

Foundations of R Software

Lecture 39

Strings – Display and Formatting

:::

Manipulations with Strings and Alphabets

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Operations with Strings: Counting characters

nchar takes a character vector as an argument and returns a vector whose elements contain the sizes of the corresponding elements of **x**.

nzchar is a fast way to find out if elements of a character vector are non-empty strings.

Operations with Strings: Counting characters

Usage

```
nchar(x, type = "chars", allowNA = FALSE,  
keepNA = NA)
```

```
nzchar(x, keepNA = FALSE)
```

Arguments

x character vector, or a vector to be coerced to a character vector.

type character string: partial matching to one of `c("bytes", "chars", "width")`.

keepNA logical: should **NA** be returned when **x** is **NA**?

Operations with Strings: Counting characters

There are a variety of commands that can be used for strings.

Example:

Count of number of characters:

```
> x = "R course 24.07.2022"
```

```
> y = "Number of participants: 25"
```

```
> nchar(x) #Count the Number of Characters in x  
[1] 19
```

```
> nchar(y) #Count the Number of Characters in y  
[1] 26
```

Operations with Strings: Counting characters

Example:

The `nzchar(x)` function takes the character vector `x` as a parameter. Its output is either `TRUE` or `FALSE`.

```
> x = "R course 24.07.2022"
```

```
> y = "Number of participants: 25"
```

```
> nzchar(x)  
[1] TRUE
```

```
> nzchar(y)  
[1] TRUE
```

Operations with Strings: Counting characters

```
R Console
> x = "R course 24.07.2022"
> x
[1] "R course 24.07.2022"
> y = "Number of participants: 25"
> y
[1] "Number of participants: 25"
> nchar(x)
[1] 19
> nchar(y)
[1] 26
>
> nzchar(x)
[1] TRUE
> nzchar(y)
[1] TRUE
> |
```

Operations with Strings: Counting characters

Example: `nzchar(x)`

```
> x = c("Apple", "Banana", "Cake")
```

```
> x
```

```
[1] "Apple"  "Banana" "Cake"
```

```
> nzchar(x)
```

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

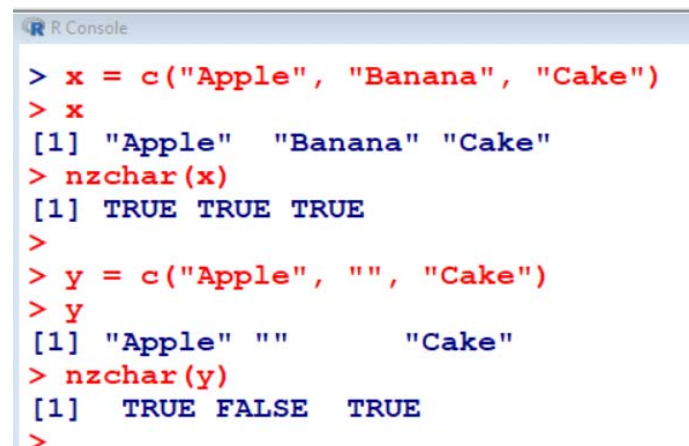
```
> y = c("Apple", "", "Cake")
```

```
> y
```

```
[1] "Apple" ""          "Cake"
```

```
> nzchar(y)
```

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



```
R Console
> x = c("Apple", "Banana", "Cake")
> x
[1] "Apple"  "Banana" "Cake"
> nzchar(x)
[1] TRUE TRUE TRUE
>
> y = c("Apple", "", "Cake")
> y
[1] "Apple" ""          "Cake"
> nzchar(y)
[1] TRUE FALSE TRUE
>
```

Operations with Strings: Counting characters

Example:

```
> x = c("Apple", "Banana", "Cake")
```

```
> nchar(x)
```

```
[1] 5 6 4
```

```
> y = c(2, 4, 6)
```

```
> nchar(y)
```

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

```
> z = c(11, 222, 3333)
```

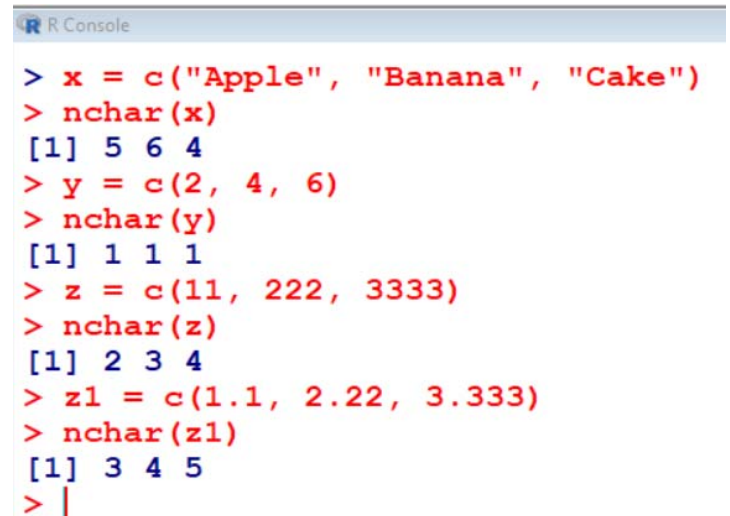
```
> nchar(z)
```

```
[1] 2 3 4
```

```
> z1 = c(1.1, 2.22, 3.333)
```

```
> nchar(z1)
```

```
[1] 3 4 5
```



```
R Console
> x = c("Apple", "Banana", "Cake")
> nchar(x)
[1] 5 6 4
> y = c(2, 4, 6)
> nchar(y)
[1] 1 1 1
> z = c(11, 222, 3333)
> nchar(z)
[1] 2 3 4
> z1 = c(1.1, 2.22, 3.333)
> nchar(z1)
[1] 3 4 5
> |
```

Operations with Strings: Lower and upper cases

`tolower(x)` and `toupper(x)` Functions:

`tolower(x)` and `toupper(x)` convert upper-case characters in a character vector to lower-case, or vice versa.

Non-alphabetic characters are left unchanged.

Operations with Strings: Lower and upper cases

`tolower(x)` and `toupper(x)` Functions:

Examples:

```
> x = "R course will start from 24.07.2022"
```

```
> toupper(x)
```

```
[1] "R COURSE WILL START FROM 24.07.2022"
```

```
> z = "INDIAN INSTITUTE OF TECHNOLOGY"
```

```
> tolower(z)
```

```
[1] "indian institute of technology"
```

Operations with Strings: Lower and upper cases

```
R Console
> x = "R course will start from 24.07.2022"
> x
[1] "R course will start from 24.07.2022"
> toupper(x)
[1] "R COURSE WILL START FROM 24.07.2022"
>
> z = "INDIAN INSTITUTE OF TECHNOLOGY"
> z
[1] "INDIAN INSTITUTE OF TECHNOLOGY"
> tolower(z)
[1] "indian institute of technology"
> |
```