DATA607_Home_Work_3

Dilip Ganesan

02/16/2017

Home Work 3 : Regular Expression and R

3. Copy the introductory example. The vector name stores the extracted names.

```
raw.data="555-1239Moe Szyslak(636) 555-0113Burns, C. Montgomery555-6542Rev. Timothy Lovejoy555 8904Ned F

name <- unlist(str_extract_all(raw.data, "[[:alpha:]., ]{2,}"))
name
```

```
## [1] "Moe Szyslak"          "Burns, C. Montgomery" "Rev. Timothy Lovejoy"
## [4] "Ned Flanders"         "Simpson, Homer"       "Dr. Julius Hibbert"
```

```
# (a) Use the tools of this chapter to rearrange the vector so that all elements conform to thestandard

first=str_extract(name,"[[:alpha:]]{1,} |\\. [[:alpha:]]{1,}|\\, [[:alpha:]]{2,}")
first=str_extract(first,"[[:alpha:]]{1,}")

last=str_extract(name,"[[:alpha:]]{1,}\\,|\\b [[:alpha:]]{2,}")
last=str_extract(last,"[[:alpha:]]{1,}")

df=data.frame(first, last)
df
```

```
##        first     last
## 1        Moe  Szyslak
## 2 Montgomery    Burns
## 3    Timothy  Lovejoy
## 4        Ned Flanders
## 5      Homer  Simpson
## 6     Julius  Hibbert
```

```
df_args <- c(df, sep=" ")
do.call(paste, df_args)
```

```
## [1] "Moe Szyslak"      "Montgomery Burns" "Timothy Lovejoy"
## [4] "Ned Flanders"     "Homer Simpson"    "Julius Hibbert"
```

```
# (b) Construct a logical vector indicating whether a character has a title (i.e., Rev. and Dr.).

unlist(str_detect(name,"^Rev|Dr"))
```

```
## [1] FALSE FALSE  TRUE FALSE FALSE  TRUE
```

```
# (c) Construct a logical vector indicating whether a character has a second name.
# Approach, when the name has a comma, i am assume it to be second name.
secondname = str_detect(name, ',')
secondname
```

```
## [1] FALSE  TRUE FALSE FALSE  TRUE FALSE
```

4. Describe the types of strings that conform to the following regular expressions and construct an example that is matched by the regular expression.

```r
#(a) [0-9]+\\$
#Ans : This will return numbers followed by a $ sign.

a="147$"
str_extract(a,'[0-9]+\\$')
```

```
## [1] "147$"
```

```r
#(b) \\b[a-z]{1,4}\\b
# Ans: This returns 1-4 lower case characters which are bounded by word edge

b = "%abc $"
str_extract(b, '\\b[a-z]{1,4}\\b')
```

```
## [1] "abc"
```

```r
#(c) .*?\\.txt$
# Ans: This will return characters which end with .txt files

c="1234 dilip.txt"
str_extract(c,".*?\\.txt$")
```

```
## [1] "1234 dilip.txt"
```

```r
#(d) \\d{2}/\\d{2}/\\d{4}
#Ans : This will give the date format
d="01/01/2017"
str_extract(d,"\\d{2}/\\d{2}/\\d{4}")
```

```
## [1] "01/01/2017"
```

```r
#(e) <(.+?)>.+?</\\1>
#Ans : This is used to extract elements like xml tags <abc> and </abc>

e="<td>dilip is doing R home work </td>"
str_extract(e,'<(.+?)>.+?</\\1>')
```

```
## [1] "<td>dilip is doing R home work </td>"
```

9. The following code hides a secret message. Crack it with R and regular expressions.

```r
# Used the hint from the book to find the removal of lower cases and numbers.
Jumblecode="clcopCow1zmstc0d87wnkig7OvdicpNuggvhryn92Gjuwczi8hqrfpRxs5Aj5dwpn0TanwoUwisdij7Lj8kpf03AT5I

newcode=str_extract_all(Jumblecode,'[^[a-z]|[0-9]]')
newcode
```

```
## [[1]]
##  [1] "C" "O" "N" "G" "R" "A" "T" "U" "L" "A" "T" "I" "O" "N" "S" "." "Y"
## [18] "O" "U" "." "A" "R" "E" "." "A" "." "S" "U" "P" "E" "R" "N" "E" "R"
## [35] "D" "!"
```