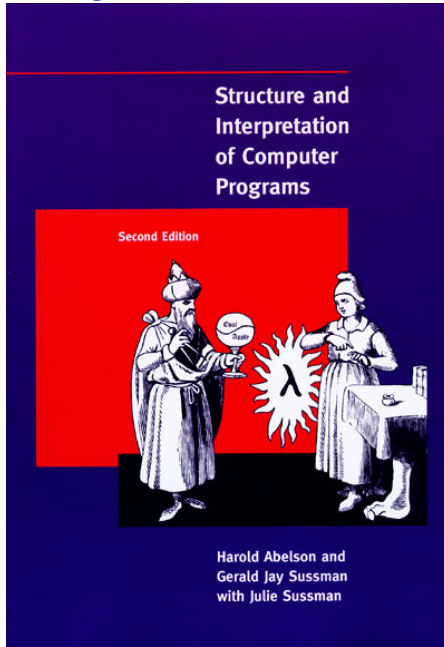


Introduction to Programming: the R perspective

Peter D Smits

April 26, 2013

What is programming?



What is programming?

Structure of Interpretation of Computer Programs by Abelson and Sussman 1996 page 1.

We are about to study the idea of a computational process. Computational processes are abstract beings that inhabit computers. As they evolve, processes manipulate other abstract things called data. The evolution of a process is directed by a pattern of rules called a program. People create programs to direct processes. In effect, we conjure the spirits of the computer with our spells.

Continued. . .

What is programming?

A computation process is indeed much like a sorcerer's idea of a spirit. It cannot be seen or touched. It is not composed of matter at all. However, it is very real. It can perform intellectual work. It can answer questions. It can affect the world by disbursing money at a bank or by controlling a robot arm in a factory. The programs we use to conjure processes are like a sorcerer's spells. They are carefully composed from symbolic expressions in arcane and esoteric programming languages that prescribe the tasks we want our process to perform.

R: a brief history

R is a programming language based on the S language.

Originally written by Ross Ihaka and Robert Gentleman, now by a team.



NyTimes

R: a brief history

Existed in a usable form since approximately 1997.

R is kind of a messy language and pretty slow. Lots of other problems/concerns.

Main advantages of R for statistical analysis

- ▶ designed with data in mind
- ▶ huge package universe
- ▶ lots of online resources
- ▶ graphics
- ▶ dynamic documents

Console and scripts

The console (or REPL) is the R command line. It is the blinking cursor you stare at when you first open R. It is useful for testing bits of code and hunting for bugs in your code.

Script files are text files with a series of commands/expressions in a row. These are then run to perform analysis. Scripts allow you to edit code and save it for later use. Write scripts, don't just use the console.

Using the console or REPL

Writing our first script

Flow control

Flow control is how you tell the computer what order to do things in.

Writing our first function

Using what we know now, let's write our own useful function.

Let's duplicate the sum function.

Writing our first function

First, let's find out what sum does.

```
x <- seq(5)
sum(x)
```

```
## [1] 15
```

```
y <- c(1, 1)
sum(y)
```

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

Writing our first function

```
sum.prime <- function(x) {  
  y <- 0 # storage  
  
  # add every value of x to y  
  for(i in seq(length(x))) {  
    # seq is for making sequences  
    # length determines how long a vector is  
  
    y <- y + x[i]  
  }  
  
  # return determines the output of a function  
  return(y)  
}
```