Strings and regular expressions

Recap: regular expressions

A *regular expression* is a pattern used to find matches in text.

Example: suppose I want to extract just the lecture number from the following file name. How would I do that?

```
"teaching/sta279-f23/slides/lecture_22.qmd"
```

number after lecture_

number after _

_\\d+ → _22

(?<=_)\\d+ → 22

Recap: regular expressions

A *regular expression* is a pattern used to find matches in text.

Example: suppose I want to extract just the lecture number from the following file name. How would I do that?

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd", "\\d+")
```

```
[1] "279"
```

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd", "_\\d+")
```

```
[1] "_22"
```

```r
str_extract("teaching/sta279-f23/slides/lecture_22.qmd",
            "(?<=_)\\d+")
```

```
[1] "22"
```

Recap: regular expressions

Last time, we learned the following regular expression tools:

- `\d` matches any digit (in R, have to type `\\d` because we write the regex in a string)
- `.` matches any character (except `\n`)
- `+` means “at least once”
- `(?<=)` and `(?=)` are positive lookbehinds and lookaheads
- `|` is alternation (one pattern or another)

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just `raspberry` and `blackberry`?

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just `raspberry` and `blackberry`?

```r
str_view(strings, "berry")
```

```
[3] │ rasp<berry>
[4] │ black<berry>
```

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “raspberry”, “blackberry”, “grrreat”, and “random”?

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “raspberry”, “blackberry”, “grrreat”, and “random”?

```r
str_view(strings, "r")
```

```
[3] │ <r>aspbe<r><r>y
[4] │ blackbe<r><r>y
[5] │ g<r><r><r>eat
[6] │ <r>andom
```

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just “raspberry”, “blackberry”, and “grrreat”?

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just “raspberry”, “blackberry”, and “grrreat”?

```r
str_view(strings, "rr+")
```

```
[3] │ raspbe<rr>y
[4] │ blackbe<rr>y
[5] │ g<rrr>eat
```

↑ at least 2 r s

```r
str_view(strings, "r{2,}")
```

```
[3] │ raspbe<rr>y
[4] │ blackbe<rr>y
[5] │ g<rrr>eat
```

↑ at least two times

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select just “grrreat”?

```r
str_view(strings, "r{3}")
```

```
[5] │ g<rrr>eat
```

3 occurences

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “apple”, “raspberry”, “blackberry”, and “grrreat”?

More regular expressions

```r
strings <- c("apple", "banana", "raspberry",
             "blackberry", "grrreat", "random")
```

How would I select “apple”, “raspberry”, “blackberry”, and “grrreat”?

```r
str_view(strings, "(.)\\1")
```

```
[1] │ a<pp>le
[3] │ raspbe<rr>y
[4] │ blackbe<rr>y
[5] │ g<rr>reat
```

← back reference

capture group

(capturing a particular pattern)

↳ one occurence of any character

\\1 : back reference

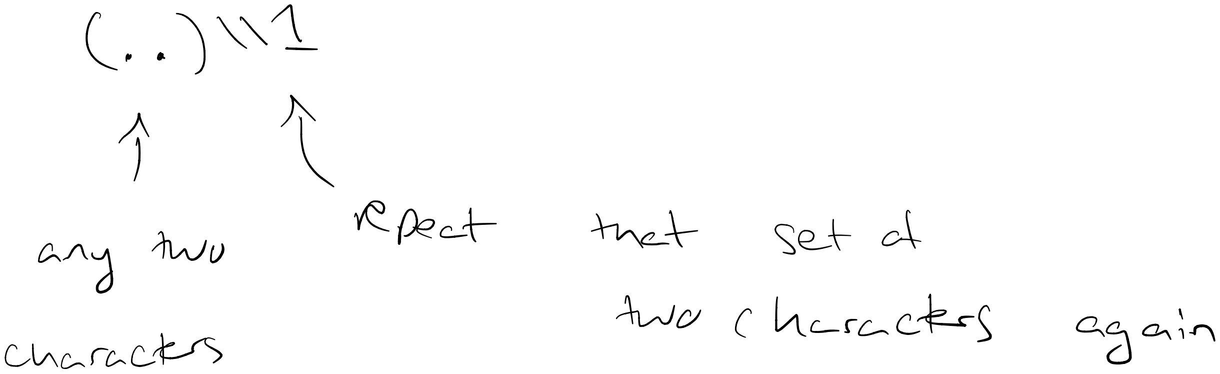
refer to a previously captured group

(.)\\1 : any character, and then that character again

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "grrreat", "random")
```

How would I select “papa”, “banana”, and “memento”?



(..)\\1

any two characters

repeat that set of two characters again

More regular expressions

```r
strings <- c("papa", "banana", "memento",
             "blackberry", "grrreat", "random")
```

How would I select “papa”, “banana”, and “memento”?

```r
str_view(strings, "(..)\\1")
```

```
[1] │ <papa>
[2] │ b<anan>a
[3] │ <meme>nto
```

```r
str_view(strings, "(..)+")
```

```
[1] │ <papa>
[2] │ <banana>
[3] │ <mement>o
[4] │ <blackberry>
[5] │ <grrrea>t
[6] │ <random>
```

More regular expressions

```
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

pattern: start with \$, end with \$

$$\backslash\backslash\$ \underbrace{.+}_{\text{something}} \backslash\backslash\$$$

More regular expressions

```r
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

```r
str_extract("The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\su
            "\\$.+\\$")
```

```
[1] "$\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

Issue: regular expressions are greedy (by default you get the biggest match)

More regular expressions

```
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

Option 1:

```
str_extract_all("The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \
                "\\$.+?\\$")
```

```
[[1]]
[1] "$\\mu$"                                  "$\\mu = \\frac{1}{n} \\sum_i
x_i$"
```

? means "don't be greedy"

\\$.+ \\$

anything

Option 2:

Start with $, then something not $, end with $

More regular expressions

```
"The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \\sum_i x_i$"
```

How would I extract `$\mu$` and `$\mu = \frac{1}{n} \sum_i x_i$`?

Option 2:

```
str_extract_all("The mean $\\mu$ is defined by $\\mu = \\frac{1}{n} \
                "\\$[^\\$]+\\$")
```

```
[[1]]
[1] "$\\mu$"                              "$\\mu = \\frac{1}{n} \\sum_i
x_i$"
```

[ ] : character class : a defined group of characters

[^] : everything except specified characters

[^\\$] : every character except $

Class activity

- Work independently or with a neighbor on the class activity
- At the end of class, submit your work as an HTML file on Canvas (one per group, list all your names)