

Exercises I, IV and V

1-2 December 2005

1 Exercises I: Fundamental concepts of R**1.1 Simple number calculations with R**

Start R (after downloading and installing it, if necessary) and use the R Console to compute the value of the following expressions:

- 5^3
- $\log(10)$
- $\ln(10)$
- $10^{3\log(5)}$
- $\sqrt[3]{125}$

There is no cubic root function in R, but you can compute this as $125^{1/3}$.

1.2 Defining variables in R

Define variables x , y and z such that

- x is $\sqrt{2}$
- y is 2^5
- z is x^y

1.3 Vector calculations with R

Write R expressions that generate the following vector results:

- 15 16 17 18 19
- 19 18 17 16 15
- -15 -16 -17 -18 -19
- 10 20 30 40 50
- 0.5 1.0 1.5 ... 9.5 10.0
- $\sqrt{1}$ $\sqrt{2}$ $\sqrt{3}$ $\sqrt{4}$ $\sqrt{5}$

Use R vector expressions to compute the following sums:

- $1 + 2 + \dots + 100$
- $\frac{1}{1} + \frac{1}{2} + \dots + \frac{1}{99} + \frac{1}{100}$
- $1 + 4 + 9 + \dots + 25$

1.4 Defining your own functions in R

Define these functions in R:

- Define a function to compute the cubic root of x :
 $\text{cbrrt}(x) = \sqrt[3]{x}$
- Define a function to evaluate the polynomial
 $p(x) = x^2 + x - 6$.
- Define a function to compute the volume of a cylindrical can with height h and diameter d :
 $\text{vol}(h, d) = h \cdot \pi \cdot \left(\frac{d}{2}\right)^2$

1.5 Creating and saving an R script as a file

- Change your working directory by choosing `FILE || CHANGE DIR`. Change it to something sensible (maybe the Desktop, your USB memory stick, or a new folder you create for your R exercises).
- Open the R Editor by choosing `File || New script`.
- Enter a function definition in the R Editor; now you have an R script.
- Save the script to a file — remember the `.R` file type.
- Execute the script in the R Console by copy-paste or by marking the function definition in the R Editor, then choose `EDIT || RUN LINE OR SELECTION`, or simply press `Ctrl+R`.
- If there's a problem in your definition, go back and fix it in the R Editor, then repeat the above step.

2 Exercises IV: Numerical mathematics in R

- Use R to compute the following integral:
 $\int_0^{10} x^2 dx$
- Use R to find both zeroes for the polynomial $p(x) = x^2 + x - 6$. It is useful to plot the polynomial first, for x in the interval $[-5, 5]$.
- Use R to find the minimal value of the polynomial $p(x) = x^2 + x - 6$.

3 Exercises V: Programming in R (REVISED)

Hint: It is most useful to solve the following problems by making definitions in the R Editor, so that you can fix bugs in them and improve them, without having to type too much.

3.1 Loops that build vectors

Recall that the `c` function can be used to combine vectors and numbers into a new vector. For instance, when `v` is `11 22`, then `c(v, 77)` is `11 22 77`. Also note that the expression `c()` constructs the empty vector, with no elements.

What is the value of the vector variable `v` after each of these loops? Try to find the answer without running the script fragments.

- ```
v <- c()
for (i in 1:10)
 v <- c(v, i^2)
```
- ```
v <- c()
for (i in 1:10)
  v <- c(2*i, v)
```
- ```
u <- c(13, 4, 13, 8, 16, 9, 4, 2)
v <- c()
for (i in 1:length(u))
 if (u[i] > 10)
 v <- c(v, u[i])
```
- ```
v <- c()
for (i in 1:4)
  v <- c(v, i, v)
```

For the three first loops above, find a vector expression that directly computes the vector built by the loop.

3.2 Gradually building a vector of results

Write `for` loops that executes 10 iterations and constructs the vector results shown below:

- `2 4 8 16 32 64 128 256 512 1024`
- `1 2 2 4 3 8 4 16 5 32 6 64 7 128 8 256 9 512 10 1024`

3.3 Creating a vector and computing its contents

This loop computes the so-called Fibonacci numbers `1 1 2 3 5 8 13 21 34 55`, in which every number, except the two first ones, is the sum of the preceding two numbers.

This sequence may be interpreted as the number of mature female rabbits in each season, provided each mature female gets one female offspring each season, a newborn female takes one season to mature, and all old females survive. The Fibonacci numbers are frequently found in nature (pineapple, sunflower).

```
v <- c(1,1,rep(0,8));
for (i in 3:10)
  v[i] <- v[i-2] + v[i-1]
```

Explain how the loop works, by showing the contents of vector `v` when `i` is 3, 4, 5 and 6.

Now, write a function `fib(n)` that creates and returns the `n` first Fibonacci numbers. Use a suitably generalized version of the initialization and the loop shown above.

Let v be the result of computing `fib(100)`. Compute the ratio (quotient) between every pair of neighbor elements of v ; the result should be a 99-element vector. Use vector indexing to do this without a loop: divide the last 99 elements of v by the first 99 elements of v . What do you observe about the resulting sequence of ratios?

The number $\frac{\sqrt{5}+1}{2} \equiv 1.618034$ is called the Golden Ratio and is frequently found in nature, art and architecture.

3.4 Simulating multiple die rolls

Continuing the lecture's die rolling example, define a function `dice`, such that `dice(n)` returns the sum obtained by rolling n dice, that is, the total number of eyes. Use a `for` loop and the lecture's `die` function.

3.5 Simple animation of a plot

The function `plotsin(z)` plots the sine function translated by z (for x in $[0, 20]$):

```
plotsin <- function(z) {  
  plot(function(x) (sin(x+z)), 0, 20);  
}
```

This means that `plotsin(0)` plots the usual sine function; and `plotsin(1)` plots the function $f(x) = \sin(x+1)$, that is, the usual sine function shifted 1 to the left; and so on. Try it.

By repeatedly calling `plotsin(i)` for increasing values of i , one can make a simple animation: it looks like the sine curve slowly moves from right to left. (What happens in reality is that one translated sine curve is plotted after another). Write a `for` loop that calls `plotsin(i)` for i from 1 to 100. In the `for` loop, after every call to `plotsin`, call R function `Sys.sleep(0.2)` to make a pause of length 0.2 seconds.