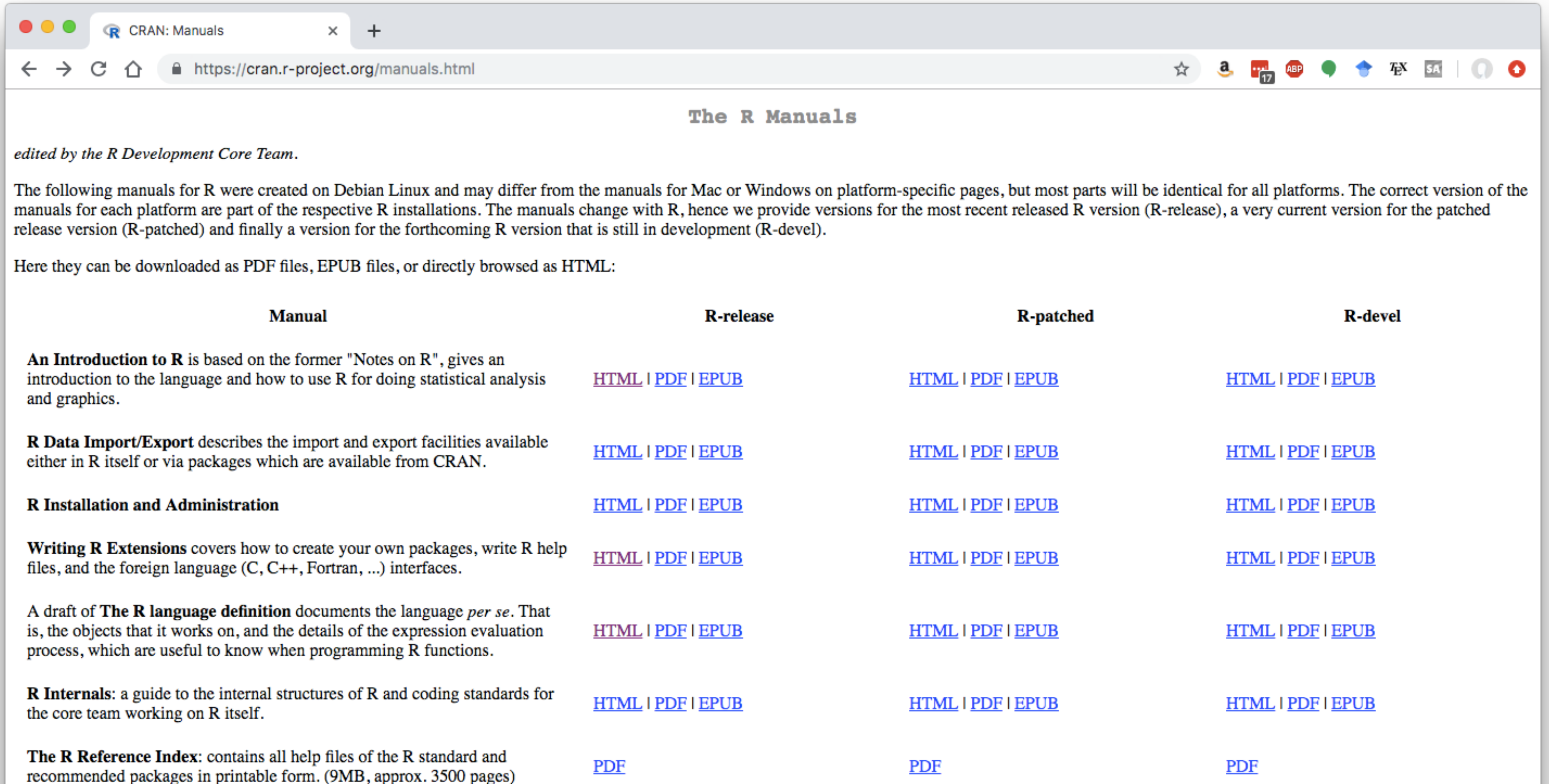


Non-standard evaluation (NSE)

An aside— how we used to learn R

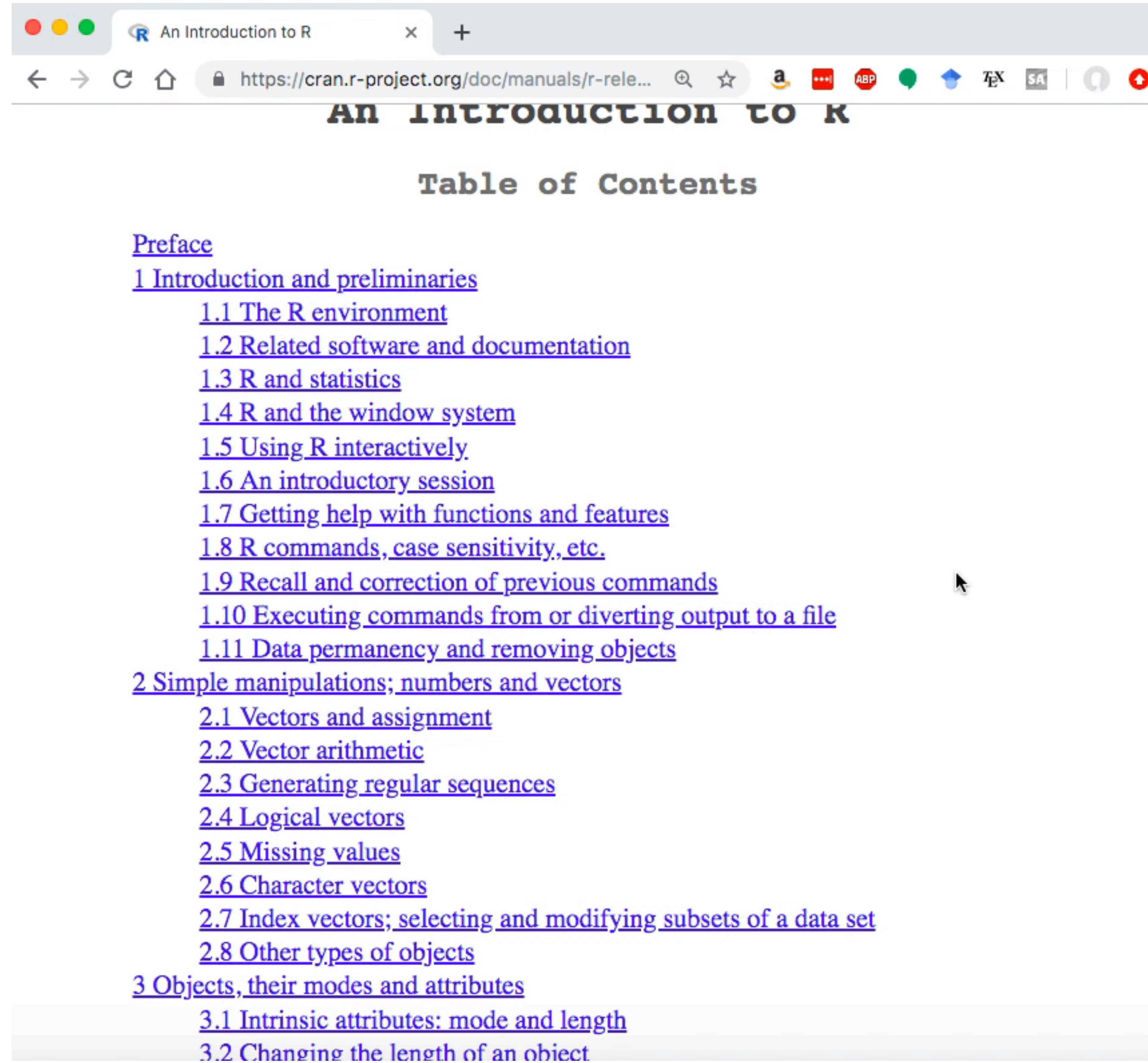


The screenshot shows a web browser window with the address bar displaying `https://cran.r-project.org/manuals.html`. The page title is "The R Manuals" and it is edited by the R Development Core Team. The page contains a table of manuals with columns for the manual name, R-release, R-patched, and R-devel versions. Each manual entry includes a brief description and links to download the manual as HTML, PDF, or EPUB files.

Manual	R-release	R-patched	R-devel
An Introduction to R is based on the former "Notes on R", gives an introduction to the language and how to use R for doing statistical analysis and graphics.	HTML PDF EPUB	HTML PDF EPUB	HTML PDF EPUB
R Data Import/Export describes the import and export facilities available either in R itself or via packages which are available from CRAN.	HTML PDF EPUB	HTML PDF EPUB	HTML PDF EPUB
R Installation and Administration	HTML PDF EPUB	HTML PDF EPUB	HTML PDF EPUB
Writing R Extensions covers how to create your own packages, write R help files, and the foreign language (C, C++, Fortran, ...) interfaces.	HTML PDF EPUB	HTML PDF EPUB	HTML PDF EPUB
A draft of The R language definition documents the language <i>per se</i> . That is, the objects that it works on, and the details of the expression evaluation process, which are useful to know when programming R functions.	HTML PDF EPUB	HTML PDF EPUB	HTML PDF EPUB
R Internals : a guide to the internal structures of R and coding standards for the core team working on R itself.	HTML PDF EPUB	HTML PDF EPUB	HTML PDF EPUB
The R Reference Index : contains all help files of the R standard and recommended packages in printable form. (9MB, approx. 3500 pages)	PDF	PDF	PDF

Translations of manuals into other languages than English are available from the [contributed documentation](#) section (only a few translations are available).

An aside— how we used to learn R



The image shows a screenshot of a web browser window. The address bar displays the URL <https://cran.r-project.org/doc/manuals/r-rele...>. The page title is "AN INTRODUCTION TO R" and the main heading is "Table of Contents". The content is a list of links to various sections of the manual, including a Preface, an Introduction and preliminaries section with 11 sub-sections, a Simple manipulations; numbers and vectors section with 8 sub-sections, and an Objects, their modes and attributes section with 2 sub-sections.

Preface
1 Introduction and preliminaries
1.1 The R environment
1.2 Related software and documentation
1.3 R and statistics
1.4 R and the window system
1.5 Using R interactively
1.6 An introductory session
1.7 Getting help with functions and features
1.8 R commands, case sensitivity, etc.
1.9 Recall and correction of previous commands
1.10 Executing commands from or diverting output to a file
1.11 Data permanency and removing objects
2 Simple manipulations; numbers and vectors
2.1 Vectors and assignment
2.2 Vector arithmetic
2.3 Generating regular sequences
2.4 Logical vectors
2.5 Missing values
2.6 Character vectors
2.7 Index vectors; selecting and modifying subsets of a data set
2.8 Other types of objects
3 Objects, their modes and attributes
3.1 Intrinsic attributes: mode and length
3.2 Changing the length of an object

"There are three kinds of language objects that are available for modification, calls, expressions, and functions."

Some useful functions for computing on the language using base R:

- `quote()`
- `enquote()`
- `substitute()`
- `deparse()`
- `eval()`

Call objects

"sometimes referred to as "unevaluated expressions", although this terminology is somewhat confusing" (thanks, R!)

We can get a call object using the function `quote()` (not to be confused with a quoted string)

```
> quote(2+2)
```

```
2 + 2
```

```
> "2+2" # just a character string
```

```
[1] "2+2"
```

Call objects

If you wanted to then evaluate a `quote()`ed call object, you could use `eval()`

```
> eval(quote(2+2))
```

```
[1] 4
```

```
> eval("2+2") # there's nothing to evaluate here
```

```
[1] "2+2"
```

Remember, R is lazy

Sometimes, `quote()` doesn't give you exactly what you were expecting, because R is lazy.

```
> a <- 1
```

```
> b <- 2
```

```
> quote(a + b)
```

```
a + b
```

Remember, R is lazy

This is where `substitute()` comes in. `substitute()` will substitute in the values it knows about in a particular environment.

```
> substitute(a + b, env = .GlobalEnv)
```

```
a + b
```

```
> ?substitute
```

"If it is an ordinary variable, its value is substituted, unless env is `.GlobalEnv` in which case the symbol is left unchanged."

Remember, R is lazy

Okay... but environments are just lists! So we can make our own.

```
> substitute(a + b, list(a = 1, b = 2))
```

```
1 + 2
```

Of course, everything needs to be defined in that environment

```
> substitute(a + b, list(a = 1, b = x))
```

```
Error: object 'x' not found
```

```
> substitute(a + b, list(a = 1, b = quote(x)))
```

```
1 + x
```

Tidyverse NSE

`quo()` is like `quote()`

`enquo()` is like `substitute()`

`!!` is like `eval()` ?

Where we're going...

I want you to create a new version of your bootstrap function, which works in the tidyverse. In other words, instead of calling

```
bootstrap(mtcars$mpg, samples = 500)
```

I want to be able to call

```
mtcars %>% bootstrap(mpg, samples = 500)
```