String manipulation

Michael Mbajwa

2021-10-25

This little tutorial aims to make me familiar with some of the functions of the `stringr` package and a few regular expressions.

Strings and escape sequences in R

Write a sentence with escape sequences. I try the sentence: `"It's the end of the world!" he said.\` . Assign the string to a variable and try as`print()`, `cat()` and `writeLines()`.

```
str_var <- '"It\'s the end of the world!" he said.\\'
print(str_var)
```

```
## [1] "\"It’s the end of the world!\" he said.\\"
```

```
cat('\n')
```

```
cat(str_var)
```

```
## "It’s the end of the world!" he said.\
```

```
cat('\n')
```

```
writeLines(str_var)
```

```
## "It’s the end of the world!" he said.\
```

stringr functions

We will be using the `words` data that is built into `stringr`.

```
words[1:50]
```

```
##  [1] "a"           "able"        "about"       "absolute"    "accept"
##  [6] "account"     "achieve"     "across"      "act"         "active"
## [11] "actual"      "add"         "address"     "admit"       "advertise"
## [16] "affect"      "afford"      "after"       "afternoon"   "again"
## [21] "against"     "age"         "agent"       "ago"         "agree"
## [26] "air"         "all"         "allow"       "almost"      "along"
## [31] "already"     "alright"     "also"        "although"    "always"
## [36] "america"     "amount"      "and"         "another"     "answer"
## [41] "any"         "apart"       "apparent"    "appear"      "apply"
## [46] "appoint"     "approach"    "appropriate" "area"        "argue"
```

```
length(words)
```

```
## [1] 980
```

```
str_length(words)[1:20]
```

```
##  [1] 1 4 5 8 6 7 7 6 3 6 6 3 7 5 9 6 6 5 9 5
```

Select words that

1. Contain a y

```
str_subset(words, 'y')
```

```
##  [1] "already"     "always"      "any"         "apply"       "authority"  
##  [6] "away"        "baby"        "beauty"      "body"        "boy"        
## [11] "busy"        "buy"         "by"          "carry"       "city"       
## [16] "community"   "company"     "copy"        "country"     "county"     
## [21] "day"         "dry"         "early"       "easy"        "economy"    
## [26] "employ"      "enjoy"       "every"       "eye"         "family"     
## [31] "fly"         "friday"      "germany"     "goodbye"     "guy"        
## [36] "happy"       "heavy"       "history"     "holiday"     "identify"   
## [41] "industry"    "key"         "lady"        "lay"         "likely"     
## [46] "many"        "marry"       "may"         "maybe"       "monday"     
## [51] "money"       "necessary"   "okay"        "only"        "opportunity"
## [56] "party"       "pay"         "play"        "policy"      "pretty"     
## [61] "quality"     "ready"       "really"      "saturday"    "say"        
## [66] "secretary"   "society"     "sorry"       "stay"        "story"      
## [71] "strategy"    "study"       "sunday"      "supply"      "system"     
## [76] "they"        "thirty"      "thursday"    "today"       "try"        
## [81] "tuesday"     "twenty"      "type"        "university"  "very"       
## [86] "way"         "wednesday"   "why"         "worry"       "year"       
## [91] "yes"         "yesterday"   "yet"         "you"         "young"
```

2. Start with y

```
str_subset(words, '^y')
```

```
## [1] "year"      "yes"       "yesterday" "yet"       "you"       "young"
```

3. Contain a y within the word

```
str_subset(words, '.y.')
```

```
## [1] "always"  "eye"     "goodbye" "maybe"   "system"  "type"
```

Extract the y and the previous character. Note: Use the function `unique()` around the results to avoid printing many empty matches.

```
unique(str_match(words, '(.{1}y)'))
```

```
##       [,1] [,2]
##  [1,] NA   NA  
##  [2,] "dy" "dy"
##  [3,] "ay" "ay"
##  [4,] "ny" "ny"
##  [5,] "ly" "ly"
##  [6,] "ty" "ty"
##  [7,] "by" "by"
##  [8,] "oy" "oy"
##  [9,] "sy" "sy"
## [10,] "uy" "uy"
## [11,] "ry" "ry"
## [12,] "py" "py"
## [13,] "my" "my"
## [14,] "ey" "ey"
## [15,] "vy" "vy"
## [16,] "fy" "fy"
## [17,] "cy" "cy"
## [18,] "gy" "gy"
## [19,] "hy" "hy"
```

Get the lengths of the first ten words I use `head(words, 10)` as a convenient way to access the elements of the `words` vector.

```
first_ten <- head(words, 10)
str_length(first_ten)
```

```
##  [1] 1 4 5 8 6 7 7 6 3 6
```

Viral research

Read the genome sequence of the Hepatitis D virus: hepd.fasta.

```
hepd_genome <- readr::read_lines("https://biostat2.uni.lu/practicals/data/hepd.fasta")
```

```


str_length(hepd_genome)
```

I find the length of the genome sequence?

```
## [1] 1682
```

```
# Length is 1682
```

```
seq_comp <- unique(str_split(hepd_genome, '')[[1]])
seq_comp
```

I find the sequence composition and how often each character occur?

```
## [1] "A" "T" "G" "C"
```

```
# [1] "A" "T" "G" "C"
str_count(hepd_genome, seq_comp)
```

```
## [1] 339 354 485 504
```

```
# [1] 339 354 485 504
```

Find motifs in the sequence using `str_locate()`. I find all matches of the sequence *ATG* in the genome sequence.

```
str_locate_all(hepd_genome, 'ATG')
```

```
## [[1]]
##       start  end
##  [1,]     1    3
##  [2,]   130  132
##  [3,]   378  380
##  [4,]   581  583
##  [5,]   586  588
##  [6,]   637  639
##  [7,]   686  688
##  [8,]   695  697
##  [9,]   758  760
## [10,]   765  767
## [11,]   858  860
## [12,]   888  890
## [13,]   893  895
## [14,]  1015 1017
## [15,]  1038 1040
## [16,]  1089 1091
## [17,]  1440 1442
## [18,]  1457 1459
```