

3.3: Doing Simple Calculations with R

Okay, now that we've discussed some of the tedious details associated with typing R commands, let's get back to learning how to use the most powerful piece of statistical software in the world as a \$2 calculator. So far, all we know how to do is addition. Clearly, a calculator that only did addition would be a bit stupid, so I should tell you about how to perform other simple calculations using R. But first, some more terminology. Addition is an example of an "operation" that you can perform (specifically, an arithmetic operation), and the **operator** that performs it is `+`. To people with a programming or mathematics background, this terminology probably feels pretty natural, but to other people it might feel like I'm trying to make something very simple (addition) sound more complicated than it is (by calling it an arithmetic operation). To some extent, that's true: if addition was the only operation that we were interested in, it'd be a bit silly to introduce all this extra terminology. However, as we go along, we'll start using more and more different kinds of operations, so it's probably a good idea to get the language straight now, while we're still talking about very familiar concepts like addition!

3.3.1 Adding, subtracting, multiplying and dividing

So, now that we have the terminology, let's learn how to perform some arithmetic operations in R. To that end, Table 3.1 lists the operators that correspond to the basic arithmetic we learned in primary school: addition, subtraction, multiplication and division.

Table 3.1: Basic arithmetic operations in R. These five operators are used very frequently throughout the text, so it's important to be familiar with them at the outset.

operation	operator	example input	example output
addition	<code>+</code>	<code>10 + 2</code>	12
subtraction	<code>-</code>	<code>9 - 3</code>	6
multiplication	<code>*</code>	<code>5 * 5</code>	25
division	<code>/</code>	<code>10 / 3</code>	3
power	<code>^</code>	<code>5 ^ 2</code>	25

As you can see, R uses fairly standard symbols to denote each of the different operations you might want to perform: addition is done using the `+` operator, subtraction is performed by the `-` operator, and so on. So if I wanted to find out what 57 times 61 is (and who wouldn't?), I can use R instead of a calculator, like so:

```
57 * 61
```

```
## [1] 3477
```

So that's handy.

3.3.2 Taking powers

The first four operations listed in Table 3.1 are things we all learned in primary school, but they aren't the only arithmetic operations built into R. There are three other arithmetic operations that I should probably mention: taking powers, doing integer division, and calculating a modulus. Of the three, the only one that is of any real importance for the purposes of this book is taking powers, so I'll discuss that one here; the other two are discussed in Chapter 7.

For those of you who can still remember your high school maths, this should be familiar. But for some people high school maths was a long time ago, and others of us didn't listen very hard in high school. It's not complicated. As I'm sure everyone will probably remember the moment they read this, the act of multiplying a number x by itself n times is called "raising x to the n -th power". Mathematically, this is written as x^n . Some values of n have special names: in particular x^2 is called x -squared, and x^3 is called x -cubed. So, the 4th power of 5 is calculated like this:

$$5^4 = 5 \times 5 \times 5 \times 5$$

One way that we could calculate 5^4 in R would be to type in the complete multiplication as it is shown in the equation above. That is, we could do this

```
5 * 5 * 5 * 5
```

```
## [1] 625
```

but it does seem a bit tedious. It would be very annoying indeed if you wanted to calculate 5^{15} , since the command would end up being quite long. Therefore, to make our lives easier, we use the power operator instead. When we do that, our command to calculate 5^4 goes like this:

```
5 ^ 4
```

```
## [1] 625
```

Much easier.

3.3.3 Doing calculations in the right order

Okay. At this point, you know how to take one of the most powerful pieces of statistical software in the world, and use it as a \$2 calculator. And as a bonus, you've learned a few very basic programming concepts. That's not nothing (you could argue that you've just saved yourself \$2) but on the other hand, it's not very much either. In order to use R more effectively, we need to introduce more programming concepts.

In most situations where you would want to use a calculator, you might want to do multiple calculations. R lets you do this, just by typing in longer commands.²¹ In fact, we've already seen an example of this earlier, when I typed in `5 * 5 * 5 * 5`. However, let's try a slightly different example:

```
1 + 2 * 4
```

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

Clearly, this isn't a problem for R either. However, it's worth stopping for a second, and thinking about what R just did. Clearly, since it gave us an answer of `9` it must have multiplied `2 * 4` (to get an interim answer of 8) and then added 1 to that. But, suppose it had decided to just go from left to right: if R had decided instead to add `1+2` (to get an interim answer of 3) and then multiplied by 4, it would have come up with an answer of `12`.

To answer this, you need to know the **order of operations** that R uses. If you remember back to your high school maths classes, it's actually the same order that you got taught when you were at school: the "**BEDMAS**" order.²² That is, first calculate things inside **B**rackets `()`, then calculate **E**xponents `^`, then **D**ivision `/` and **M**ultiplication `*`, then **A**ddition `+` and **S**ubtraction `-`. So, to continue the example above, if we want to force R to calculate the `1+2` part before the multiplication, all we would have to do is enclose it in brackets:

```
(1 + 2) * 4
```

```
## [1] 12
```

This is a fairly useful thing to be able to do. The only other thing I should point out about order of operations is what to expect when you have two operations that have the same priority: that is, how does R resolve ties? For instance, multiplication and division are actually the same priority, but what should we expect when we give R a problem like `4 / 2 * 3` to solve? If it evaluates the multiplication first and then the division, it would calculate a value of two-thirds. But if it evaluates the division first

it calculates a value of 6. The answer, in this case, is that R goes from *left to right*, so in this case the division step would come first:

```
4 / 2 * 3
```

```
## [1] 6
```

All of the above being said, it's helpful to remember that *brackets always come first*. So, if you're ever unsure about what order R will do things in, an easy solution is to enclose the thing *you* want it to do first in brackets. There's nothing stopping you from typing `(4 / 2) * 3`. By enclosing the division in brackets we make it clear which thing is supposed to happen first. In this instance you wouldn't have needed to, since R would have done the division first anyway, but when you're first starting out it's better to make sure R does what you want!

This page titled [3.3: Doing Simple Calculations with R](#) is shared under a [CC BY-SA 4.0](#) license and was authored, remixed, and/or curated by [Danielle Navarro](#) via [source content](#) that was edited to the style and standards of the LibreTexts platform.