Week 3 Strings and Regex

Ethan Haley

2021-02-19

```
dataURL <- "https://raw.githubusercontent.com/fivethirtyeight/data/master/college-majors/majors-list.cs
majors <- read.csv(url(dataURL))
majors <- majors$Major
majors[1:11]
```

Get some data about college majors, from fivethirtyeight.com's Github

```
##  [1] "GENERAL AGRICULTURE"
##  [2] "AGRICULTURE PRODUCTION AND MANAGEMENT"
##  [3] "AGRICULTURAL ECONOMICS"
##  [4] "ANIMAL SCIENCES"
##  [5] "FOOD SCIENCE"
##  [6] "PLANT SCIENCE AND AGRONOMY"
##  [7] "SOIL SCIENCE"
##  [8] "MISCELLANEOUS AGRICULTURE"
##  [9] "FORESTRY"
## [10] "NATURAL RESOURCES MANAGEMENT"
## [11] "FINE ARTS"
```

Identify the majors that contain either "DATA" or "STATISTICS"

```
majors[str_detect(majors, "DATA|STATISTICS")]
```

```
## [1] "MANAGEMENT INFORMATION SYSTEMS AND STATISTICS"
## [2] "COMPUTER PROGRAMMING AND DATA PROCESSING"
## [3] "STATISTICS AND DECISION SCIENCE"
```

Practice manipulating strings with regex in R

```
fruitvec <- c("bell pepper", "bilberry", "blackberry", "blood orange",
              "blueberry", "cantaloupe", "chili pepper", "cloudberry",
              "elderberry", "lime", "lychee", "mulberry", "olive",
              "salal berry")
fruitvec
```

```
##  [1] "bell pepper"  "bilberry"     "blackberry"   "blood orange" "blueberry"
##  [6] "cantaloupe"   "chili pepper" "cloudberry"   "elderberry"   "lime"
## [11] "lychee"       "mulberry"     "olive"        "salal berry"
```

Now the task is to create a string representation of `fruitvec`:

```
strfruitvec <- str_flatten(fruitvec, collapse = '", "')
cat(c('c("', strfruitvec, '")'), sep = '')
```

```
## c("bell pepper", "bilberry", "blackberry", "blood orange", "blueberry", "cantaloupe", "chili pepper"
```

So the `cat` function produces the desired output format, but in order to produce an actual object that is a string and includes double quotes around each item, I don't know if that's possible, without backslashes appearing in the object.

```
stringy_vec <- function(charvec) {
  flat <- str_flatten(charvec, collapse = '", "')
  cat(c('c("', flat, '")'), sep = '')
}
cheeses <- c('gouda', 'brie', 'stilton', 'american', 'string cheese')
stringy_vec(cheeses)
```

```
## c("gouda", "brie", "stilton", "american", "string cheese")
```

Describe, in words, what these expressions will match:

`(.)\1\1`

A character that occurs 3 times in a row

`"(.)(.)\\2\\1"`

A palindromic series of 4 characters, like "anna" or "zzzz"

`(..)\1`

Four characters where 3 and 4 are the same as 1 and 2, like "yoyo"

`"(.).\\1.\\1"`

A 5-char sequence where chars 1, 3, and 5 are the same, like "orono"

`"(.)(.)(.).*\\3\\2\\1"`

A series of at least 6 chars, ending with the same 3 it started with,

but in reversed order, like "redder" or "madam, i'm adam"

Construct regular expressions to match words that:

–Start and end with the same character.

`"\\b(.)\\S*\\1\\b"` **allows internal numbers and symbols, so it**

matches "pop-up" and "s#$%s", e.g.

–Contain a repeated pair of letters (e.g. "church" contains "ch" repeated twice.)

`"\\b.*([a-zA-Z][a-zA-Z])[a-zA-Z-]*\\1[a-zA-Z]*\\b"` **allows only letters,**

plus hyphens between the repeated pairs, so it matches 'mai-tai'

but only the "yoyo" part of "yoyo-lover", e.g.

–Contain one letter repeated in at least three places (e.g. "eleven" contains three "e"s.)

```
"\\b[a-zA-Z]*([a-zA-Z])[a-zA-Z]+\\1[a-zA-Z]+\\1[a-zA-Z]*\\b"  ...
```