String Manipulation

String é um tipo de dado que no R também é conhecido como vetor de caracteres, ou seja é uma lista ordenada que podemos acessar através de índices. O conteúdo de uma string é delimitado por aspas "".

Acessando um vetor de números

```
num <- 1:5

num[3]
```

```
## [1] 3
```

Acessando um vetor de String

```
texto <- "Aprendendo Strings"

texto[2]
```

```
## [1] NA
```

Acima o resultado retornou nulo, pois apesar de sabermos que tem duas palavras na sentença acima, para o R contém um único elemento. Para acessar parte desse elemento, precisamos da função `substr`.

```
substr(texto, 1, 10)
```

```
## [1] "Aprendendo"
```

Outra alternativa é com a função `str_sub` presente na biblioteca `stringr`.

```
library(stringr)
str_sub(texto, 1, 10)
```

```
## [1] "Aprendendo"
```

Trocando nome de colunas do dataset (Letras Maiúsculas e Minúsculas)

Vamos criar um dataframe e conferir o nome das colunas.

```
df <- data.frame(Col1 = c(1,2,3),
                 Col2 = c(4,5,6),
                 Col3 = c(7,8,9))

names(df)
```

```
## [1] "Col1" "Col2" "Col3"
```

tolower

Uma boa prática ao trabalhar com um dataset, para evitar dores de cabeça, é transformar todas as letras em minúscula.

```
names(df) <- tolower(names(df))
names(df)
```

```
## [1] "col1" "col2" "col3"
```

toupper

Alternativamente, a função toupper vai converter todas as letras para maiúsculo. Novamente na biblioteca `stringr` temos as funções `str_tolower` e `str_toupper` que são equivalentes.

```
names(df) <- str_to_upper(names(df))
names(df)
```

```
## [1] "COL1" "COL2" "COL3"
```

title e setence

Para alterar strings que possuem um texto longo, temos a opção title que a cada palavra deixa a primeira letra maiúscula.

```
texto = "Isto é um tExTo. tEnte mAnipuLa-Lo"

str_to_title(texto)
```

```
## [1] "Isto É Um Texto. Tente Manipula-Lo"
```

Ou a opção setence que apenas altera a primeira letra.

```
str_to_sentence(texto)
```

```
## [1] "Isto é um texto. tente manipula-lo"
```

Trabalhando com espaços nas strings

trim

Em alguns casos, podemos receber bases ou ao realizar webscrapping, se deparar com strings que possuem espaços desnecessários e precisamos realizar a devida limpeza. O processo de retirar esses espaços é conhecido como trim.

```
texto_espaco <- "   Texto com espaços   "

str_trim(texto_espaco)
```

```
## [1] "Texto com espaços"
```

Por padrão a função remove de ambos os lados, mas é possível escolher apenas um determinado lado.

```
str_trim(texto_espaco, side = "left")
```

```
## [1] "Texto com espaços   "
```

padding

Em outras situações, é possível que nosso objetivo seja o oposto, e desejamos adicionar um espaço ao texto. A opção width leva em consideração a quantidade de caracteres já incluído o texto presente antes da alteração.

```
texto_padding <- "Teste"
str_pad(texto_padding, width=20, side="both")
```

```
## [1] "       Teste        "
```

Ou no lugar do espaço podemos colocar outro caracter.

```
str_pad(texto_padding, width=20, side="both", pad="-")
```

```
## [1] "-------Teste--------"
```

Tratar sentença com múltiplas palavras

O R entende que a frase abaixo é um único elemento, mas para os olhos humanos sabemos que existem vários elementos separados por espaços. Sendo assim a função `str_split` nos ajuda a separar de forma a podermos acessar mais facilmente cada elemento da frase.

split

```
frase <- "Isto é uma frase a ser trabalhada no RStudio"

palavras <- str_split(frase,  " ")[[1]]

palavras[1:4]
```

```
## [1] "Isto"  "é"     "uma"   "frase"
```

Como o resultado da função é uma lista, e queremos acessar o vetor em si, é necessário utilizar conchetes duplos para acessar os elementos.

concatenate

Indo no caminho contrário, podemos pegar elementos separados e concatenar usando um delimitador, por exemplo o espaço para formar uma frase.

```r
palavras <- c("Isto", "é", "uma", "frase",".")

frase <- str_c(palavras, collapse=" ")
frase
```

```
## [1] "Isto é uma frase ."
```

Expressões Regulares (RegEx)

É um mecanismo poderoso e complexo que pode ser usado de diversas formas. Aqui vamos abordar apenas o básico que é utilizar para realizar buscas por determinados caracteres ou padrões.

```r
df <- data.frame(
  nome = c("fulano","ciclano","beltrano","marciano"),
  nota = c(1,5,3,5),
  comentario = c("Péssimo","Incrível","Regular",
                 "Muito bom, incrível!!! incrível mesmo"))
df
```

```
##       nome nota                                comentario
## 1   fulano    1                                  Péssimo
## 2  ciclano    5                                 Incrível
## 3 beltrano    3                                  Regular
## 4 marciano    5 Muito bom, incrível!!! incrível mesmo
```

detect

Para buscar um termo espec[ifico, podemos usar o `str_detect`, no entanto o resultado apenas informa se encontrou ou não.

```
str_detect(df[["comentario"]],"Incrível")
```

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

No exemplo acima, apenas encontrou o termo “Incrível” em uma sentença, isso se dá, pois uma delas estava com letra minúscula. Podemos usar o `str_replace` para mudar.

replace

```
df[["comentario"]] <- df[["comentario"]] %>%
                        str_replace(pattern="incrível",
                                    replacement= "Incrível")
df
```

```
##       nome nota                                comentario
## 1   fulano    1                                   Péssimo
## 2  ciclano    5                                  Incrível
## 3 beltrano    3                                   Regular
## 4 marciano    5 Muito bom, Incrível!!! incrível mesmo
```

```
str_detect(df[["comentario"]],"Incrível")
```

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

Como você pode nostar acima, o termo “incrível” foi substituído por “Incrível” apenas na primeira aparição, na segunda permanceu como estava. Para corrigir isso podemos usar o `str_replace_all`, ele vai buscar todas as aparições do respectivo termo.

```
df[["comentario"]] <- df[["comentario"]] %>%
                        str_replace_all(pattern="incrível",
                                        replacement= "Incrível")
df
```

```
##       nome nota                                comentario
## 1   fulano    1                                  Péssimo
## 2  ciclano    5                                Incrível
## 3 beltrano    3                                  Regular
## 4 marciano    5 Muito bom, Incrível!!! Incrível mesmo
```

```
str_detect(df[["comentario"]],"Incrível")
```

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