RegEx

Haewon Lee

Table of contents

1 Regular Expression

정규표현식의 정의 : 특정 문자열을 가리키는 패턴

아주 간단하고 쉬운 정규표현식의 한가지 예를 들어보면

```
\d
```

이것은 0부터 9까지의 숫자와 매칭된다. 그리고 다음처럼 제법 복잡하게 발전시킬 수도 있다.

```
^0\d{1,2}-?\d{2,4}-?\d{4}
```

이 패턴은 한국의 전화번호에 대응하는 패턴인데 0으로 시작하는 핸드폰번호 또는 지역번호 - 0으로 시작하지 않는 2~4자리 번호 - 4자리 번호와 매칭이 되는 패턴이다.

1.1 RegexPal 연습하기

http://www.regexpal.com

이 웹사이트에 들어가면 위에 정규표현식 입력영역이 있고 아래에 목표 텍스트 영역이 있다.

https://regexr.com/ : 설명이 같이 있어서 복잡한 패턴을 공부하는데 도움이 된다.

일단 RegexPal로 숫자 패턴 연습을 해보자. 아래 영역에는 다양한 전화번호 또는 여러가지 번호나 텍스트를 붙여넣어놓자

위의 영역을 지우고 3이라는 숫자만 입력해보자

[0-9] 라고 입력해보자

숫자의 범위를 제한해보자 0,1,2,3,5,8 의 번호만 찾고 싶으면 [012358] 을 입력한다

11 개의 숫자를 찾는 방법은? [0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9][0-9]

[0-9] 표현은 다음과 같은 단축문자 로 바꿀 수 있다.

```
\d
```

다음과 같이 전화번호를 찾아볼 수 있다

```
\d\d\d-\d\d\d\d-\d\d\d\d
```

숫자 이외의 문자를 찾는데 쓰는 단축문자를 쓰면..

```
\d\d\d\D\d\d\d\d\D\d\d\d\d
```

```
\d\d\d.\d\d\d\d.\d\d\d\d
```

점 (.) 을 사용하면 대부분의 아무 문자나 찾을 수 있는 와일드카드 역할을 한다. (개행문자 제외)

수량자를 사용해보자

```
\d{3}-?\d{3}-?\d{4}
```

{3} 중괄호 안의 숫자는 몇번 나오는지 숫자를 지정한다. {3,9}는 3~9회, {2,}는 2회 이상, {,99}는 99회 이하, ..

물음표(?) 도 수량자인데 한개 이하를 의미 (0 또는 1)

덧셈기호 (+)는 하나 이상

별표 (*)는 0 이상을 의미

```
(\d{3,4}[.-]?)+
```

위의 정규식은 숫자 세개 또는 네개 다음에 점이나 하이픈이 한번 이하로 등장하는 패턴을 묶은 그룹이 한번 이상 등장한다는 뜻이다. (괄호는 참조그룹의 의미)

1.1.1 정규표현식 기호들 (1)

표현	설명
*	0개 또는 그 이상 예) x* = x가 0번또는 그 이상 반복
+	1개 또는 그 이상 예) x+
?	0 또는 1개 (1개 이하) 예) x?
.	무엇이든 한글자
^	시작 문자를 지정, 예) ^[abc] = abc중 한글자 포함해서 시작
[^]	해당 문자를 제외한 모든 것, 예) [^abc] = a,b,c만 빼고 모두
$	끝 문자 예) x$ = x로 종료
[가-힣]	모든 한글 글자 중 1개
|	또는
()	소괄호에 묶인 문자는 한개의 그룹으로 표현되거나 and로 됨

1.1.2 정규표현식 기호들 (2)

\d - 숫자와 매치, [0-9]와 동일한 표현식이다.

\D - 숫자가 아닌 것과 매치, [^0-9]와 동일한 표현식

\s - whitespace 문자와 매치, [\t\n\r\f\v]와 동일한 표현식이다.
맨 앞의 빈 칸은 공백문자(space)를 의미

\S - whitespace 문자가 아닌 것과 매치, [^\t\n\r\f\v]와 동일한 표현식

\w - 문자+숫자(alphanumeric)와 매치, [a-zA-Z0-9_]와 동일한 표현식

\W - 문자+숫자(alphanumeric)가 아닌 문자와 매치, [^a-zA-Z0-9_]와 동일한 표현식

\n - 줄바꿈 문자

\t - 탭 문자

\v - 수직 탭 문자

\b - 단어 경계 : 백스페이스는 반드시 [\b]로 입력해야 한다.

\c - 제어 문자

\f - 폼 피드 문자

\x - 16진수값

\0 - 8진수 값

1.1.3 특수표현 : "[: :]" 형태로 존재하는 특수한 표현식이 존재

[::]	의미
[:digit:]	숫자
[:alpha:]	문자
[:lower:]	소문자
[:upper:]	대문자
[:alnum:]	문자+숫자
[:punct:]	기호
[:graph:]	문자+숫자+기호
[:space:]	띄어쓰기
[:blank:]	띄어쓰기+탭

1.1.4 이어지거나 이어지지 않는 문자열

이어지는 문자열 (?=)

- 가나다(?=a) : 뒤에 "a"가 있는 "가나다"

이어지지 않는 문자열 (?!=)

- 가나다(?!=b) : 뒤에 "b"로 이어지지 않는 "가나다"

앞에서 이어지는 문자열 (?<=)

- (?<=c) 가나다 : 앞에 "c"가 있는 "가나다"

앞에서 이어지 않는 문자열 (?<!)

- (?<!c) 가나다 : 앞에 "c"가 없는 "가나다"

E-mail

```
^[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*@[0-9a-zA-Z]([-_\.]?[0-9a-zA-Z])*\.[a-zA-Z]{2,3}$
```

'시작을' 0~9 사이 숫자 or a-z A-Z 아무거나로 시작하고
중간에 - _ . 같은 문자가 있을 수도 있고 없을 수도 있으며
그 후에 0~9 사이 숫자 or a-z A-Z 중 하나의 문자가 없거나 연달아 나올수 있으며
@ 가 반드시 존재하고
0-9a-zA-Z 여기서 하나가 있고
중간에 - _ . 같은 문자가 있을수도 있고 없을수도 있으며,
그 후에 0~9 사이 숫자 or a-z A-Z 중 하나의 문자가 없거나 연달아 나올수 있으며
반드시 . 이 존재하고, [a-zA-Z] 의 문자가 2개나 3개가 존재하면서 종료

1.1.5 탐욕적 및 게으른 수량자

정규 표현식에서는 일치하는 패턴을 찾는 횟수 제한이 없어 필요 이상의 상황을 연출하기도 하는데 이것은 의도적으로 수량자를 탐욕적으로 만들었기 때문이다. 문법에서 말하는 탐욕적 수량자(Greedy Quantifier)란 가능하면 가장 큰 덩어리를 찾는다는 뜻이다. 반대의 개념인 게으른 수량자(Lazy Quantifier)는 패턴에 근접하는 최소한의 덩어리를 찾는다.

- 탐욕적 수량자: `*`, `+`, `{n,}`
- 게으른 수량자: `*?`, `+?`, `{n,}?`

실제 사용례 - Pathology report

병리결과지에서 추출

```
\d{8}[\w\W]*?(?=\d{8}|$)
```

8개의 숫자 (병록번호)로 시작하는 텍스트 …… 8개의 병록번호 전까지 또는 마지막 전까지 모든 문자 선택

```r
library(magrittr)
path_reading <- read.delim(file="pathology reading.txt", header = F)
str(path_reading)
```

```
'data.frame':   294 obs. of  1 variable:
 $ V1: chr  "01354696" "【병리번호】S24-7979" "【확 인 자】공준석 / 공준석" "【 Gross  】" ...
```

```r
pattern1 <- "\\d{8}"
match1 <- regexpr(pattern1,path_reading[,1])
str(match1)
```

```
 int [1:294] 1 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
 - attr(*, "match.length")= int [1:294] 8 -1 -1 -1 -1 -1 -1 -1 -1 -1 ...
```

```r
seq1 <- 1:294
seq1[match1==1] -> match2
match22 <- c(match2,295)
mat1 <- cbind(match2, match22[2:9]-1)
report1 <- NULL
for (i in 1:nrow(mat1)) {
  report1 <- c(report1,
               paste(path_reading[mat1[i,1]:mat1[i,2],1]
                     ,collapse = "\n"))
}
report1[1] %>%cat
```

```
01354696
【병리번호】S24-7979
【확 인 자】공준석 / 공준석
【 Gross  】
# Right upper lobe + right middle lobe. Received is a right upper and mid lobectomy specimen. Right upper l
<A; peribronchial lymph node, B; bronchial resection margin, C; tumor with bronchial resection margin, D~G
# Lymph node #2, #4, #7, #10, #11. Entirely embedded.
【확인결과】
Lung, right upper lobe and right middle lobe, right bilobectomy:
     1. Carcinoid tumor (typical carcinoid), right upper lobe
             with 1) size of invasion; 6.3x6.0x4.0cm
                    2) mitotic count: 1/10HPF
                    3) necrosis: absent
                    4) no involvement of visceral pleura (PL0)
                    5) lymphatic invasion: absent
                    6) vascular invasion: absent
                    7) perineural invasion: absent
```

```
                    8) clear bronchial resection margin
          [pathologic stage: pT3]
     2. No abnormal findings in the background lung
Lymph node, dissection:
     No metastasis in all 16 lymph nodes
     (peribronchial; 0/5, #2: 0/1, #4: 0.1, #7: 0/6, #10: 0/2, #11: 0/1)
     [pathologic stage: pN0]
PL0
PL1; Tumor invades beyond the elastic layer
PL2; Tumor invade to the pleural surface
PL3; Tumor invade into any component of the parietal pleura
<Proposed grading of resected early-stage invasive non-mucinous lung adenocarcinoma>
Grade 1: Lepidic-predominant with no or <20% high-grade pattern
Grade 2: Acinar or papillary-predominant with no or <20% high-grade pattern
Grade 3: Any tumor with  20% high-grade pattern (solid, micropapillary, cribriform, or complex glandular pa
*관련병리번호 :
   I24-8919    CD56 [F]  -  Positive(+)
   I24-8920    Chromogranin A [F]  -  Positive(+), focal
   I24-8921    Ki-67 [F]  -  Positive(+), 2%
   I24-8923    Synaptophysin [F]  -  Positive(+), weak
   I24-8924    TTF-1 ( Thyroid Transcription Factor-1 ) [F]  -  Negative(-)
   I24-8922    P40 [F]  -  Negative(-)
```

```
library(stringr)
regexpr(pattern1, report1) -> match_ch
match_ch %>% str
```

```
 int [1:8] 1 1 1 1 1 1 1 1
 - attr(*, "match.length")= int [1:8] 8 8 8 8 8 8 8 8
```

```
chartno <- regmatches(report1,match_ch)
chartno
```

```
[1] "01354696" "01237451" "01347549" "01352042" "01351999" "01319678" "01352389"
[8] "01354696"
```

```
pattern_soi <- "(size[ ]+of[ ]+invasion;[ ]+)([\\w\\d.]+)"
regexpr(pattern_soi,report1,perl = T) -> match_soi
size_invasion <- regmatches(report1, match_soi)
size_invasion
```

```
[1] "size of invasion; 6.3x6.0x4.0cm" "size of invasion; 1.8x0.9cm"
[3] "size of invasion; 1.5x1.0cm"     "size of invasion; 3.7x3.3x3.2cm"
[5] "size of invasion; 2.5x2.4x2.3cm" "size of invasion; 2.7x2.6cm"
[7] "size of invasion; 6.3x6.0x4.0cm"
```

```
match_soi <- str_match(report1, pattern_soi)
size_invasion <- match_soi[,3]
size_invasion
```

```
[1] "6.3x6.0x4.0cm" "1.8x0.9cm"     "1.5x1.0cm"     "3.7x3.3x3.2cm"
[5] "2.5x2.4x2.3cm" NA              "2.7x2.6cm"     "6.3x6.0x4.0cm"
```

```
pattern_size <- "([\\d.]{1,5})x([\\d.]{1,5})x?([\\d.]{1,5})?"
str_match(size_invasion,pattern_size)
```

```
     [,1]           [,2]  [,3]  [,4]
[1,] "6.3x6.0x4.0" "6.3" "6.0" "4.0"
[2,] "1.8x0.9"     "1.8" "0.9" NA
[3,] "1.5x1.0"     "1.5" "1.0" NA
[4,] "3.7x3.3x3.2" "3.7" "3.3" "3.2"
[5,] "2.5x2.4x2.3" "2.5" "2.4" "2.3"
[6,] NA            NA    NA    NA
[7,] "2.7x2.6"     "2.7" "2.6" NA
[8,] "6.3x6.0x4.0" "6.3" "6.0" "4.0"
```

KDRG code 읽기

PDF file에서 글자 판독하여 코드 정리하기

```
load(file = "txt0.RData")
txt0 <- txt0[txt0!=""]
txt_mdc <- paste0(txt0[81:1299])
split_mdc <- strsplit(txt_mdc,"\n")
split_mdc <- unlist(split_mdc)
split_mdc <- split_mdc[split_mdc!=""]
sampletext <- "[A-Z]\\d{1,4}\\s{3,4}[a-zA-Z0-9. *(),]{3,54}"
s_mdc <- gregexec(pattern = sampletext, split_mdc)
t_mdc <- regmatches(split_mdc,s_mdc)
ut_mdc <- unlist(t_mdc)
str(ut_mdc)
```

```
 chr [1:39760] "A066    Amoebic brain abscess(G07*)                              " ...
```

```
ut_mdc[3000:3010]
```

```
 [1] "H442     Degenerative myopia                                 "
 [2] "H521    Myopia"
 [3] "H443     Other degenerative disorders of globe               "
```

```
 [4] "H5220   Irregular astigmatism"
 [5] "H444     Hypotony of eye                                     "
 [6] "H5221   Regular astigmatism"
 [7] "H445     Degenerated conditions of globe                     "
 [8] "H5228   Other and unspecified astigmatism"
 [9] "H446     Retained (old) intraocular foreign body, magnetic    "
[10] "H523    Anisometropia and aniseikonia"
[11] "H447     Retained (old) intraocular foreign body, nonmagnetic "
```

2 R에서 사용하는 함수들

2.1 grep

정규표현식을 사용해 패턴에 일치하는 것을 찾아오는 함수

grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE, fixed = FALSE, useBytes = FALSE, invert = FALSE)

pattern : 정규표현식

x : 데이터

해당 인덱스로 찾아주기도 하고

```
text <- c('a','ab','acb','accb','acccb','accccb')
grep('a',text)
```

```
[1] 1 2 3 4 5 6
```

```
grep('acb',text)
```

```
[1] 3
```

해당 값으로 찾아주기도 함

```
grep('ac?b',text, value=T)
```

```
[1] "ab"  "acb"
```

```
grep('ac{2,3}',text,value=T)
```

```
[1] "accb"   "acccb"  "accccb"
```

2.2 gsub

문자열을 바꿔주는 함수

gsub(pattern, replacement, x, ignore.case = FALSE, perl = FALSE, fixed = FALSE, useBytes = FALSE)

2.3 regexpr, gregexpr(regexpr의 global 버전)

grep()과 grepl()의 한계점 보완: 특정 문자 패턴의 일치여부에 대한 정보를 제공하지만 위치 및 정규식의 일치 여부를 알려주지는 않음

`regexpr(pattern, text, ignore.case = FALSE, perl = FALSE,  fixed = FALSE, useBytes = FALSE)`

```
x <- c("Darth Vader: If you only knew the power of the Dark Side.
       Obi-Wan never told you what happend to your father",
       "Luke: He told me enough! It was you who killed him!",
       "Darth Vader: No. I'm your father")
regexpr("you", x) # 각 x의 문자열에서 you가 처음 나타난 위치 및 길이 반환
```

```
[1] 17 33 22
attr(,"match.length")
[1] 3 3 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
```

```
regexpr("father", x) # 패턴을 포함하지 않은 경우 -1 반환
```

```
[1] 111  -1  27
attr(,"match.length")
[1]  6 -1  6
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
```

```
gregexpr("you", x) # 각 x의 문자열에서 you가 나타난 모든 위치 및 길이 반환
```

```
[[1]]
[1]  17  86 106
attr(,"match.length")
[1] 3 3 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE

[[2]]
[1] 33
attr(,"match.length")
[1] 3
```

```
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE

[[3]]
[1] 22
attr(,"match.length")
[1] 3
attr(,"index.type")
[1] "chars"
attr(,"useBytes")
[1] TRUE
```

2.4 stringr 패키지에서 사용하는 함수들

2.4.1 str_detect

2.4.2 str_extract

2.4.3 str_extract_all

2.4.4 str_count

2.4.5 str_subset

2.4.6 str_locate

2.4.7 str_locate_all

2.4.8 str_replace

2.4.9 str_replace_all

2.4.10 str_split