

Introduction to R Software

Strings – Display and Formatting

:::

Replacement and Evaluation of Strings

Shalabh

Department of Mathematics and Statistics

Indian Institute of Technology Kanpur

Operations with Strings

R has various functions for regular expression based match and replaces.

Some functions (e.g., `grep`, `grepl`, etc.) are used for searching for matches and functions whereas `sub` and `gsub` are used for performing replacement.

Operations with Strings

grep function:

The **grep** function is used for searching the matches.

(**sub** and **gsub** are used for performing replacement.)

grep : Globally search regular expression and print it

grep(pattern, x) search for matches to argument

pattern within each element of a character vector **x**.

It returns an integer vector of the indices of the elements of **x** that yielded a match

Operations with Strings

`grep(pattern, x, value = TRUE)` returns a character vector containing the selected elements of `x`.

```
> str <- c("R Course", "exercises", "include  
examples of R language")
```

```
> grep("ex", str, value=T)
```

```
[1] "exercises" "include examples of R language"
```

Operations with Strings

`grep(pattern, x, value = FALSE)` returns an integer vector of the indices of the elements of `x` that yielded a match

`value = FALSE` is default.

```
> str <- c("R Course", "exercises", "include  
examples of R language")
```

```
> grep("ex", str, value=F)
```

```
[1] 2 3
```

Operations with Strings

```
R Console
> str <- c("R Course", "exercises", "include examples of R language")
>
> grep("ex",str,value=T)
[1] "exercises"                "include examples of R language"
>
> grep("ex",str,value=F)
[1] 2 3
```

Operations with Strings

Example:

```
> x <- "R course 24.07.2017"
```

```
> y <- "Number of participants: 25"
```

```
> c(x,y)  # Combine the two strings
```

```
[1] "R course 24.07.2017"  "Number of  
participants: 25"
```

```
> grep("our", c(x,y) )
```

```
[1] 1
```

"our" is in the 1st element (in the word "**course**"), therefore in x.
There is no "our" in y.

Operations with Strings

Example:

```
x <- "R course 24.07.2017"
```

```
y <- "Number of participants: 25"
```

```
> c(x,y)  # Combine the two strings
```

```
[1] "R course 24.07.2017"  "Number of  
participants: 25"
```

```
> grep("Num", c(x,y) )
```

```
[1] 2
```

“Num” is in the 2nd element (in the word “Number”), therefore in y.

There is no “Num” in x.

Operations with Strings

grep function:

```
R Console
> x <- "R course 24.07.2017"
> x
[1] "R course 24.07.2017"
> y <- "Number of participants: 25"
> y
[1] "Number of participants: 25"
> c(x,y)
[1] "R course 24.07.2017"
[2] "Number of participants: 25"
> grep("our", c(x,y) )
[1] 1
```

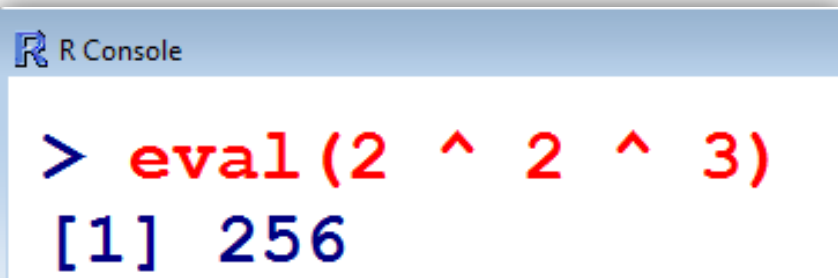
Operations with Strings

`eval` function:

`eval` function evaluates an (Unevaluated) R expression in a specified environment.

Example:

```
> eval(2 ^ 2 ^ 3)
[1] 256
```

A screenshot of an R Console window. The title bar is light blue with the R logo and the text "R Console". The console area is white and shows the command "> eval(2 ^ 2 ^ 3)" in red text, followed by the output "[1] 256" in blue text.

```
> eval(2 ^ 2 ^ 3)
[1] 256
```

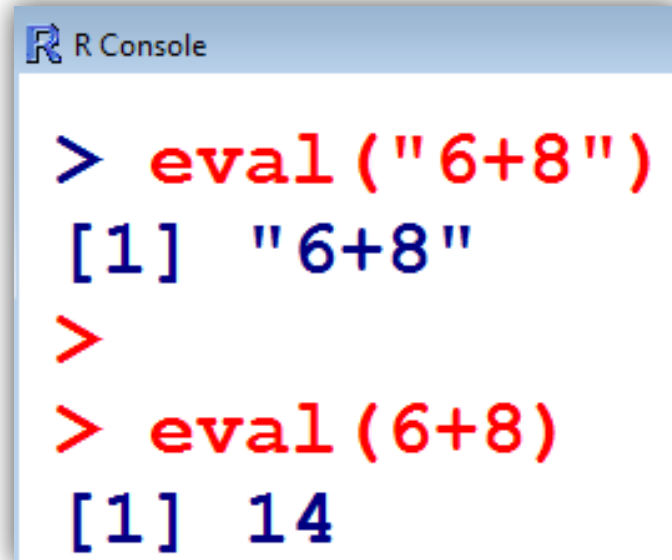
Operations with Strings

`eval` function:

Example:

```
> eval("6+8")  
[1] "6+8"
```

```
> eval(6+8)  
[1] 14
```

A screenshot of an R Console window. The title bar says 'R Console'. The console shows two commands and their outputs. The first command is '> eval("6+8")' in red, followed by the output '[1] "6+8"' in blue. The second command is '> eval(6+8)' in red, followed by the output '[1] 14' in blue.

```
> eval("6+8")  
[1] "6+8"  
>  
> eval(6+8)  
[1] 14
```

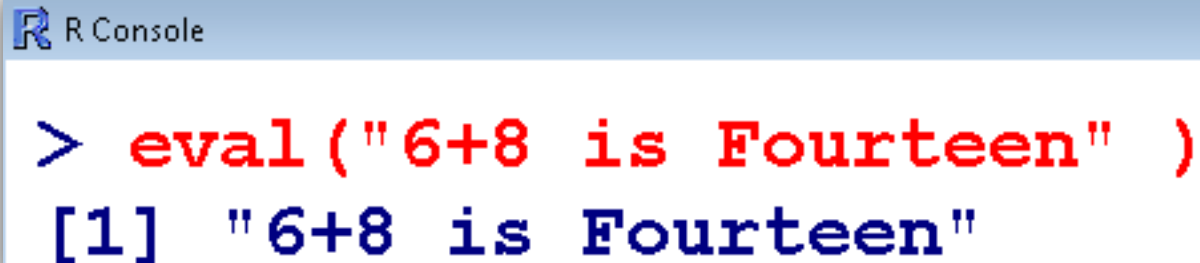
The `eval()` function evaluates an expression, but `"6+8"` is a string, not an expression whereas `6+8` is not an expression.

Operations with Strings

`eval` function:

Example:

```
> eval("6+8 is Fourteen" )  
[1] "6+8 is Fourteen"
```

A screenshot of an R console window. The title bar is light blue and contains the R logo and the text "R Console". The console area has a white background. It shows the same R code and output as the previous block: a prompt followed by the command `eval("6+8 is Fourteen")` in red text, and the output `[1] "6+8 is Fourteen"` in dark blue text. A small red cursor is visible at the end of the output line.

```
> eval("6+8 is Fourteen" )  
[1] "6+8 is Fourteen"
```

Operations with Strings

`parse` function:

`parse()` with `text=string` is used to change the string into an expression.

Example:

```
> eval("6+8")
```

```
[1] "6+8"
```

```
> class("6+8")
```

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

```
> eval(parse(text="6+8"))
```

```
[1] 14
```

```
> class(parse(text="6+8"))
```

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

Operations with Strings

parse function:

```
R Console  
  
> eval("6+8")  
[1] "6+8"  
>  
> class("6+8")  
[1] "character"  
>  
> eval(parse(text="6+8"))  
[1] 14  
>  
> class(parse(text="6+8"))  
[1] "expression"
```