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] "mingyuwang"
```

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

```
## [1] "C:/Users/rhong/Documents"
```

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

```
library("tidyverse")
library("Biostrings")
```

```
## 代码写这里，并运行；
x <- c("weihua", "chen")
## 1. 每个 element/成员的长度
str_length(x)
```

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

```
## 2. 每个成员首字母大写！首字母大写！不是全部大写！
str_to_title(x)
```

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

```
## 3. 取每个成员的前两个字符
str_sub(x, 1, 2)
```

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

```
## 4. 合并为一个字符串，用 ', ' 间隔
str_c(x, collapse = ", ")
```

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

```
## 5. 数一下每个成员中`元音字母` （vowel letter）的数量
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；
mtcars %>%
  # 筛选行名中包含'Mercedes-Benz'的行
  filter(row.names(mtcars) %>% str_detect(pattern = "Merc"))
```

```
##               mpg cyl  disp  hp drat   wt qsec vs am gear carb
## Merc 240D    24.4   4 146.7  62 3.69 3.19 20.0  1  0    4    2
## Merc 230     22.8   4 140.8  95 3.92 3.15 22.9  1  0    4    2
## Merc 280     19.2   6 167.6 123 3.92 3.44 18.3  1  0    4    4
## Merc 280C    17.8   6 167.6 123 3.92 3.44 18.9  1  0    4    4
## Merc 450SE   16.4   8 275.8 180 3.07 4.07 17.4  0  0    3    3
## Merc 450SL   17.3   8 275.8 180 3.07 3.73 17.6  0  0    3    3
## Merc 450SLC  15.2   8 275.8 180 3.07 3.78 18.0  0  0    3    3
```

```
mtcars %>%
  # 筛选行名中不包含'Mercedes-Benz'的行
  filter(!(row.names(mtcars) %>% str_detect(pattern = "Merc")))
```

```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4          21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag      21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710         22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive     21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant            18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360         14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
```

```
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```
mtcars %>%
  # 处理行名，将其中的品牌与车型分开，品牌与车型是以空格分隔的
  mutate(brand = row.names(mtcars) %>% str_extract("^[a-zA-Z0-9]+"),
         model = row.names(mtcars) %>% str_extract("\\s[a-zA-Z0-9-\\s]+"))
```

```
##                    mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360        14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D         24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230          22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280          19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
```

```
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
##                        brand        model
## Mazda RX4              Mazda          RX4
## Mazda RX4 Wag          Mazda      RX4 Wag
## Datsun 710            Datsun          710
## Hornet 4 Drive        Hornet      4 Drive
## Hornet Sportabout     Hornet   Sportabout
## Valiant              Valiant         <NA>
## Duster 360            Duster          360
## Merc 240D               Merc         240D
## Merc 230                Merc          230
## Merc 280                Merc          280
```

```
## Merc 280C               Merc         280C
## Merc 450SE              Merc        450SE
## Merc 450SL              Merc        450SL
## Merc 450SLC             Merc       450SLC
## Cadillac Fleetwood  Cadillac    Fleetwood
## Lincoln Continental  Lincoln  Continental
## Chrysler Imperial   Chrysler     Imperial
## Fiat 128                Fiat          128
## Honda Civic             Honda        Civic
## Toyota Corolla         Toyota      Corolla
## Toyota Corona          Toyota       Corona
## Dodge Challenger        Dodge   Challenger
## AMC Javelin               AMC      Javelin
## Camaro Z28              Camaro          Z28
## Pontiac Firebird      Pontiac     Firebird
## Fiat X1-9                 Fiat         X1-9
## Porsche 914-2         Porsche        914-2
## Lotus Europa            Lotus       Europa
## Ford Pantera L           Ford    Pantera L
## Ferrari Dino          Ferrari         Dino
## Maserati Bora        Maserati         Bora
## Volvo 142E              Volvo        142E
```

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)
# na 值替换为字符串"NA"
```

```
x[is.na(x)] <- "NA"
str_c("|-", 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；
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
```

```
starwars %>%
  # R 正则表达式中，不区分大小写，pattern = '(?i)ar'
  filter(str_detect(name, pattern = "(?i)ar")) %>%
  select(name)
```

```
## # A tibble: 19 x 1
##    name
##    <chr>
##  1 Darth Vader
##  2 Owen Lars
##  3 Beru Whitesun lars
##  4 Biggs Darklighter
##  5 Wilhuff Tarkin
##  6 Ackbar
##  7 Arvel Crynyd
##  8 Wicket Systri Warrick
##  9 Jar Jar Binks
## 10 Roos Tarpals
## 11 Quarsh Panaka
## 12 Darth Maul
## 13 Ben Quadinaros
## 14 Yarael Poof
## 15 Gregar Typho
## 16 Cliegg Lars
## 17 Luminara Unduli
## 18 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
a <- vec %>%
  str_detect("[0-9]") %>%
  which()
vec[a]
```

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

```
# 找出数字的位置
vec %>% str_locate("[0-9]+")
```

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

```
# 找出所有数字的位置
vec %>% str_locate_all("[0-9]+")
```

```
## [[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
vec %>% str_extract("[0-9]+")
```

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

```
# 提取所有的数字
vec %>% str_extract_all("[0-9]+")
```

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

```
# 将数字替换为 666
vec %>% str_replace("[0-9]+", "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("Biostrings")
seq <- "ATCTCGGCGCGCATCGCGTACGCTACTAGC"

# 得到它的反向互补序列
seq %>%
  DNAStringSet() %>%
  reverseComplement() %>%
  as.character()
```

```
## [1] "GCTAGTAGCGTACGCGATGCGCGCCGAGAT"
```

0.6.2 问答

问：`stringr::str_pad` 的作用是什么？请举例回答

答：`stringr::str_pad` 用于字符串的填充，可以指定填充的位置，填充的字符，以及填充的长度。

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" )
```

```r
## 代码写这里，并运行；
vec <- c("banana", "coconut", "1232323", "database")
# 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
vec %>% str_extract_all("([a-z]{2})\\1+") %>% unlist()
```

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

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
```

```r
## 代码写这里，并运行；
vec <- c("123", "123.45", "0.124", "-1.5", "-0.2", "+1.3", "-11", "-199.62")
# 设计一个正则表达式，可以完整识别上面所有格式的数字
pattern <- "[-+]?[0-9]+(\\.[0-9]+)?"

# 检查是否完整识别
vec %>% str_detect(pattern)
```

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