

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

Работа со строками в R

Г. Мороз

Как получить строку?

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

○ a <- "the quick brown fox jumps over the lazy dog"

○ a <- 'the quick brown fox jumps over the lazy dog'

○ a <- "the quick 'brown' fox jumps over the lazy dog"

○ a <- 'the quick "brown" fox jumps over the lazy dog'

○ b <- ""

○ c <- character(3)

создает пустые строки

```
[1] "" "" ""
```

○ d <- as.character(3)

превращает в строку

```
[1] "3"
```

○ d <- as.character(c(4:9))

вектор в строку

```
[1] "4" "5" "6" "7" "8" "9"
```

○ letters[3:8], LETTERS[3:8]

○ data.frame(letters[3:8], letters[8:3], stringsAsFactors = F)

Чтение файлов?

- `z <- read.table("a.csv sep = "...")` # текст и разделители
- `y <- read.csv("b.txt")` # разделитель ","
- `x <- read.csv2("c.csv")` # разделитель ";"
- `w <- read.delim("d.txt")` # разделитель "\t"
- `v <- readLines("e.txt")` # чтение "голового" текста

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

Соединение строк: paste()

- `paste("Школа", "Лингвистики")`
`[1] "Школа Лингвистики"`
- `paste(9, 7, 8, sep = "-")` # другой разделитель
`[1] "9-7-8"`
- `paste(LETTERS[1:5], letters[1:5], sep = "")` # векторы
`[1] "Aa" "Bb" "Cc" "Dd" "Ee"`
- `paste("F", letters[1:5], sep = "")` # векторы разной длины
`[1] "Fa" "Fb" "Fc" "Fd" "Fe"`
- `paste("F", letters[1:5], sep = "", collapse = " ")` # в один вектор
`[1] "Fa Fb Fc Fd Fe"`
- `paste("F", letters[1:5], sep = "", collapse = "-")` # другой разделитель
`[1] "Fa-Fb-Fc-Fd-Fe"`
- `paste(c("F", letters[1:5]), collapse = "-")` # развекторизовать
`[1] "F-a-b-c-d-e"`

Соединение строк: library(stringr)

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

- `str_c("Школа", "Лингвистики")`
`[1] "ШколаЛингвистики"`
- `str_c("Школа", "Лингвистики", sep = " ")` # разделитель
`[1] "Школа Лингвистики"`
- `str_c(LETTERS[1:5], letters[1:5], sep = "")` # векторы
`[1] "Aa" "Bb" "Cc" "Dd" "Ee"`
- `str_c("F", letters[1:5], sep = "")` # векторы разной длины
`[1] "Fa" "Fb" "Fc" "Fd" "Fe"`
- `str_c("F", letters[1:5], sep = "", collapse = " ")` # в один вектор
`[1] "Fa Fb Fc Fd Fe"`
- `str_c("F", letters[1:5], sep = "", collapse = "-")` # другой разделитель
`[1] "Fa-Fb-Fc-Fd-Fe"`
- `str_c(c("F", letters[1:5]), collapse = "-")` # развекторизовать
`[1] "F-a-b-c-d-e"`

Соединение строк: paste() vs. str_c()

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

paste нативная функция

paste paste("1", "3")

```
[1] "1 3"
```

paste paste("f", NULL, "", "g", sep = "-")

```
[1] "f---g"
```

str_c library(stringr)

str_c str_c("1", "3")

```
[1] "13"
```

str_c str_c("f", NULL, "", "g", sep = "-")

```
[1] "f--g"
```

Количество символов

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

```
a <- "the quick brown fox jumps over the lazy dog"
```

- `nchar(a)`

```
[1] 43
```

- `library(stringr); str_length(a)`

```
[1] 43
```

```
b <- c("the", "quick", "brown", "fox")
```

- `nchar(b)`

```
[1] 3 5 5 3
```

- `str_length(b)`

```
[1] 3 5 5 3
```

```
c <- c("the", "quick", NULL, "brown", "", "fox")
```

- `nchar(c)`

```
[1] 3 5 5 0 3
```

- `str_length(c)`

```
[1] 3 5 5 0 3
```

Изменение регистра

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

`a <- "the QuiCk bRown fOx juMps ovEr tHe laZy dOg"`

- `tolower(a)`

`[1] "the quick brown fox jumps over the lazy dog"`

- `toupper(a)`

`[1] "THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG"`

`b <- "любЯ, сЪешЬ щипЦы, — взДохнёт мэр, — кайф Жгуч"`

- `tolower(b)`

`[1] "любя, съешь щипцы, — вздохнёт мэр, — кайф жгуч"`

- `toupper(b)`

`[1] "ЛЮБЯ, СЪЕШЬ ЩИПЦЫ, — ВЗДОХНЁТ МЭР, — КАЙФ ЖГУЧ"`

Выделение подстроки: substr()

```
a <- "the quick brown fox"
```

```
b <- c("the", "quick", "brown", "fox")
```

- substr(a, 2, 3)

второй и третий символ

```
[1] "he"
```

- substr(b, 2, 3)

второй и третий символ

```
[1] "he" "ui" "ro" "ox"
```

- substr(b, 2, 2) <- "."; b

заменяет второй символ точкой

```
[1] "t.e" "q.ick" "b.own" "f.x"
```

- substr(b, 2, 2) <- c("1", "2"); b # заменяет второй символ вектором

```
[1] "t1e" "q2ick" "b1own" "f2x"
```

Выделение подстроки: substring()

```
a <- "the quick brown fox"
```

```
b <- c("the", "quick", "brown", "fox")
```

- `substring(b, 2, 3)` # второй и третий символ

```
[1] "he" "ui" "ro" "ox"
```

- `substring(b, 2, 2) <- "."; b` # заменяет второй символ точкой

```
[1] "t.e" "q.ick" "b.own" "f.x"
```

- `substring(b, 2, 2) <- c("1", "2"); b` # заменяет второй символ вектором

```
[1] "t1e" "q2ick" "b1own" "f2x"
```

- `substring(b, 2) <- c("1", "22", "333"); b` # заменяет второй символ вектором

```
[1] "t1e" "q22ck" "b333n" "f1x"
```

- `substring(a, 1:19, 1:19)` # извлекает каждую букву

```
[1] "t" "h" "e" " " "q" "u" "i" "c" "k" " " "b" "r" "o" "w" "n" " " "f" "o" "x"
```

Выделение подстроки: library(stringr)

```
a <- "the quick brown fox"
```

```
b <- c("the", "quick", "brown", "fox")
```

- `str_sub(b, 2, 3)` # второй и третий символ

```
[1] "he" "ui" "ro" "ox"
```

- `str_sub(b, 2, 2) <- "."; b` # заменяет второй символ точкой

```
[1] "t.e" "q.ick" "b.own" "f.x"
```

- `str_sub(b, 2, 2) <- ".;"; b` # заменяет второй символ другими

```
[1] "t.;.e" "q.;.ick" "b.;.own" "f.;.x"
```

- `str_sub(b, 2) <- c("1", "2"); b` # заменяет второй символ вектором

```
[1] "t1e" "q2ick" "b1own" "f2x"
```

- `str_sub(a, 1:19, 1:19)` # извлекает каждую букву

```
[1] "t" "h" "e" " " "q" "u" "i" "c" "k" " " "b" "r" "o" "w" "n" " " "f" "o" "x"
```

- `str_sub(b, -1, -1)` # последний символ

```
[1] "e" "k" "n" "x"
```

Выделение подстроки: library(stringr)

```
a <- c("the quick brown fox jumps over")
```

```
b <- c("jumps over the lazy dog")
```

- `word(a, 2)` # второе слово
[1] "quick"

- `word(c(b, a), 4)` # четвертое слово каждой строки
[1] "lazy" "fox"

- `word(c(b, a), -3)` # третье слово с конца каждой строки
[1] "the" "fox"

- `word(c(b, a), 2, 4)` # со второго по четвертое слово каждой строки
[1] "over the lazy" "quick brown fox"

- `word(c(b, a), -2, -1)` # последние два слова каждой строки
[1] "lazy dog" "jumps over"

Поиск вектора по фрагменту: grep()

```
a <- c("the quick", "brown", "fox", "jumps", "over")
```

```
b <- c("jumps", "over", "the", "lazy", "dog")
```

○ `grep("the", c(a, b))` номер элемента, содержащего подстроку
[1] 1 8

○ `grep("the", c(a, b), value = T)` элемент, содержащий подстроку
[1] "the quick" "the"

○ `grep("the", c(a, b), invert = T)` номер элемента, не содержащего подстроку
[1] 2 3 4 5 6 7 9 10

○ `grep("the", c(a, b), invert = T, value = T)` элемент, не содержащий подстроку
[1] "brown" "fox" "jumps" "over" "jumps" "over" "lazy" "dog"

○ `grepl("the", c(a, b))` элемент, содержащий подстроку
[1] TRUE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE

Поиск вектора по фрагменту: library("stringr")

```
a <- c("the quick", "brown", "fox", "jumps", "over")
```

```
b <- c("jumps", "over", "the", "lazy", "dog")
```

- `str_detect("the", c(a, b))` элемент, содержащий подстроку
[1] TRUE FALSE FALSE FALSE FALSE FALSE FALSE TRUE FALSE FALSE
- `c(a,b)[str_detect("the c(a, b))]` элемент, содержащий подстроку
[1] "the" "the"

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

Замена фрагмента

```
a <- c("the quick brown fox jumps over")
```

```
b <- c("jumps over the lazy dog")
```

- `sub("o", "_", c(a, b))` # одно вхождение
`[1] "the quick br_wn fox jumps over" "jumps _ver the lazy dog"`

- `gsub("o", "_", c(a, b))` # все вхождения
`[1] "the quick br_wn f_x jumps _ver" "jumps _ver the lazy d_g"`

- `library("stringr"); str_replace("o", "_", c(a, b))` # одно вхождение
`[1] "the quick br_wn fox jumps over" "jumps _ver the lazy dog"`

- `library("stringr"); str_replace_all("o", "_", c(a, b))` # все вхождения
`[1] "the quick br_wn f_x jumps _ver" "jumps _ver the lazy d_g"`

Разбиение

```
a <- c("the quick brown fox jumps over")  
b <- c("jumps over the lazy dog")
```

- `strsplit(c(a,b), " ")` # возвращает список слов
- `library("stringr"); str_split(c(a,b), " ")` # возвращает список слов

Вектор уникальных строк

```
a <- c("the", "quick", "brown", "fox", "jumps", "over")
```

```
b <- c("jumps", "over", "the", "lazy", "dog")
```

- `union(a, b)`

```
[1] "the" "quick" "brown" "fox" "jumps" "over" "lazy" "dog"
```

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

Пересечение векторов строк

```
a <- c("the", "quick", "brown", "fox", "jumps", "over")
```

```
b <- c("jumps", "over", "the", "lazy", "dog")
```

- intersect(a, b)

```
[1] "the" "jumps" "over"
```

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

Разность векторов строк

```
a <- c("the", "quick", "brown", "fox", "jumps", "over")
```

```
b <- c("jumps", "over", "the", "lazy", "dog")
```

- `setdiff(a, b)`

```
[1] "quick" "brown" "fox"
```

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

Сравнение векторов строк

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

```
a <- c("the", "quick", "brown", "fox", "jumps", "over")
```

```
b <- c("jumps", "over", "the", "lazy", "dog")
```

- `setequal(a, b)`

```
[1] FALSE
```

```
a <- c("the", "quick", "brown", "fox", "jumps", "over")
```

```
c <- c("brown", "fox", "jumps", "over", "the", "quick")
```

- `setequal(c, a)`

```
[1] TRUE
```

```
a <- c("the", "quick", "brown", "fox", "jumps", "over")
```

```
c <- c("brown", "fox", "jumps", "over", "the", "quick")
```

- `identical(c, a)`

```
[1] FALSE
```

Является ли вектор подмножеством другого?

```
a <- c("the", "quick", "brown", "fox")
```

```
b <- "the"
```

```
c <- "lazy"
```

```
d <- "dog"
```

- `is.element(b, a)`

```
[1] TRUE
```

- `is.element(c, a)`

```
[1] FALSE
```

- `is.element(c(b, c, d), a)`

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

Экранирование метасимволов

В R для экранирования метасимволов используется двойной слэш (а для экранирования слэша аж три слэша):

\\.

\\\$

*

\\+

\\?

\\|

\\\\\\

\\^

\\[

\\]

\\{

\\}

\\(

\\)

Классы знаков

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

`\\d`
`\\D`
`\\s`
`\\S`
`\\w`
`\\W`

цифры
не цифры
пробел
не пробел
слово
не слово

- `gsub("\\d", "\\.", "2016 год")`
`[1] ".... год"`
- `gsub("\\D", "\\.", "2016 год")`
`[1] "2016...."`
- `gsub("\\s", "\\.", "2016 год")`
`[1] "2016.год"`

- `gsub("\\S", "\\.", "2016 год")`
`[1] ".... ..."`
- `gsub("\\w", "\\.", "2016 год")`
`[1] ".... ..."`
- `gsub("\\W", "\\.", "2016 год")`
`[1] "2016.год"`

Наборы символов

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

- `gsub("[0-9]", "\\.", "2016 год")`
`[1] ".... год"`

- `gsub("[a-z]", "\\.", "2016 year")`
`[1] "2016"`

- `gsub("[a-я]", "\\.", "2016 год")`
`[1] "2016 ..."`

- `gsub("[A-Z]", "\\.", "2016 yEar")`
`[1] "2016 y.ar"`

- `gsub("[А-Я]", "\\.", "2016 rОД")`
`[1] "2016 r.."`

- `gsub("[a-zA-Z]", "\\.", "yEar")`
`[1] "...."`

- `gsub("[^a-я]", "\\.", "2016 год")`
`[1] "....год"`

Квантификаторы

создание

чтение

манипуляции

соединение

количество символов

регистр букв

подстроки

поиск

замена

разбиение

векторы строк

объединение

пересечение

разность

сравнение

подмножество

регулярки

метасимволы

наборы символов

квантификация

```
a <- c("тара", "коса", "шоссе", "касса", "масса")
```

- o `gsub("cc?", "\\.", a)`

ноль или один раз

```
[1] "тара" "ко.а" "шо.е" "ка.а" "ма.а"
```

- o `gsub("cc*", "\\.", a)`

ноль и более раз

```
[1] "тара" "ко.а" "шо.е" "ка.а" "ма.а"
```

- o `gsub("cc+", "\\.", a)`

один и более раз

```
[1] "тара" "коса" "шо.е" "ка.а" "ма.а"
```

- o `gsub("a.a", "\\.", a)`

любой символ

```
[1] "т." "коса" "шоссе" "касса" "масса"
```

- o `gsub("a.*a", "\\.", a)`

любой символ ноль и более раз

```
[1] "т." "коса" "шоссе" "к." "м."
```

- o `gsub("c{2}", "\\.", a)`

два раза

```
[1] "тара" "коса" "шо.е" "ка.а" "ма.а"
```

- o `gsub("c{1,}", "\\.", a)`

один и более раз

```
[1] "тара" "ко.а" "шо.е" "ка.а" "ма.а"
```

- o `gsub("c{1,2}", "\\.", a)`

от одного до двух раз

```
[1] "тара" "ко.а" "шо.е" "ка.а" "ма.а"
```

Пробел: `c{1,2}` — правильно, `c{1, 2}` — неправильно.

Спасибо за внимание!

Пишите письма
agricolamz@gmail.com