Scientific programming in R – pandemic edition

Dr. Henry Löffler-Wirth
Maria Schmidt

Interdisciplinary Center for Bioinformatics - IZBI, University Leipzig
Loops

Aim: Learn how to run code in iterations.

Fill a vector with first 100 square numbers:

```r
> s <- c( 1^2, 2^2, 3^2, 4^2 .................. # this is a bad idea: annoying and not flexible
```

Simple loop in R to fill an initially empty vector `s`:

```r
for( variable in vector ) command/codeblock
!! i has a different value in each iteration !!
```

{Code blocks} subsume multiple commands which belong together in a loop, condition, or function definition. All commands contained in a block after `for(...)` are executed iteratively.
Conditional execution and loop control

Aim: Learn how to use conditions and to control execution of loops.

```r
> i <- 5
> i < 50
[1] TRUE

> if( i < 50 ) print( "ok" )
[1] "ok"

> for( i in 1:100 )
  {
    if( i < 50 )
    {
      s[ i ] <- i^2
    }
    else
    {
      s[ i ] <- i
    }
  }

> s
  [1] 1  4  9 16 25 36 49 64 81 100 121 144 169 196 225 256 289 324 361
```

Use `<`, `>`, `==` and `!=` to compare values. Result is Boolean.

Simple condition in R:
```
if( condition ) command/codeblock
```

Two-way condition in R:
```
if( condition ) command/block else command/block
```

Code blocks can be defined within another code blocks.
Conditional execution and loop control

Aim: Learn how to use conditions and to control execution of loops.

> s <- c()
> for( i in 1:100 )
  { if( i < 40 ) next
    if( i == 50 ) break
    s[ i ] <- i^2
  }

> length( s )
[1] 49

> s
[1]  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
[21]  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
[41] 1681 1764 1849 1936 2025 2116 2209 2304 2401

*next*: proceed to next iteration

*break*: skip remaining iterations

gaps in s will be filled with NA
Conditional execution and loop control

Aim: Learn how to use conditions and to control execution of loops.

```r
> for( i in 1:100 )
+ {
+   if( i < 10 || i > 90 ) print( i )
+ }
[1] 1
[1] 2
[1] 3
[1] 4
[1] 5
[1] 6
[1] 7
[1] 8
[1] 9
[1] 91
[1] 92
[1] 93
[1] 94
[1] 95
[1] 96
[1] 97
[1] 98
[1] 99
[1] 100
```

&&: AND operator
||: OR operator
!: NOT operator
Programming tasks II

• Simulate a bakery: You have 20 cookies for sale.
  – Hour 1: You eat one cookie.
  – Hour 2: You eat two cookies.
  – Hour 3: You eat three cookies.
  – Hour 4: You eat four cookies.
  – Hour 5: You eat five cookies.

→ How many cookies are left after 5 hours?

• Generate a vector containing square numbers up to 200².
  1.801.800 is the sum of how many consecutive square numbers (starting from 1²)?

• Generate the Fibonacci sequence (first 20 numbers): 1, 1, 2, 3, 5, 8, 13, 21, ... \( \Rightarrow f_n = f_{n-1} + f_{n-2} \)