R_intro_4

Karel Fišer, 2018

comparison operators

==, !=, <, >, <=, >=

```
> 1 == 2-1  # returns logical
[1] TRUE

> x <- 3
> x == 3 # returns logical
[1] TRUE
```

comparison operators

==, !=, <, >, <=, >=

```
> x <- 1:5
> x < 2 # returns logical vector
[1]  TRUE FALSE FALSE FALSE FALSE

> x == 2
[1] FALSE  TRUE FALSE FALSE FALSE

> x != 2
[1]  TRUE FALSE  TRUE  TRUE  TRUE
```

logical operators

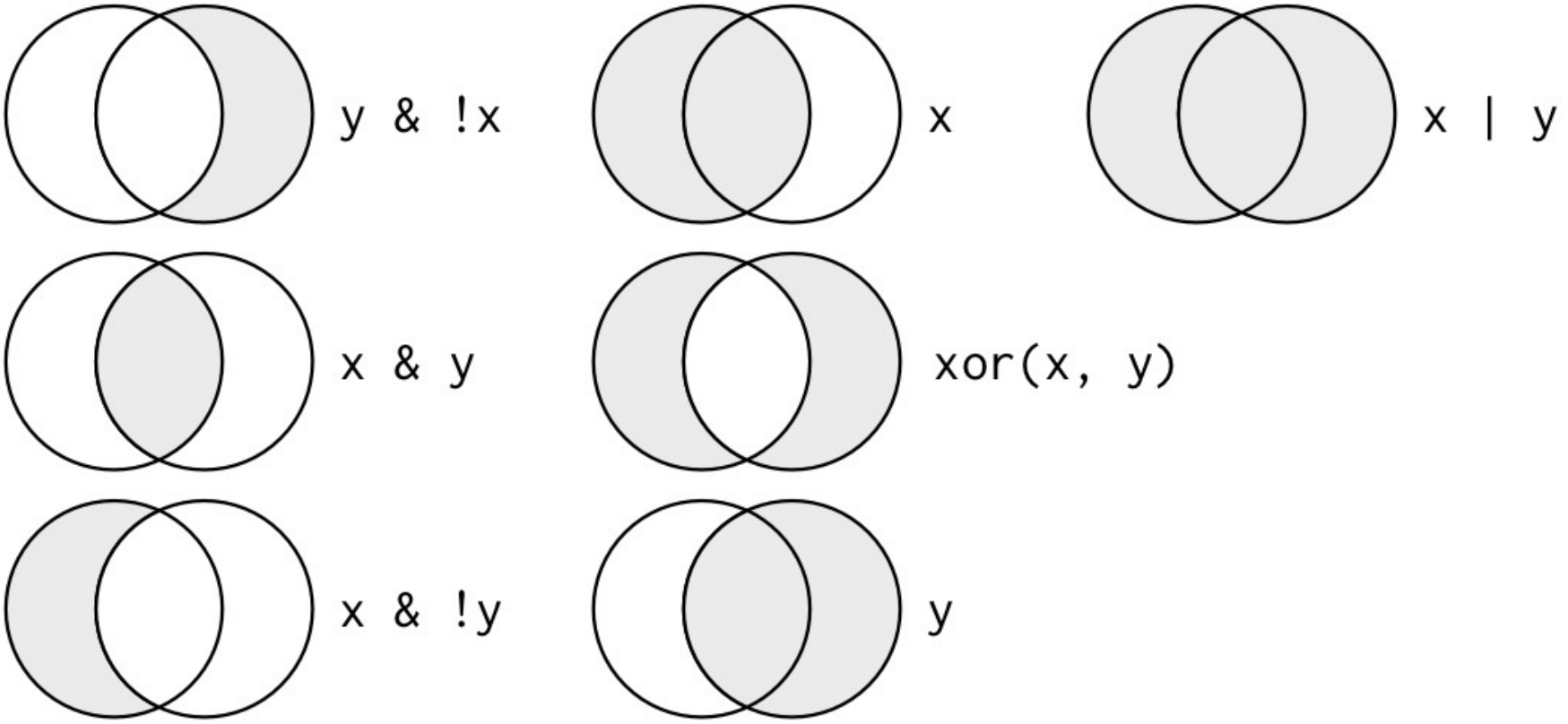
&, |, !

```
> x<2 | x>4 # returns logical vector
> x>2 & x<=4


> x>2 & x<=4
> x>2 && x<=4 # returns logical vector of length 1
  (compares only first elements of each)
> # a bit like x[1]>2 & x[1]<=4, but "Evaluation proceeds
  only until the result is determined"


> all(x>2 & x<=4) # returns logical vector of length 1
> any(x>2 & x<=4) # returns logical vector of length 1
```

logical operators



http://r4ds.had.co.nz/transform.html#comparisons

operators – returning values

```
> x[x < 2] # returns values
> # which(x < 2)
> x[x<2 | x>4]
> x[x>2 & x<=4]
```

white blood cell count

A data frame of 210 patients with Chronic Myeloid Leukemia from the Benelux CML study (Kluin-Nelemans et al. 1998).

```
> wbc[wbc < 4.5]
> wbc[wbc > 10]
> wbc[wbc >= 4.5 & wbc <= 10] # normal samples
> wbc[wbc >= 4.3 & wbc <= 10.8]
```

IF

if (condition is true) { command }

```
> x <- 0
> if (x==0) { print("zero") }
> 
> 
> 
> 
> 
> # Condition: a length-one logical vector.
```

IF

if (condition is true) { command }

```
> x <- 0
> if (x==0) { print("zero") }
```

if (condition is true) { command } else { command2 }

```
> if (x==0) { print("zero") }  else { print(x) }
```

IF

```
> if (x==0) {
>     x <-  "zero"
>     print(x)
> }

> if (x==0) {
>     print("zero")
> }  else {
>     print(x)
> }
```

IF

```
if (condition1 is true) {
    command1
} else if (condition2 is true) {
    command2
} else {
    command3
}
```

IF

```
> if (x==0) {
>     print("zero")
> } else if (x>0) {
>     print("positive")
> } else {
>     print("negative")
> }
```

Your turn

if

FOR

for (variable in sequence) {

statements

}

```
> for (ii in 1:5) {
>   print(ii)
> }
```

```
> for (ii in 1:5) {
>   2^ii
>   print(ii^ii)
>   x <- ii
> }
```

FOR – assignment

```
> y <- NULL; z <- NULL
> for (ii in 1:5) {
>   x <- ii^ii
>   y <- append(y, ii^ii)
>   z[ii] <- ii
> }
> x; y; z
```

FOR – sequence

```
> for (ii in 10:1) {
>   print(ii)
> }


> x <- c("cat", "snake", "spider")
> for (ii in 1:length(x)) { # ii in seq_along(x)
>   print(ii)
>   print(x[ii])
> }


> for (ii in x) {
>   print(ii)
>   print(x)
> }
```

Your turn

for

strings

```
> b2m <-
  scan(file=url("http://www.uniprot.org/uniprot/P
  61769.fasta"), what="character", sep="\t")


> b2m_seq <- paste(b2m[2:3], collapse="")
> substr(b2m_seq, 1, 12)
> strsplit(b2m_seq, "Y")
```

Retrieve sequence of 3 proteins and get Tyr surrounding aa

```
> proteins <- c("P00519", "P46109", "P61769") # ABL1, CRKL, B2M
> for (ii in proteins) {
>   prot_url <- paste("http://www.uniprot.org/uniprot/", ii, ".fasta", sep="")
>   protein_fasta <- scan(file=url(prot_url), what="character", sep="\t")
>   protein_seq <- paste(protein_fasta[2:length(protein_fasta)], collapse="")
>   # print(protein_seq)
>   for (ik in which(strsplit(protein_seq, '')[[1]]=='Y')) {
>     tyr_context <- substr(protein_seq, ik-1, ik+3)
>     print(tyr_context)
>     # abl1 specific sequence:  I/V/L-Y-X-X-P/F
>   }
> }
```

```
[1] "CYLEE"
[1] "LYDFV"
[1] "GYNHN"
[1] "NYITP"
...
```

TODO: find ABL1 Tyr kinase consensus sequences

Your turn

use both for and if

Print the Tyr context only if it matches ABL1 consensus sequence, which is I/V/L-Y-X-X-P/F

http://www.bioinformin.net