StartR Creating Mathematical Functions

A function is an important concept in mathematics, used throughout science and technology and the basis for organizing computer programs. As you know, a function is a transformation from inputs to an output. Mathematical functions have a narrower definition, for example that the same inputs will always produce the same output.

In creating a function, you need to specify two things:

* What are the inputs. Another word used for “inputs” is arguments.
* What is the rule that maps the input to an output.

Since functions can have more than one input, it’s very helpful to be able to name the individual inputs so that you can refer to them without any ambiguity.

R provides a standard syntax for creating a function, and for giving the function a name so that you can refer to it later. Here’s an example, creating a function called h:

  > h = function(x){ 3\*x + 5 }

As you can see, part of this statement is assignment: storing the function under the name h. The function itself is created with the keyword function. Following the keyword is a pair of parentheses, within which is the name of the argument. (You’ll create functions with multiple arguments in a little bit.) Finally, there is a matched pair of curly braces — the { } — that contains the rule for transforming the inputs to the output. In this example, the rule is a very simple mathematical statement: multiply the input by 3, then add 5.

To use a function, you give the name, followed by parentheses containing the value of the input. For instance:

  > h(2)

  [1] 11

When there is more than one input, separate the names of the different inputs by commas. For instance:

  > g = function(x, m, b) {m\*x + b}

To use such a function, you give all three inputs. Keeping track of which input is which requires some attention. In the function g, the first input will become x, the second m, and the third b, like this:

  > g(2, 3, 5)

  [1] 11

If you neglect to give one of the required inputs, you will get an error message.

  > g(2, 3)

  Error in m \* x + b : 'b' is missing

To remind you which argument is which, it’s helpful to show the function itself. You do this by giving the name of the function as a command, without the parentheses and arguments.

  > h

  function(x){ 3\*x + 5 }

  > g

  function(x, m, b) {m\*x + b}

It is, of course, easy to make mistakes with the order of the arguments. To help you out, you can refer to the arguments by name, rather than by position. For instance:

  > g(x=2, b=5, m=3)

  [1] 11

  > g(m=3, x=2, b=5)

  [1] 11

Sometimes, you will want to define values for parameters and then pass them into functions. You can do this easily:

  > intercept = 5  
  > slope = 3  
  > g(x=2, m=slope, b=intercept)

  [1] 11

Exercise 1 Create a function named hypotenuse that takes as arguments the lengths of the sides of a right triangle and produces as an output the length of the hypotenuse. (Hint: Remember the Pythagorean Theorem: a2 + b2 = c2.)

Use your function to calculate the hypotenuse of a right triangle with edge lengths 7 and 19.

 20.248  20.348  20.448  20.538

Exercise 2 Here’s the formula for the gravitational force between two masses, m1 and m2 separated by a distance r. the universal constant of gravity G = 6.67384 × 10-11 meters3 kg-1 s-2

 = Gm1m2--.
       r2  

G is the universal constant of gravity G = 6.67384 × 10-11 meters3 kg-1 s-2. (When m1 and m2 are given in kilograms, and r in meters, the formula gives a force with units of Newtons.)

Write a function, to be named grav.force that takes m1, m2, and r as inputs. (In the computer equivalent of scientific notation, the value of G is 6.67384e-11.)

Use your function to calculate the force between two 100 kg objects separated by 1 meter.

 6.6738e-9  6.6738e-8  6.6738e-7  6.673e-6

Exercise 3: Here’s a function that generates random numbers.

  > runif(1)

  [1] 0.76243

Is this a single-valued function?

The function **date()** doesn’t take any arguments. What does it return?