prop
   <dbl>
1 0.0926
```

### b. What proportion of these tweets include "great"?

```
tweets %>%
  filter(str_detect(str_to_title(text), "America")) %>%
  summarize(prop = mean(str_detect(str_to_lower(text), "great")))
# A tibble: 1 x 1
   prop
  <dbl>
1   0.4
```

### c. What proportion of the tweets mention `@`?

```
#tweets %>% summarize(prop = mean(str_detect(text, "@"))
#Mentions
```

### d. Remove the tweets having mentions `@`.

```
#Mentions <- c("@[^\\s]+")

#tw_noMentions <- tweets %>% mutate(textNoMention. str_remove_all(text, Mentions))
```

### e. What poportion of tweets originated from an iPhone?

```
tweets %>% group_by(source) %>%  summarize(count = n()) %>%
  mutate(prop = count/sum(count)) %>% filter(row_number() == 3)
# A tibble: 1 x 3
  source count  prop
  <chr>  <int> <dbl>
1 iPhone   628 0.415
```