Introduction to Writing functions

Revolution Analytics

Outline

1 Overview

2 Custom functions

3 Understanding scope

Overview

In this session you write your own functions and learn about scope

- Write your own functions
- Get familiar with scope

Outline

1 Overview

2 Custom functions

3 Understanding scope

Control Flow

- Repeating a command multiple times
- Conditional command execution
- "Distant" command execution

Distant command execution: `function` execution.

We can think of this as a subroutine.

Writing functions

A function is an `object` in the same way a vector, matrix or dataframe is.

We can name functions we write as objects in the R workspace, e.g.:

```
test <- function(a, b) return(a + b)
test
```

```
## function(a, b) return(a + b)
## <environment: 0x2f2b600>
```

```
test(1, 2)
```

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

Writing functions (continued)

Short functions can be contained on a single line, but typically the function and it's environment will be defined as:

```
function.name <- function(arguments) {
  # everything the function does in here
}
```

A Few Notes

- `arguments` denotes a comma-separated list of "inputs" or "arguments."
- These values are passed to the internal function environment during program execution.

See `help('function')` for details.

Writing functions with default values

- Often the function you use in R will have default values for the arguments that you modify.
- These values are passed to the internal function environment during program execution.

```
test <- function(a = 1, b = 2) return(a + b)
test()
```

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

```
test(a = 3, b = 5)
```

```
## [1] 8
```

Exercise: Write a function

Your turn:

- Write a function to compute the maximum and minimum value of a vector
- Write another function to compute the 25th and 75th percentiles of a vector.

Hint: `look up quantile()`

Remember the quadratic formula?

$$ax^2 + bx + c = 0$$

Solving for x:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Exercise

- Write your own function to compute it.
- How do you handle imaginary numbers?

Questions about function definition?

Outline

1. Overview

2. Custom functions

3. Understanding scope

Scope

Multiple local environments can be nested within each other, e.g. in more complicated programs.

When we define a function, we create "local environment" separate from the global environment/out environment and only stored for the duration of program execution.

Scope Example 1

For example, if you have a `my.var` in the global environment and you assign another number to `my.var` in your function, R makes a local variable `my.var` and the global variable `my.var` will be intact.

```
my.var <- 1
temp.func <- function() my.var <- 5
temp.func()
my.var
```

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

Global and local scope

But global variables are accessible inside the function:

```
my.var <- 1
temp.func2 <- function() print(5 + my.var)
temp.func2()
```

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

Scope (continued)

Objects from one local environment only enter an outer environment via `return()` and `<<-`

```
a <- 0
env.function <- function() {
  a <<- 1
}
env.function()
a
```

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

Notes

Generally, it is considered bad practice to use <<−.

You can perform the same kind of operation using the function `assign()`.

Coding conventions

There are many (unofficial) coding conventions out there.

Check out the Google style guide for further guidance:
http://google-styleguide.googlecode.com/svn/
trunk/google-r-style.html

Write a summary stats function

Use `mtcars`, and write a function that accepts:

1. cylinder size (`cyl`) and
2. the name of any other column

and then returns the `mean` of that column-cylinder size subset.

How does `cyl` relate to `hp`?

Module review questions

- How do you define a `function` in R?
- How do you specify default arguments?
- What is meant by a `global` versus `local` environment?
- How do you pass values from one local environment to a different, outer environment?

Questions?

Thank you

Revolution Analytics is the leading commercial provider of software and support for the popular open source R statistics language.

www.revolutionanalytics.com
1.855.GET.REVO
Twitter: @RevolutionR