Introduction to R Software

Swayam Prabha

Lecture 27

Splitting and Substitution in Strings

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Slides can be downloaded from

http://home.iitk.ac.in/~shalab/sp

Operations with Strings

Command `strsplit`, split the elements of a character vector.

"Split" can be a single character, or a character string:

Usage

```
strsplit(x, split, fixed = FALSE, …)
```

Arguments

`x` character vector, each element of which is to be split.

`split` character vector containing regular expression(s) (unless fixed = `TRUE`) to use for splitting.

Operations with Strings

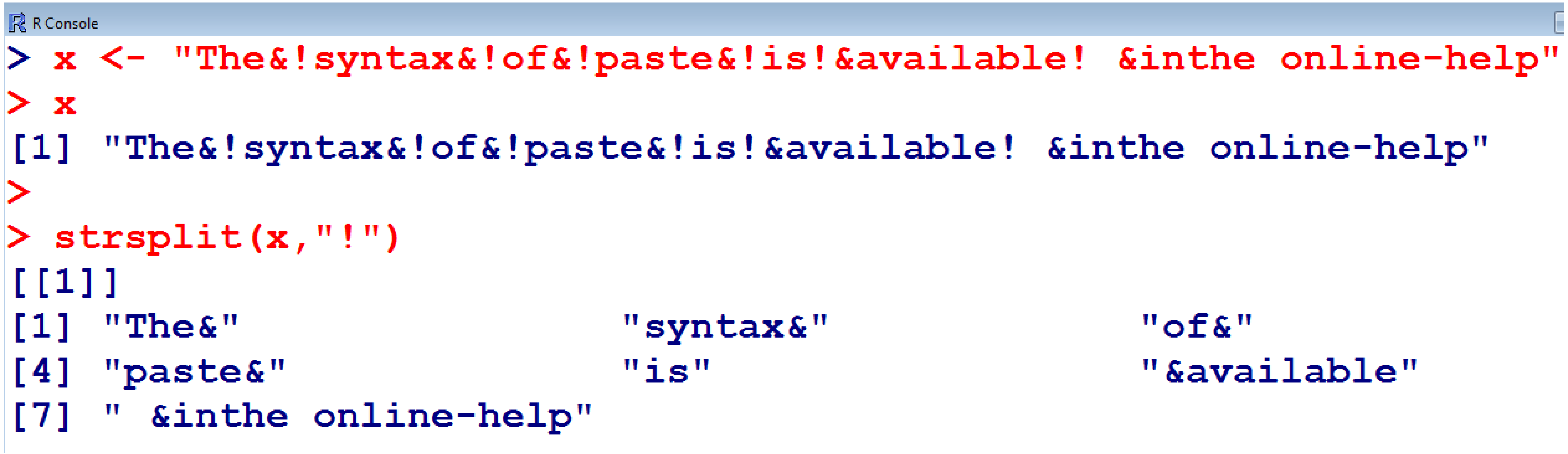
- ❑ With a command `strsplit`, we can split a string in pieces.

```
> x <- "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> x
[1] "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> strsplit(x,"!")
[[1]]
[1] "The&"           "syntax&"    "of&"
[4] "paste&"         "is"         "&available"
[7] " &inthe online-help"
```

Operations with Strings



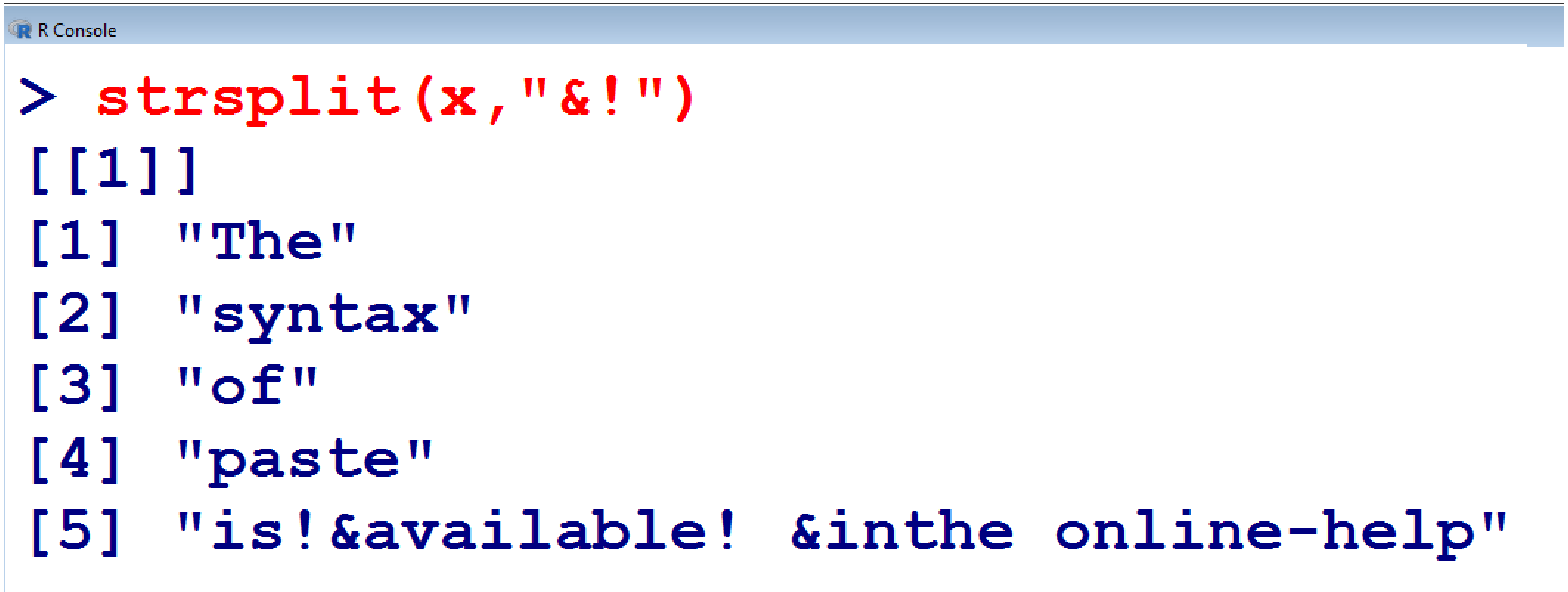
```
> x <- "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
>
> strsplit(x,"!")
[[1]]
[1] "The&"                "syntax&"             "of&"
[4] "paste&"              "is"                  "&available"
[7] " &inthe online-help"
```

Operations with Strings

```
> x <- "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> strsplit(x,"&!")
[[1]]
[1] "The"           "syntax&"
[3] "of"            "paste"
[5] "is!&available!&in the online-help"
```

Operations with Strings



```
> strsplit(x,"&!")
[[1]]
[1] "The"
[2] "syntax"
[3] "of"
[4] "paste"
[5] "is!&available! &inthe online-help"
```

Operations with Strings

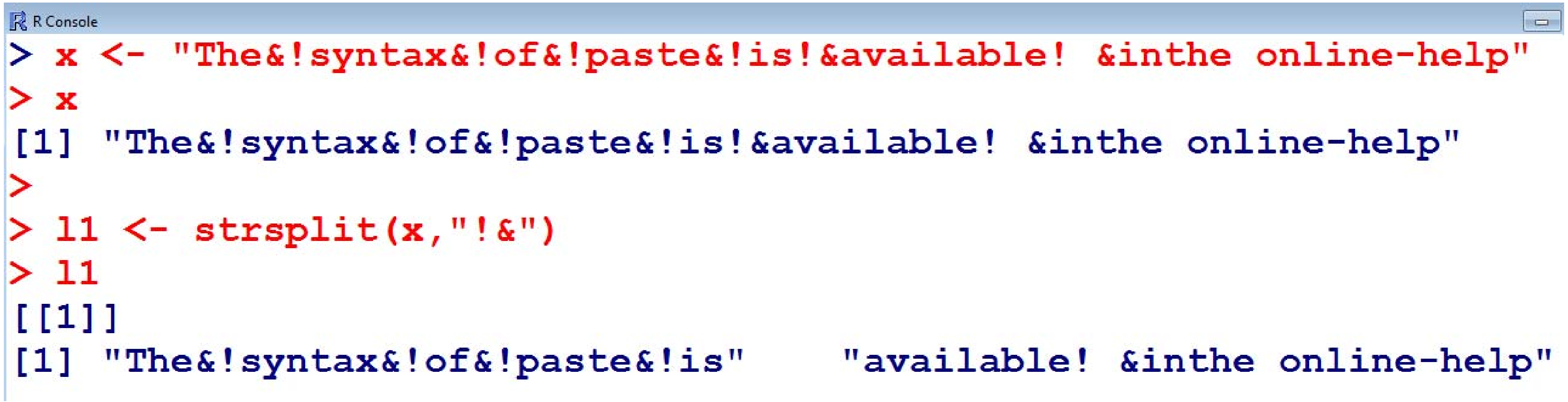
```
> x <- "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> x
[1] "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> l1 <- strsplit(x,"!&")

> l1
[[1]]
[1] "The&!syntax&!of&!paste&!is"    "available!
&inthe online-help"
```

Operations with Strings



```
> x <- "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
>
> l1 <- strsplit(x,"!&")
> l1
[[1]]
[1] "The&!syntax&!of&!paste&!is"    "available! &inthe online-help"
```

Operations with Strings

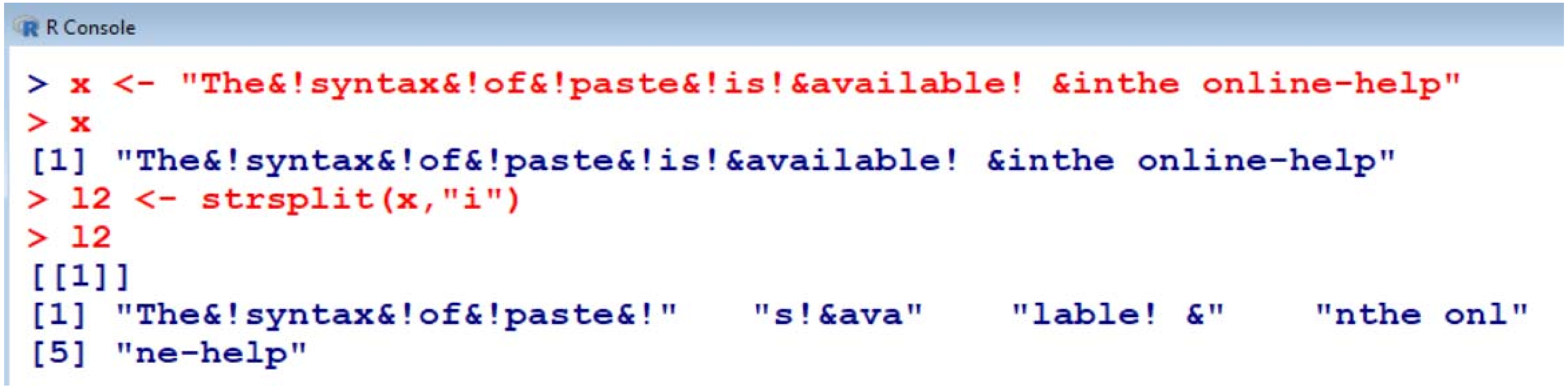
```
> x <- "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> x
[1] "The&!syntax&!of&!paste&!is!&available!
&inthe online-help"

> l2 <- strsplit(x,"i")

> l2
[[1]]
[1] "The&!syntax&!of&!paste&!"    "s!&ava"
"lable! &"    "nthe onl"
[5] "ne-help"
```

Operations with Strings



```
> x <- "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> x
[1] "The&!syntax&!of&!paste&!is!&available! &inthe online-help"
> l2 <- strsplit(x,"i")
> l2
[[1]]
[1] "The&!syntax&!of&!paste&!"   "s!&ava"    "lable! &"    "nthe onl"
[5] "ne-help"
```

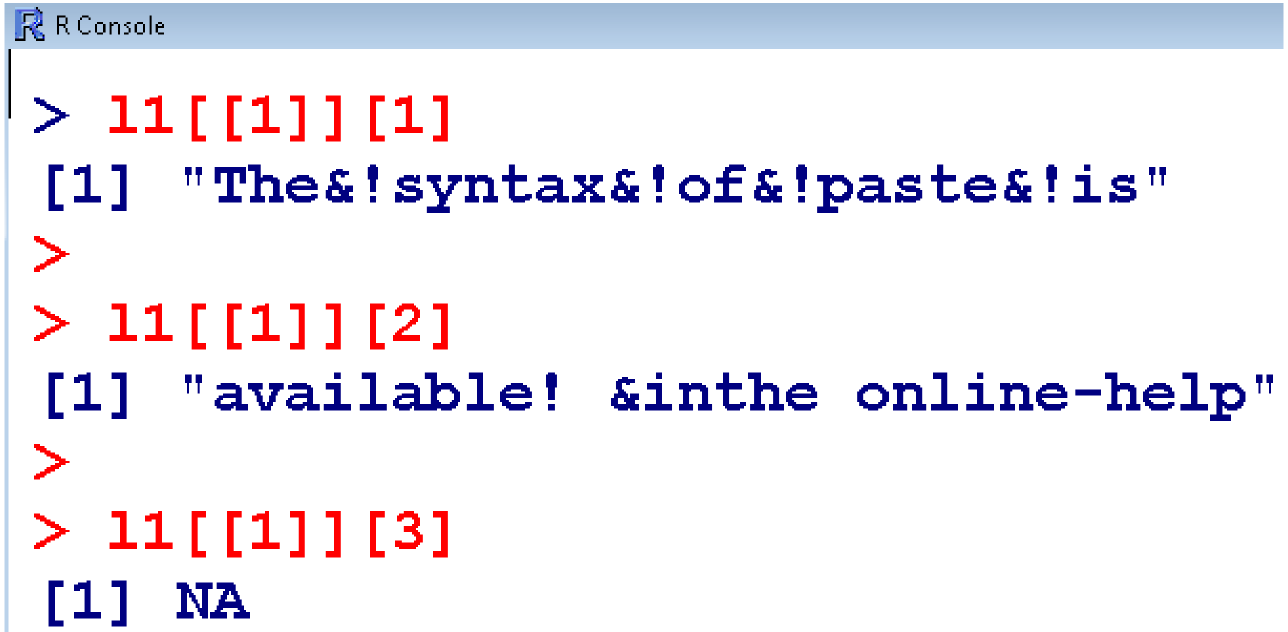
Operations with Strings

Notice the access to single components:

```
> l1[[1]][1]
[1] "The&!syntax&!of&!paste&!is"

> l1[[1]][2]
[1] "available! &inthe online-help"

> l1[[1]][3]
[1] NA
```



```
> l1[[1]][1]
[1] "The&!syntax&!of&!paste&!is"
>
> l1[[1]][2]
[1] "available! &inthe online-help"
>
> l1[[1]][3]
[1] NA
```

Operations with Strings

`sub` and `gsub` Functions:

Within a string, we want to replace one substring with another.

Use `sub` and `gsub` to replace the first instance of a substring:

```
sub(old, new, string)
```

The `sub` function finds the first instance of the old substring within string and replaces it with the new substring.

`gsub` does the same thing, but it replaces all instances of the substring (a global replace), not just the first.

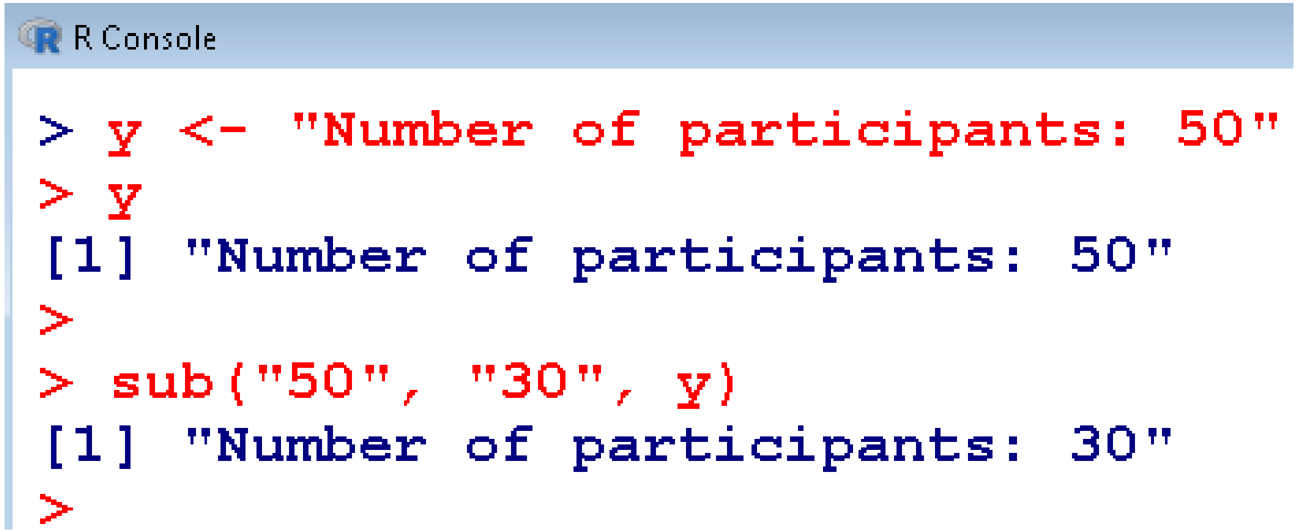
```
gsub(old, new, string)
```

Operations with Strings

Examples:

```
> y <- "Number of participants: 50"

> sub("50", "30", y)
[1] "Number of participants: 30"
```



```
> y <- "Number of participants: 50"
> y
[1] "Number of participants: 50"
>
> sub("50", "30", y)
[1] "Number of participants: 30"
>
```

Operations with Strings

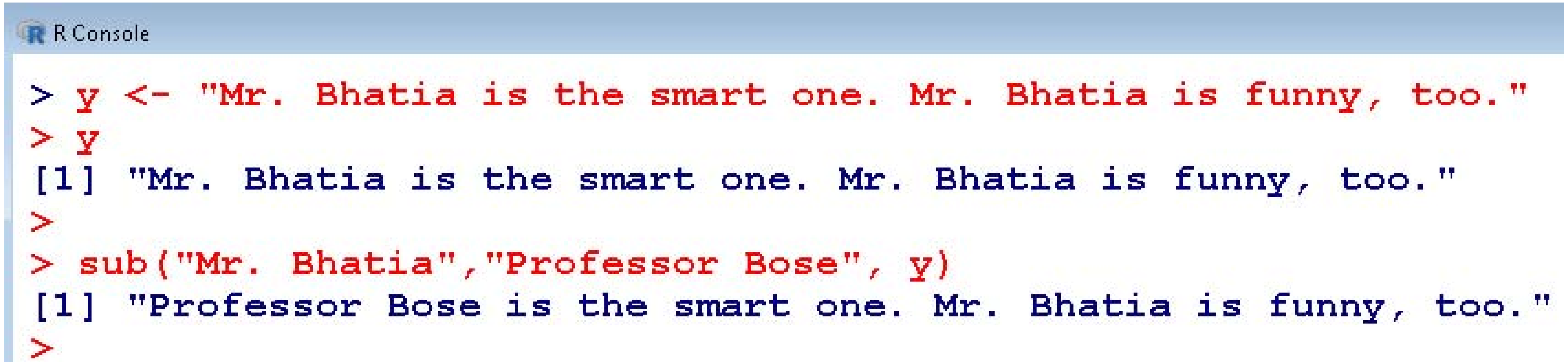
Examples:

```
>  y <- "Mr. Bhatia is the smart one. Mr.
Bhatia is funny, too."

> y
[1] "Mr. Bhatia is the smart one. Mr. Bhatia is
funny, too."

> sub("Mr. Bhatia","Professor Bose", y)
[1] "Professor Bose is the smart one. Mr.
Bhatia is funny, too."
```

Operations with Strings



```
> y <- "Mr. Bhatia is the smart one. Mr. Bhatia is funny, too."
> y
[1] "Mr. Bhatia is the smart one. Mr. Bhatia is funny, too."
>
> sub("Mr. Bhatia","Professor Bose", y)
[1] "Professor Bose is the smart one. Mr. Bhatia is funny, too."
>
```

Operations with Strings

Examples:

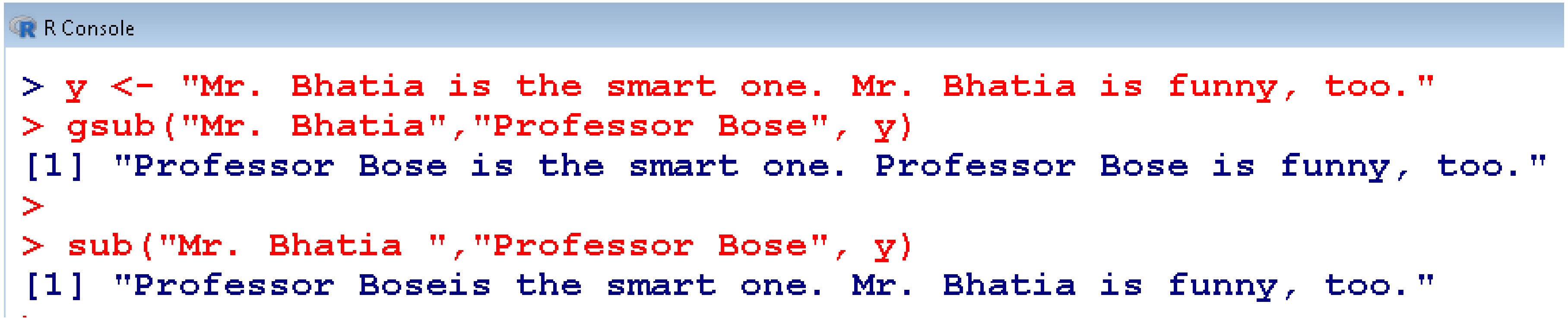
```
>  y <- "Mr. Bhatia is the smart one. Mr.
Bhatia is funny, too."

> gsub("Mr. Bhatia","Professor Bose", y)
[1] "Professor Bose is the smart one. Professor
Bose is funny, too."
```

Recall

```
> sub("Mr. Bhatia ","Professor Bose", y)
[1] "Professor Bose is the smart one. Mr.
Bhatia is funny, too."
```

Operations with Strings



```
> y <- "Mr. Bhatia is the smart one. Mr. Bhatia is funny, too."
> gsub("Mr. Bhatia","Professor Bose", y)
[1] "Professor Bose is the smart one. Professor Bose is funny, too."
>
> sub("Mr. Bhatia ","Professor Bose", y)
[1] "Professor Boseis the smart one. Mr. Bhatia is funny, too."
```

Operations with Strings

`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

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

Examples:

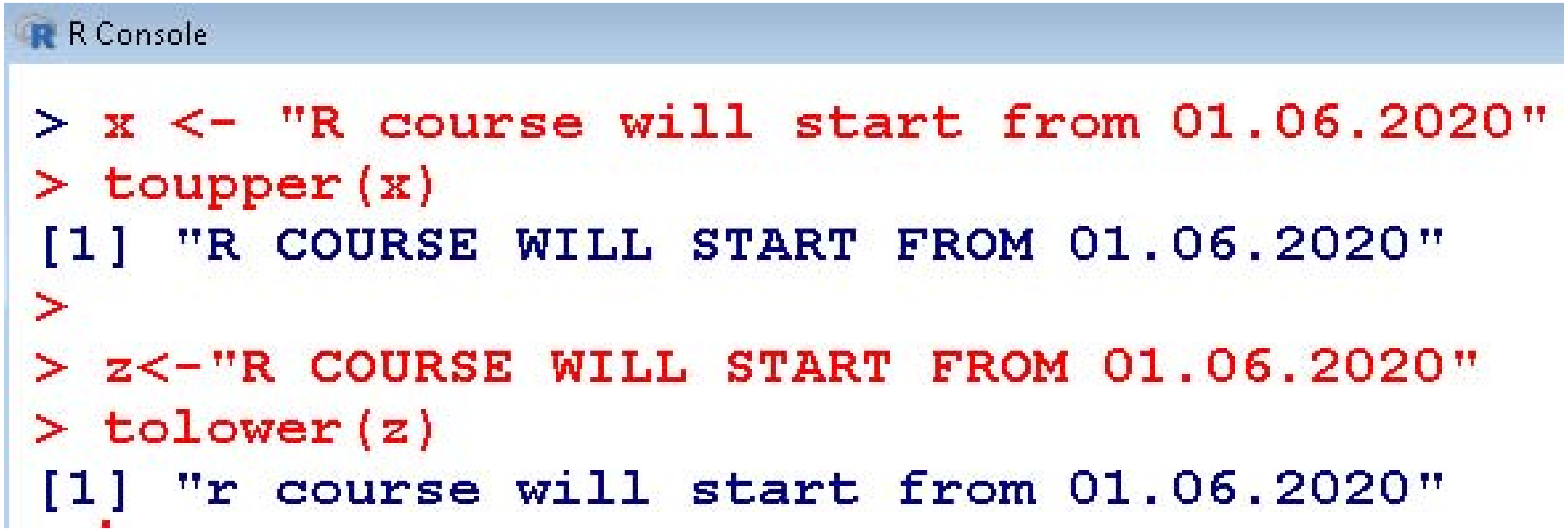
```
> x <- "R course will start from 01.06.2020"

> toupper(x)
[1] "R COURSE WILL START FROM 01.06.2020"

> z<-"R COURSE WILL START FROM 01.06.2020"

> tolower(z)
[1] "r course will start from 01.06.2020"
```

Operations with Strings



```
> x <- "R course will start from 01.06.2020"
> toupper(x)
[1] "R COURSE WILL START FROM 01.06.2020"
>
> z<-"R COURSE WILL START FROM 01.06.2020"
> tolower(z)
[1] "r course will start from 01.06.2020"
```