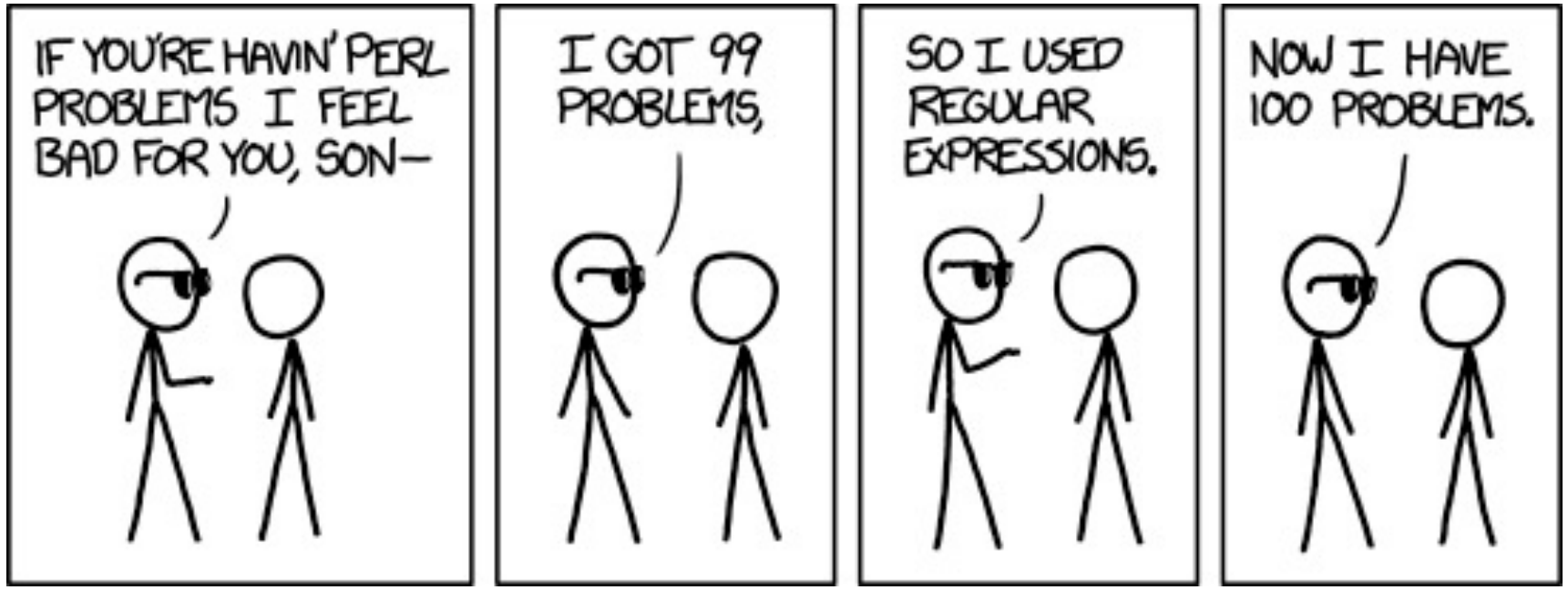
Regular Expressions



https://xkcd.com/1171/

https://xkcd.com/208/

Regular Expressions

As you read, regular expressions (or regex) provide a language for describing a search pattern that will match patterns in a string. They can get very complex, but usually we only need relatively simple regular expressions for data science applications.

\A	start of string
\Z	end of string
\w	match word character (note: lowercase w)
\W	match any non-word character (note: uppercase W)
.	any character (other than new line)
[ ]	match any (single) character inside bracket
[ ]*	match 0 or more characters inside bracket
[ ]+	match 1 or more characters inside bracket
[^]	match any number of character NOT inside bracket

[note: you will need to write two slashes in R]

0-9	numeric digits
a-z	lowercase letters
A-Z	uppercase letters

[note: these must be used inside brackets]

Example #1

How would you describe the following regex query?

[\w]+ing

Example #2

How would you describe the following regex query?

"[^"]+"

stringi

The stringi package is probably my favorite R package—at least, my favorite R package that I did not create! It provides an amazing amount of fast functionality that almost always just works as I expect it to. All functions start with the prefix “stri_” and most accept either a regex OR fixed string search pattern (the latter turns out to be very helpful in a lot of cases).

As an example, here is an example of a regular expression that counts the number of words (from the latin alphabet) with capital letters:

```
stri_count(input, regex = "[A-Z][a-z]+")
```

Using Regular Expressions

There are several commons things we can do once we are able to match a pattern in a string, all supported by the R stringi package:

stri_count	**count the number of time the pattern occurs**
stri_detect	**does pattern occur at least once?**
stri_extract_all	**extract all occurances of the pattern**
stri_replace_all	**replace all occurances of the pattern with a fixed string**
stri_split	**split string into substrings, using patterns as the split**

There are also some related stringi functions that do not take a regular expression as an input:

stri_length	**how many characters are in the string**
stri_sub	**take a substring by position**
stri_trans_toupper	**convert to uppercase**
stri_trans_tolower	**convert to lowercase**
stri_trans_totitle	**convert to title case**