talk07 练习与作业

目录

0.1 练习和作业说明

将相关代码填写入以 “‘{r} “‘ 标志的代码框中，运行并看到正确的结果；

完成后，用工具栏里的”Knit” 按键生成 PDF 文档；

将 PDF 文档改为：`姓名-学号-talk07 作业.pdf`，并提交到老师指定的平台/钉群。

0.2 talk07 内容回顾

1. string basics

- length
- uppercase, lowercase
- unite, separate

- string comparisons, sub string

2. regular expression

- detect patterns
- locate patterns
- extract patterns
- replace patterns

0.3 练习与作业：用户验证

请运行以下命令，验证你的用户名。

如你当前用户名不能体现你的真实姓名，请改为拼音后再运行本作业！

```
Sys.info()[["user"]]
```

```
## [1] "sicheng.wu"
```

```
Sys.getenv("HOME")
```

```
## [1] "/home/vkorpela"
```

0.4 练习与作业 1：字符串操作

0.4.1 用 stringr 包实现以下操作

使用变量：`x <- c('weihua', 'chen');`

1. 每个 element/成员的长度
2. 每个成员首字母大写
3. 取每个成员的前两个字符

4. 合并为一个字符串，用 ‘,’ 间隔
5. 数一下每个成员中元音字母（vowel letter）的数量

```
## 代码写这里，并运行;
library(stringr)
x <- c('weihua', 'chen')

# 每个成员的长度
str_length(x)
```

```
## [1] 6 4
```

```
# 每个成员首字母大写
str_to_title(x)
```

```
## [1] "Weihua" "Chen"
```

```
# 取每个成员的前两个字符
str_sub(x, start = 1, end = 2)
```

```
## [1] "we" "ch"
```

```
# 合并为一个用 “, ” 间隔的字符串
str_c(x, collapse = ", ")
```

```
## [1] "weihua, chen"
```

```
# 数元音字母数量
str_count(x, "[aeiou]")
```

```
## [1] 4 1
```

0.4.2 用 mtcars 变量作练习

1. 筛选出所有的奔驰车（Mercedes-Benz）；
2. 筛选出所有非奔驰车；
3. 处理行名，将其中的品牌与车型分开。比如：Mazda RX4 Wag => 'Mazda', 'RX4 Wag'

```
## 代码写这里，并运行；
library(tidyverse)
```

```
## Warning in system("timedatectl", intern = TRUE): running command 'timedatectl'
## had status 1
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.0      v forcats 0.5.2
## v readr   2.1.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
mtcars.tibble <- as_tibble(mtcars, rownames = "brandtype")

# 筛选出所有的奔驰车
mtcars.tibble %>%
  filter(str_detect(brandtype, "Merc"))
```

```
## # A tibble: 7 x 12
##   brandtype     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##   <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 Merc 240D    24.4     4  147.    62  3.69  3.19  20       1     0     4     2
## 2 Merc 230     22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
## 3 Merc 280     19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
```

```
## 4 Merc 280C    17.8     6  168.   123  3.92  3.44  18.9     1     0     4     4
## 5 Merc 450SE   16.4     8  276.   180  3.07  4.07  17.4     0     0     3     3
## 6 Merc 450SL   17.3     8  276.   180  3.07  3.73  17.6     0     0     3     3
## 7 Merc 450SLC  15.2     8  276.   180  3.07  3.78  18       0     0     3     3
```

```
# 筛选出所有的非奔驰车
mtcars.tibble %>%
  filter(!str_detect(brandtype, "Merc"))
```

```
## # A tibble: 25 x 12
##    brandtype     mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##    <chr>       <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Mazda RX4    21       6   160   110  3.9   2.62  16.5     0     1     4     4
##  2 Mazda RX4 ~  21       6   160   110  3.9   2.88  17.0     0     1     4     4
##  3 Datsun 710   22.8     4   108    93  3.85  2.32  18.6     1     1     4     1
##  4 Hornet 4 D~  21.4     6   258   110  3.08  3.22  19.4     1     0     3     1
##  5 Hornet Spo~  18.7     8   360   175  3.15  3.44  17.0     0     0     3     2
##  6 Valiant      18.1     6   225   105  2.76  3.46  20.2     1     0     3     1
##  7 Duster 360   14.3     8   360   245  3.21  3.57  15.8     0     0     3     4
##  8 Cadillac F~  10.4     8   472   205  2.93  5.25  18.0     0     0     3     4
##  9 Lincoln Co~  10.4     8   460   215  3     5.42  17.8     0     0     3     4
## 10 Chrysler I~  14.7     8   440   230  3.23  5.34  17.4     0     0     3     4
## # ... with 15 more rows
```

```
# 品牌和车型分开
mtcars.tibble %>%
  separate(brandtype, c("brand", "type"), extra = "merge")
```

```
## Warning: Expected 2 pieces. Missing pieces filled with `NA` in 1 rows [6].
```

```
## # A tibble: 32 x 13
##    brand type    mpg   cyl  disp    hp  drat    wt  qsec    vs    am  gear  carb
##    <chr> <chr> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 Mazda RX4    21       6  160    110  3.9   2.62  16.5     0     1     4     4
```

```
##  2 Mazda RX4 ~  21       6  160    110  3.9   2.88  17.0     0     1     4     4
##  3 Dats~ 710    22.8     4  108     93  3.85  2.32  18.6     1     1     4     1
##  4 Horn~ 4 Dr~  21.4     6  258    110  3.08  3.22  19.4     1     0     3     1
##  5 Horn~ Spor~  18.7     8  360    175  3.15  3.44  17.0     0     0     3     2
##  6 Vali~ <NA>   18.1     6  225    105  2.76  3.46  20.2     1     0     3     1
##  7 Dust~ 360    14.3     8  360    245  3.21  3.57  15.8     0     0     3     4
##  8 Merc  240D   24.4     4  147.    62  3.69  3.19  20       1     0     4     2
##  9 Merc  230    22.8     4  141.    95  3.92  3.15  22.9     1     0     4     2
## 10 Merc  280    19.2     6  168.   123  3.92  3.44  18.3     1     0     4     4
## # ... with 22 more rows
```

用 str_c 操作

为下面字符增加前缀和后缀，

`x <- c("abc", NA)`

使其最终结果为：

`"|-abc-|" "|-NA-|"`

```
## 代码写这里，并运行；
x <- c("abc", NA)
str_c("|-", str_replace_na(x), "-|")
```

```
## [1] "|-abc-|" "|-NA-|"
```

0.5 练习与作业 2：regular expression 正则表达式练习

0.5.1 用 starwars 变量作练习

注：需要先导入 tidyverse 包；

1. 选出所有 skin_color 包含为 white 的人，显示其 name, homeworld, species 和 skin_color；注意：有些人的 skin color 可为多个；

2. 打印出所有含有 ar 的名字；不区分大小写；

```
## 代码写这里，并运行；
# 筛选 skin_color
starwars %>%
  filter(str_detect(skin_color, "white")) %>%
  select(name, homeworld, species, skin_color)
```

```
## # A tibble: 7 x 4
##   name        homeworld species  skin_color
##   <chr>       <chr>     <chr>    <chr>
## 1 R2-D2       Naboo     Droid    white, blue
## 2 Darth Vader Tatooine  Human    white
## 3 R5-D4       Tatooine  Droid    white, red
## 4 Gasgano     Troiken   Xexto    white, blue
## 5 Yarael Poof Quermia   Quermian white
## 6 Shaak Ti    Shili     Togruta  red, blue, white
## 7 Grievous    Kalee     Kaleesh  brown, white
```

```
# 打印含有 ar 的名字
starwars %>%
  filter(str_detect(name, "[Aa][Rr]")) %>%
  pull(name)
```

```
##  [1] "Darth Vader"           "Owen Lars"             "Beru Whitesun lars"
##  [4] "Biggs Darklighter"     "Wilhuff Tarkin"        "Ackbar"
##  [7] "Arvel Crynyd"          "Wicket Systri Warrick" "Jar Jar Binks"
## [10] "Roos Tarpals"          "Quarsh Panaka"         "Darth Maul"
## [13] "Ben Quadinaros"        "Yarael Poof"           "Gregar Typho"
## [16] "Cliegg Lars"           "Luminara Unduli"       "Barriss Offee"
## [19] "Tarfful"
```

0.5.2 用下面的 vec 变量作练习

```
vec <- c( "123", "abc", "wei555hua666" );
```

1. 找出含有数字的字符串；
2. 找出数字的位置；如果字符串含有多组数数字，只显示第一组；
3. 找出所有数字的位置；
4. 提取出找到的数字；如果字符串含有多组数数字，只提取第一组；
5. 提取所有的数字；
6. 将数字替换为 666；

```
## 代码写这里，并运行；
vec <- c("123", "abc", "wei555hua666")

# 找出含有数字的字符串
str_subset(vec, "\\d+")
```

```
## [1] "123"          "wei555hua666"
```

```
# 找出第一组数字的位置
str_locate(vec, "\\d+")
```

```
##      start end
## [1,]     1   3
## [2,]    NA  NA
## [3,]     4   6
```

```
# 找出所有数字的位置
str_locate_all(vec, "\\d+")
```

```
## [[1]]
##      start end
## [1,]     1   3
##
```

```
## [[2]]
##      start end
##
## [[3]]
##      start end
## [1,]     4   6
## [2,]    10  12
```

```
# 提取出找到的数字
str_extract(vec, "\\d+")
```

```
## [1] "123" NA    "555"
```

```
# 提取所有的数字
str_extract_all(vec, "\\d+")
```

```
## [[1]]
## [1] "123"
##
## [[2]]
## character(0)
##
## [[3]]
## [1] "555" "666"
```

```
# 将数字替换为 666
str_replace_all(vec, "\\d+", "666")
```

```
## [1] "666"          "abc"          "wei666hua666"
```

0.6 练习与作业 3：探索题

0.6.1 序列分析

用序列：`seq <- "ATCTCGGCGCGCATCGCGTACGCTACTAGC"` 实现以下分析；注：可使用任何包：

1. 得到它的反向互补序列；
2. 计算它的 GC 含量，用百分数表示；
3. 把它拆分成一个个 codon（即三个 nucleotide 形成一个 codon；最后一个长度可以不为 3；

```
## 代码写这里，并运行；
library(scales)
```

```
##
## Attaching package: 'scales'

## The following object is masked from 'package:purrr':
##
##     discard

## The following object is masked from 'package:readr':
##
##     col_factor
```

```
seq <- "ATCTCGGCGCGCATCGCGTACGCTACTAGC"

# 反向互补序列
cmpl_lower = c("A" = "t", "T" = "a", "G" = "c", "C" = "g")
seq %>%
  str_to_upper() %>%
  str_replace_all(cmpl_lower) %>%
  str_to_upper()
```

```
## [1] "TAGAGCCGCGCGTAGCGCATGCGATGATCG"
```

```
# GC 含量
percent(str_count(seq, "[GC]") / str_length(seq))
```

```
## [1] "63%"
```

```
# 拆分为 codon
str_extract_all(seq, "[ATCG]{1,3}")
```

```
## [[1]]
##  [1] "ATC" "TCG" "GCG" "CGC" "ATC" "GCG" "TAC" "GCT" "ACT" "AGC"
```

0.6.2 问答

问：stringr::str_pad 的作用是什么？请举例回答

答：将字符串补足到期望的长度。例如说 str_pad("3.14", 6, side = "right", pad = "0") 是在字符串的右侧补足"0"，使其长度为 6。

0.6.3 提取字符串中的 N 次重复字段

问：如何用正则表达式从字符串中提取任意长度为 2 字符的两次以上重复，比如：1212, abab, tata, 是 12 等的两次重复，898989 则是 89 的 3 次重复，以下面的变量为输入：

```
c( "banana", "coconut", "1232323", "database" )
```

```
## 代码写这里，并运行；
c("banana", "coconut", "1232323", "database") %>%
  str_extract("(..)\\1{1,}")
```

```
## [1] "anan"   "coco"   "232323" NA
```

0.6.4 正则表达式

设计一个正则表达式，可以完整识别所有以下格式的数字

```
123
123.45
0.124
-1.5
-0.2
+1.3
-11
-199.62
```

```
## 代码写这里，并运行;
num.regex = regex("[+-]?\\d+(.\\d+)?")
```