1-1 introduction

Levente Orban

2018-07-26

Introduction

After starting R, the console is ready for input. You can enter values next to the prompt (>)

```
1 + 2
```

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

To active a line of code, press << Enter >>.

To square numbers, use the hat (^) sign:

```
3^2
```

```
## [1] 9
```

Use slash (/) to divide numbers and star (*) to multiply them:

```
4/4
```

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

```
7*7
```

```
## [1] 49
```

You can even execute a variety of math computations easily:

```
sin(0.5)
```

```
## [1] 0.4794255
```

```
atan(pi)
```

```
## [1] 1.262627
```

```
abs(-5)
```

```
## [1] 5
```

```
sqrt(16)
```

```
## [1] 4
```

```
factorial(20)
```

```
## [1] 2.432902e+18
```

This is all great so far, but if I want to refer back to an earlier calculation? You can assign values to variables using the operator "=". Just typing the variable by itself at the prompt will print out the value. Another form of the operator is "<-" is also good.

```
x = 1
x
```

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

This is fine, but in R, there is another way:

```
x <- 1
```

This is called an assignment operator. For now you can use either, but the differences will become clearer later on when we discuss functions in depth.

Variable names can follow a lot of different naming schemes. They can be virtually any word: mix of upper and lower case letters, numbers, periods or underscores. Variables names however, cannot contain spaces. For example: myVariable, my.Variable, my_variable or even myvariable

Take a look at Google's R style guide: https://google.github.io/styleguide/Rguide.xml It's a bit like APA syle guide for R. There are other style guides out there and it probably doesn't matter which one you use, even if you create your own, as long as you use one.

An R function is invoked by its name, then followed by the parenthesis, and zero or more arguments. The following applies the function c to combine these three numeric values into a vector.

```
y = c(1,2,3)
```

Vectors can contain string values too

```
colours = c("red","yellow","blue")
```

Data Types

Numerics

```
k = 1
k
```

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

```
class(k)    # prints the class name of k, "numeric"
```

```
## [1] "numeric"
```

```
is.integer(k)  # is k an integer?
```

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

```
y = as.integer(1)
y           # prints the value of y
```

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

```
class(y)    # prints the class name of y, "integer"
```

```
## [1] "integer"
```

```
as.integer("3.45")   # will convert it to an integer, 3
```

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

```
as.integer("red")   # will produce an error
```

```
## Warning: NAs introduced by coercion
```

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

```
as.numeric() ## will convert to a float value
```

```
## numeric(0)
```

Text after the hashtag on a line with computations is considered a comment and not part of the calculation:

```r
1 + 1    # this is a comment
```

```
## [1] 2
```

Basic vector operations

```r
employmentStatus = c(TRUE,FALSE,FALSE,TRUE,TRUE,TRUE,FALSE)
employmentStatus
```

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

Tells you the number of elements in the list

```r
length(employmentStatus)
```

```
## [1] 7
```

Adds up all the values in the list

```r
sum(employmentStatus)
```

```
## [1] 4
```

You can do arithmetics with vector operations:

```r
sum(employmentStatus)/length(employmentStatus)
```

```
## [1] 0.5714286
```

Pick out the third element from the list

```r
employmentStatus[3]
```

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

Load some data

```r
siblings = c(1,3,3,1,2,2,1,1,1,1,0,3,1,2,5,2,0,0,2,2,1,1,2,1,6,2,2,2,1,1,2)
```

How do we compute the sample size? Count the number values in your data set

```r
length(siblings)
```

```
## [1] 31
```

Let's load another data set

```r
age=c(17,27,19,23,25,20,54,21,22,22,20,19,19,28,20,19,20,19,21,59,19,19,20,21,29,21,20,35)
```

A partial calculation, needed for the mean is to sum all the scores. We save the result in a variable.

```r
summed_age = sum(age)
```

Another partial calculation is to find the sample size (n). We save this in a variable too.

```r
sample_size = length(age)
```

Calculate the mean by dividing the sum of scores with the sample size

```r
mymean = summed_age/sample_size
mymean
```

```
## [1] 24.21429
```

You can verify your calculations by using the built-in R function to compute the mean:

```r
mean(age)
```

```
## [1] 24.21429
```

Find the median of the data set

```r
median(age)
```

```
## [1] 20.5
```