R

2020-09-17

Contents

Chapter 1

“ R”. - ,
!

ivanspozdniakov@gmail.com, VK Telegram:@pozdniakovivan.

Chapter 2

1. R

2.1

2.1.1 R Rstudio

R .

- R
  - Windows, **Download R ( ) for Windows.**
  - Mac, , 5 , *.pkg . , .
  - Linux, :

```
sudo apt-get install r-cran-base
```

R:

```r
sessionInfo()$R.version$version.string
```

```
## [1] "R version 4.0.2 (2020-06-22)"
```

R RStudio:

- RStudio

- ( ), , - :

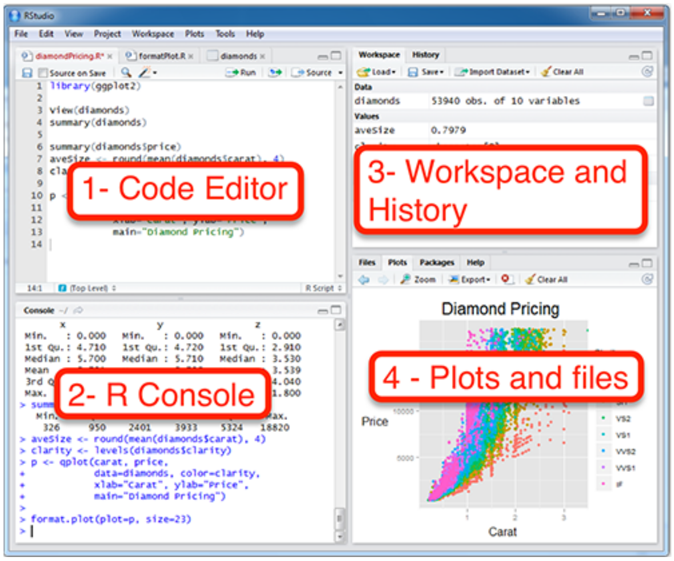
- RStudio cloud

: R - RStudio? , R — , RStudio — (IDE), .

RStudio — R, , , .
- .
Jupyter Notebook, R
RMarkdown — - , .
RMarkdown !

2.1.2 RStudio

, , :


RStudio
File Edit View Project Workspace Plots Tools Help
diamondPricing.R* formatPlot.R diamonds
Source on Save Run Source
14:1 (Top Level) R Script
Console ~/
Workspace History
Load Save Import Dataset Clear All
Files Plots Packages Help
Zoom Export Clear All
1- Code Editor
2- R Console
3- Workspace and History
4 - Plots and files
Diamond Pricing
Price
Carat
VS2
VS1
VVS2
VVS1
IF

```
library(ggplot2)

View(diamonds)
summary(diamonds)

summary(diamonds$price)
aveSize <- round(mean(diamonds$carat), 4)
cla
p <
       main="Diamond Pricing")
```

```
Data
diamonds      53940 obs. of 10 variables
Values
aveSize       0.7979
```

```
       x                   y                   z
Min.   : 0.000   Min.   : 0.000   Min.   : 0.000
1st Qu.: 4.710   1st Qu.: 4.720   1st Qu.: 2.910
Median : 5.700   Median : 5.710   Median : 3.530
Mean                                       : 3.539
3rd Q                                       4.040
Max.                                        1.800
> summ
  Min.                                        Max.
   326     950    2401    3933    5324    18820
> aveSize <- round(mean(diamonds$carat), 4)
> clarity <- levels(diamonds$clarity)
> p <- qplot(carat, price,
+            data=diamonds, color=clarity,
+            xlab="Carat", ylab="Price",
+            main="Diamond Pricing")
>
> format.plot(plot=p, size=23)
>
```

Figure 2.1:

: **1 - Code Editor** ( )[1] **2 - R Console** ( ). .
.

2 - R Console `Enter`.
- . , .
.

1 - Code Editor `Ctrl + Enter` (`Cmd + Enter` macOS). , .
. `Ctrl + A` Windows Linux, `Cmd + A` macOS[2]. (

[1] RStudio . , , `File - New File - R Script`.

[2] RStudio . , `Help - Keyboard Shortcuts Help`.

), **2 - R Console**, . - , - . . File - Save As.... R *.R*, , - “ ”.

3 - Workspace and History — . , . , .

4 - Plots and files. . - , , - , , . (Packages) Help . .

2.1.3 R

R — , . R . , R . , , .

R +, -, *, /, ^ ( ), () . .

:

```
40+2
```

```
## [1] 42
```

```
3-2
```

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

```
5*6
```

```
## [1] 30
```

```
99/9 #
```

```
## [1] 11
```

```
2^3 #
```

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

```
13 %/% 3 #
```

```
## [1] 4
```

```
13 %% 3 #
```

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

- .

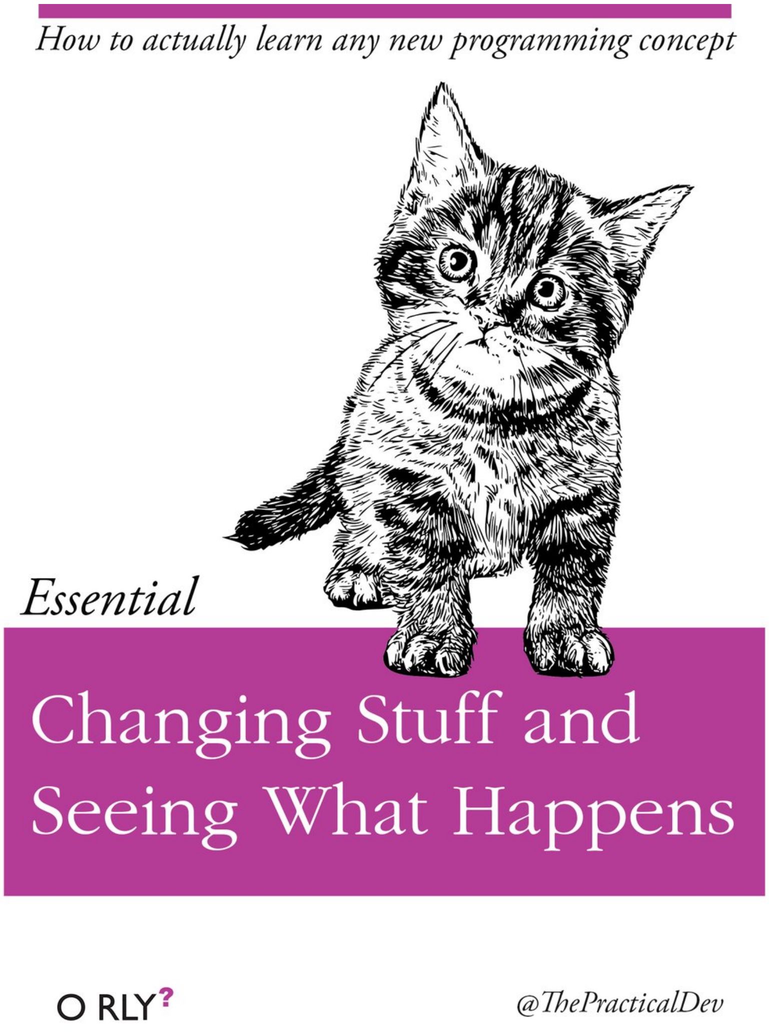

How to actually learn any new programming concept
Essential
Changing Stuff and Seeing What Happens
O RLY?
@ThePracticalDev

Figure 2.2:

, ? .

, (#). , #
R . ,
(-), [3].
, , - ,

[3] , ,
. !

: “ ” - # , . : Ctrl + Shift + C (Cmd + Shift + C macOS) — # .

, 14% :

```
2 + 2 * 2
```

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

, - , 6 (,) .

(. . , *operator precedence*) R , .

```
(2+2)*2
```

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

, , , , . ?Syntax.

2.1.4

- . , , , :

```
16^0.5
```

```
## [1] 4
```

, : - . ( ) - , - , - (, . .).

, , :

```
sqrt(16)
```

```
## [1] 4
```

R — case-sensitive , . . . SQRT(16) .

:

```
log(8)
```

```
## [1] 2.079442
```

, , … - , , - .

!

— , , ,
.

8 2 3:

$\log_2 8 = 3$

2 3 8:

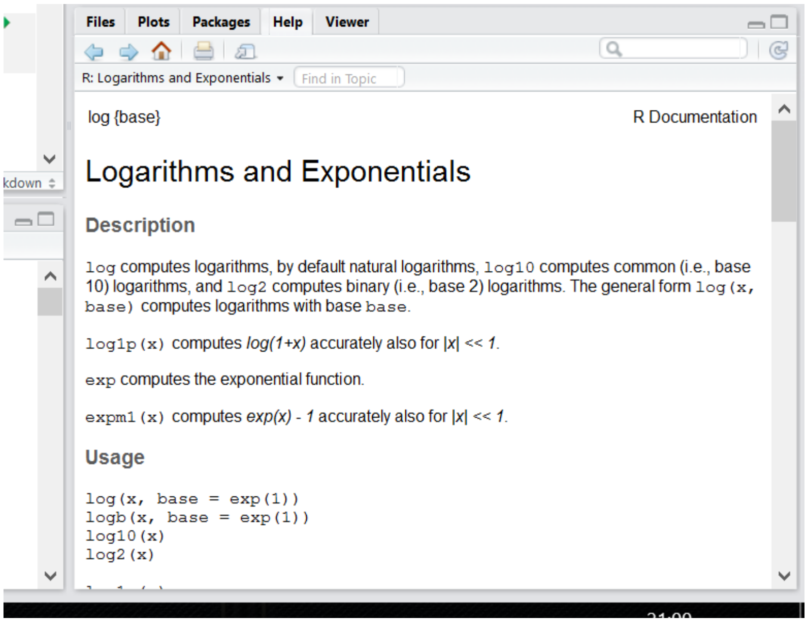
$2^3 = 8$

- .

, , :

```
?log
```

RStudio :



log {base} R Documentation

Logarithms and Exponentials

Description

`log` computes logarithms, by default natural logarithms, `log10` computes common (i.e., base 10) logarithms, and `log2` computes binary (i.e., base 2) logarithms. The general form `log(x, base)` computes logarithms with base `base`.

`log1p(x)` computes *log(1+x)* accurately also for |x| << *1*.

`exp` computes the exponential function.

`expm1(x)` computes *exp(x) - 1* accurately also for |x| << *1*.

Usage

```
log(x, base = exp(1))
logb(x, base = exp(1))
log10(x)
log2(x)
```

Figure 2.3:

, *base =*. (2.7182818...),
. . . R - —
, , ,
.

```
log(x = 8, base = 2)
```

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

… ( ):

```
log(8,2)
```

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

, :

```
log(8, sqrt(4))
```

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

. . , — R (Python). , - . R . , .

, : +, - , /, ^ . . :

```
'+'(3, 4)
```

```
## [1] 7
```

2.1.5

— . R : <- (=,). : Alt + - (option + - macOS).

```
a <- 2
a
```

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

, ! , , .

Environment RStudio:

:

```
b <- a ^ a + a * a
b
```

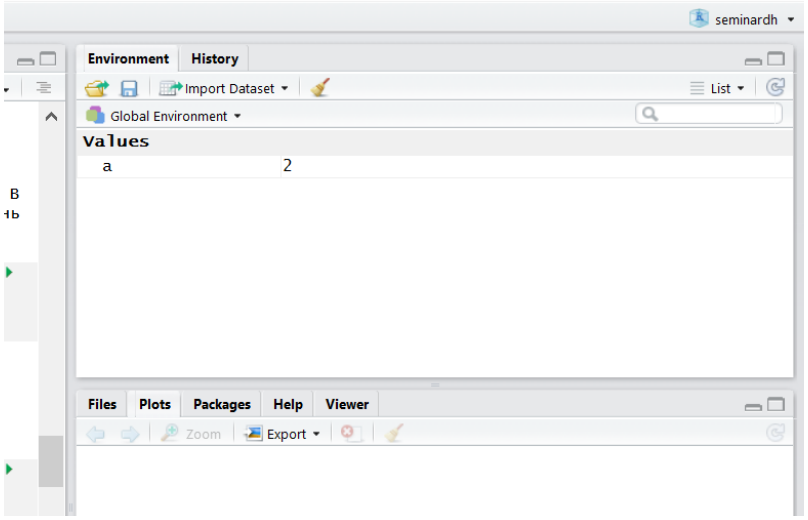
```
## [1] 8
```

```
log(b, a)
```

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

:

```
a == b
```



Values

a 2

Figure 2.4:

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

, ==, =.
.

```
a = b
a
```

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

(
)
, (=( ).

:

```
a <- 2
b <- 3

a==b
```

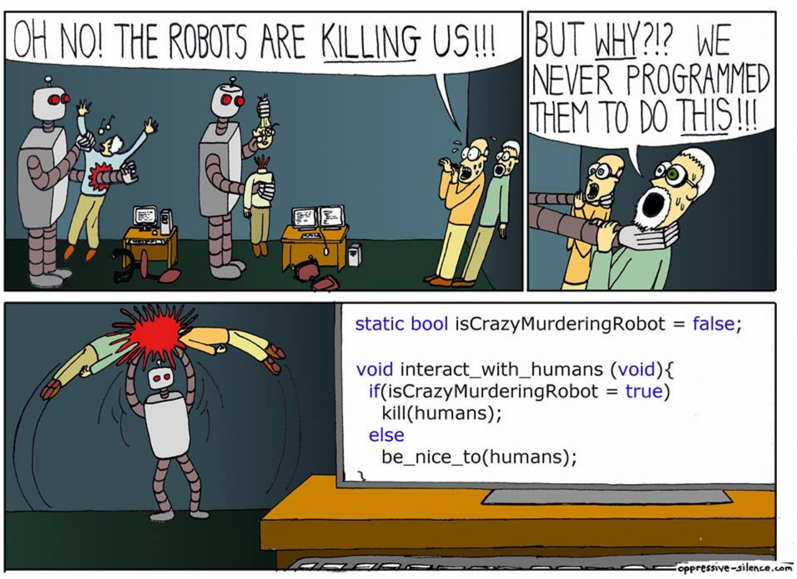
```
## [1] FALSE
```

```
a!=b
```

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

R .

/ :


OH NO! THE ROBOTS ARE KILLING US!!!
BUT WHY?!? WE NEVER PROGRAMMED THEM TO DO THIS!!!
oppressive-silence.com

```
static bool isCrazyMurderingRobot = false;

void interact_with_humans (void){
  if(isCrazyMurderingRobot = true)
    kill(humans);
  else
    be_nice_to(humans);
}
```

Figure 2.5:

```r
a > b
```

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

```r
a < b
```

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

```r
a >= b
```

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

```r
a <= b
```

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

2.2

(numeric):

```r
class(a)
```

```
## [1] "numeric"
```

, R numeric: integer ( ), double ( ), complex ( ). : `complexnumber <- 2+2i` R

, R .
:

numeric integer, R

R:

1. **character**: . . ", ' (
 , -).

```
s <- "        !"
s
```

```
## [1] "        !"
```

```
class(s)
```

```
## [1] "character"
```

2. **logical**: TRUE FALSE.

```
t1 <- TRUE
f1 <- FALSE

t1
```

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

```
f1
```

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

, T F (True False!)

```
t2 <- T
f2 <- F
```

, R TRUE FALSE, T F

```
TRUE <- FALSE
```

```
## Error in TRUE <- FALSE:          (do_set)
```

```
TRUE
```

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

```
T <- FALSE
T
```

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

, , ,
!

```
comparison <- a == b
comparison
```

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

, : -
, .

, .
— (!):

```
t1
```

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

```
!t1
```

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

```
!!t1 #           !
```

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

(TRUE TRUE):

```
t1 & t2
```

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

```
t1 & f1
```

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

(TRUE TRUE):

```
t1 | f1
```

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

```
f1 | f2
```

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

, (!) . .
!

2.3

( ), , (**atomic vector** **atomic**) — ( ) .

, .
{0,0} - , , {2,3}:

.
, {2,3}:

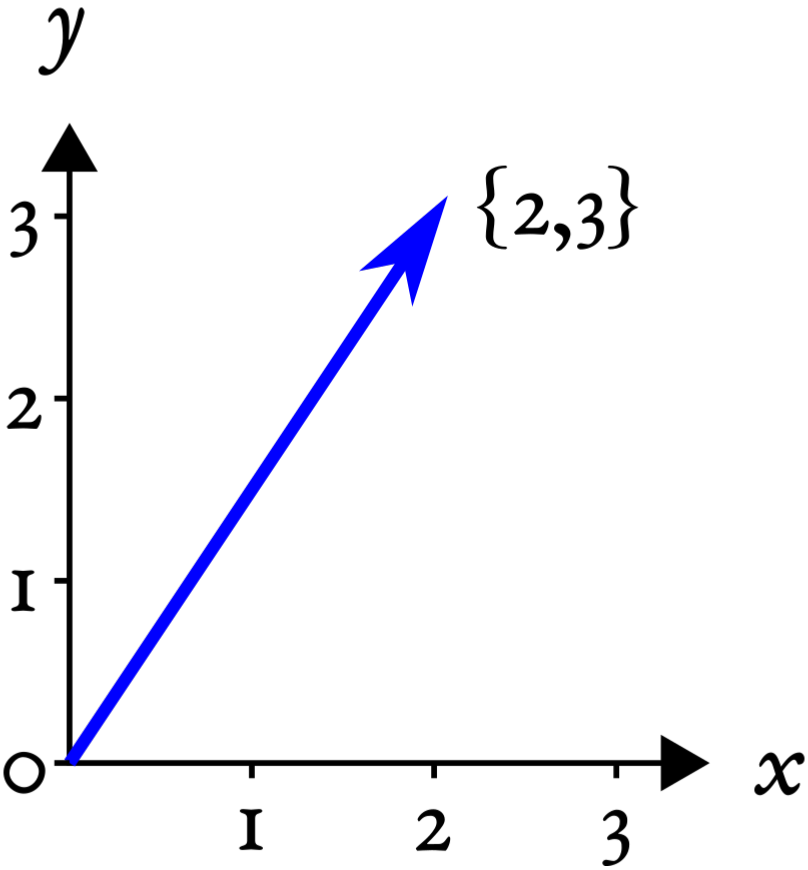


Figure 2.6:

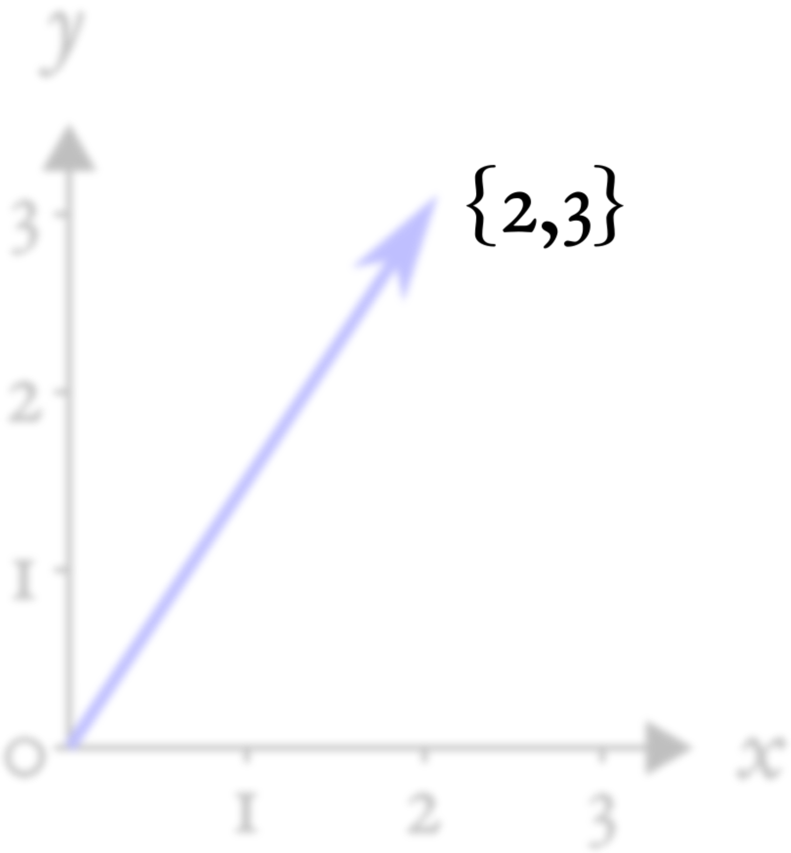


Figure 2.7:

, R, , , . , R (. .) , **1.** !

, *c()*:

```
c(4,8,15,16,23,42)
```

```
## [1]  4  8 15 16 23 42
```

numeric, *character* *logical*:

```
c("  ", "  ", " ")
```

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

```
c(TRUE, FALSE)
```

```
## [1]  TRUE FALSE
```

—

c . ? . R .

:

```
1:10
```

```
##  [1]  1  2  3  4  5  6  7  8  9 10
```

```
5:-3
```

```
## [1]  5  4  3  2  1  0 -1 -2 -3
```

1. , … , seq():

```
seq(10,100, by = 10)
```

```
##  [1]  10  20  30  40  50  60  70  80  90 100
```

, , . seq() :

```
seq(1,13, length.out = 4)
```

```
## [1]  1  5  9 13
```

— rep() — . — , , — .

```
rep(1, 5)
```

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

, !

```
rep(1:3, 3)
```

```
## [1] 1 2 3 1 2 3 1 2 3
```

```r
rep(1:3, 1:3)
```

```
## [1] 1 2 2 3 3 3
```

(, , , 1):

```r
v1 <- c("Hey", "Ho")
v2 <- c("Let's", "Go!")
c(v1, v2)
```

```
## [1] "Hey"   "Ho"    "Let's" "Go!"
```

R . , `sum()` ( ) `mean()` ( ):

```r
sum(1:10)
```

```
## [1] 55
```

```r
mean(1:10)
```

```
## [1] 5.5
```

2.3.1

, ? ?

, *atomic* . . R " ", " " .

:

```r
c(FALSE, 2)
```

```
## [1] 0 2
```

`FALSE` `0` (`TRUE` `1`), . :

```r
2 + TRUE
```

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

(implicit coercion).

:

```r
c(TRUE, 3, "   ")
```

```
## [1] "TRUE"   "3"      "   "
```

R :

NULL < raw < logical < integer < double < complex < character < list < expression.

, , — TRUE FALSE — 0 1 , 0 1 "0" "1".

, as. (**explicit coercion**):

```
as.numeric(c(T, F, F))
```

```
## [1] 1 0 0
```

```
as.character(as.numeric(c(T, F, F)))
```

```
## [1] "1" "0" "0"
```

, , . , NA — ( ).

```
as.numeric(c("1", "2", "  "))
```

```
## Warning:                          NA
```

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

— sum() mean() TRUE . !

2.3.2

, , :

```
n <- 1:4
m <- 4:1
n + m
```

```
## [1] 5 5 5 5
```

```
n - m
```

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

```
n * m
```

```
## [1] 4 6 6 4
```

```
n / m
```

```
## [1] 0.2500000 0.6666667 1.5000000 4.0000000
```

```
n ^ m + m * (n - m)
```

```
## [1] -11   5  11   7
```

,
. (**vectorization**).

- MATLAB ,
`m*n` (*dot product*), .
`%` :

```
n %*% m
```

```
##      [,1]
## [1,]   20
```

R, .

, R, , .
, , (
). “ ”
:

```
sqrt(1:10)
```

```
##  [1] 1.000000 1.414214 1.732051 2.000000 2.236068 2.449490 2.645751 2.828427
##  [9] 3.000000 3.162278
```

R .
(C, C++, FORTRAN), ,
, .

R — ,
. ,
`for` `while` ??. !
. , —
`for` `while` ??.

2.3.3 Recycling

- . , ,
. ? !
(, *recycling rule*). ,
, ,
:

```
n <- 1:4
m <- 1:2
n * m
```

```
## [1] 1 4 3 8
```

, ? :
1 , , :

```
n * 2
```

```
## [1] 2 4 6 8
```

(, 3, 4), R ,
.

```
n + c(3,4,5)
```

```
## Warning in n + c(3, 4, 5):
##
```

```
## [1] 4 6 8 7
```

, .
. . . —
, .

2.3.4

, . . , ,
R- !

. , R —
(,) - . [] (
— !).

— , . . .

```
n <- 1:10
n[1]
```

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

```
n[10]
```

```
## [1] 10
```

(MATLAB,)
, 0 — ( ),
R . — 1,
— length(). .

, :

```
n[3] <- 20
n
```

```
##  [1]  1  2 20  4  5  6  7  8  9 10
```

, :

```
n[4:7]
```

```
## [1] 4 5 6 7
```

```
n[10:1]
```

```
##  [1] 10  9  8  7  6  5  4 20  2  1
```

:

```
n[-1]
```

```
## [1]  2 20  4  5  6  7  8  9 10
```

```
n[c(-4, -5)]
```

```
## [1]  1  2 20  6  7  8  9 10
```

, . :

```
n[c(T,F,T,F,T,F,T,F,T,F)]
```

```
## [1]  1 20  5  7  9
```

, !

```
n[c(T,F)] #        - recycling rule!
```

```
## [1]  1 20  5  7  9
```

, : . ,
( ) :

```
my_named_vector <- c(first = 1, second = 2, third = 3)
my_named_vector['first']
```

```
## first
##     1
```

“ ” names() .

```
d <- 1:4
names(d) <- letters[1:4]
d["a"]
```

```
## a
## 1
```

letters — “ ” R — a z. !
, LETTERS — , .
pi.

n:

```
mean(n)
```

```
## [1] 7.2
```

, ?

— :

```
larger <- n>mean(n)
larger
```

```
##  [1] FALSE FALSE  TRUE FALSE FALSE FALSE FALSE  TRUE  TRUE  TRUE
```

n:

```
n[larger]
```

```
## [1] 20  8  9 10
```

:

```
n[n>mean(n)]
```

```
## [1] 20  8  9 10
```

, R: (subset)

.

2.3.5 NA -

- . , - - . R NA (*Not Available* -). NA — "NA", 0, "" FALSE. NA — NA.

, NA NA:

```
missed <- NA
missed == "NA"
```

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

```
missed == ""
```

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

```
missed == NA
```

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

: NA c NA NA!

NA :

```r
n[5] <- NA
n
```

```
##  [1]  1  2 20  4 NA  6  7  8  9 10
```

```r
mean(n)
```

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

?
, NA . :

```r
n == NA
```

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

, , NA c NA NA…

, is.na():

```r
is.na(n)
```

```
##  [1] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE
```

is.na(n) FALSE , TRUE , NA. . ! ( ), :

```r
n[!is.na(n)]
```

```
## [1]  1  2 20  4  6  7  8  9 10
```

, !

```r
mean(n[!is.na(n)])
```

```
## [1] 7.444444
```

, (!)

, , NA. *mean()*:

```r
?mean()
```

na.rm =, FALSE. , !

```r
mean(n, na.rm = T)
```

```
## [1] 7.444444
```

!

> NA . , , NA: NA, NA_integer_, NA_real_, NA_complex_ and NA_character_.

NA NaN — . NaN *Not a Number*
0 / 0. , is.na() TRUE
NaN, is.nan() TRUE NaN FALSE NA:

```
is.na(NA)
```

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

```
is.na(NaN)
```

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

```
is.nan(NA)
```

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

```
is.nan(NaN)
```

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

2.3.6 —

, , , … !

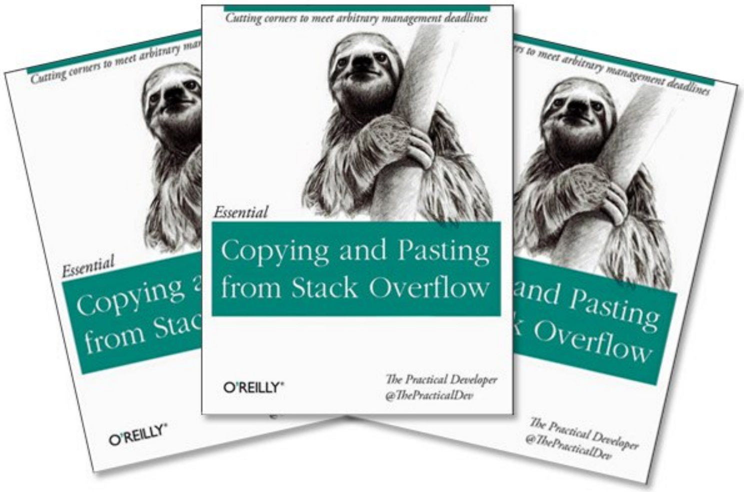

Cutting corners to meet arbitrary management deadlines
Essential
Copying and Pasting from Stack Overflow
O'REILLY
The Practical Developer
@ThePracticalDev

Figure 2.8:

, . .
Stackoverflow[4] R- !

[4] Stackoverflow — . *Quora*, *The Question*, *Mail.ru*
.

Computer Programming To Be Officially Renamed “Googling Stack Overflow”Source: http://t.co/xu7acfXvFF pic.twitter.com/iJ9k7aAVhd

— Stack Exchange (**?**) July 20, 2015

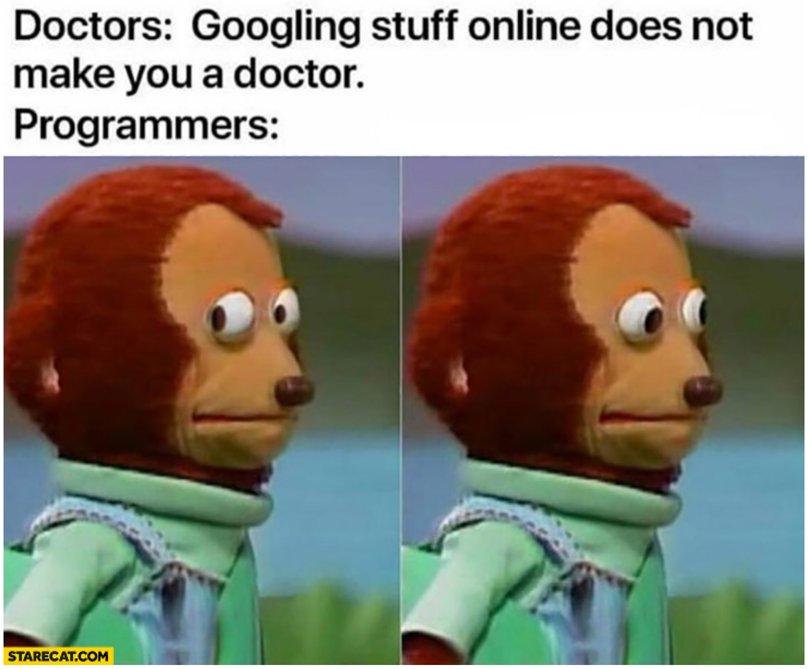


Figure 2.9:

, : . , .
- .

, — .

Does anyone ever get good at R or do they just get good at googling how to do things in R

— Lauren M. Seyler, Ph.D. (**?**) May 6, 2019

’ - . , —
R. , . , — .
: (**list**) (**matrix**).
“ ” , . R —
data.frame.

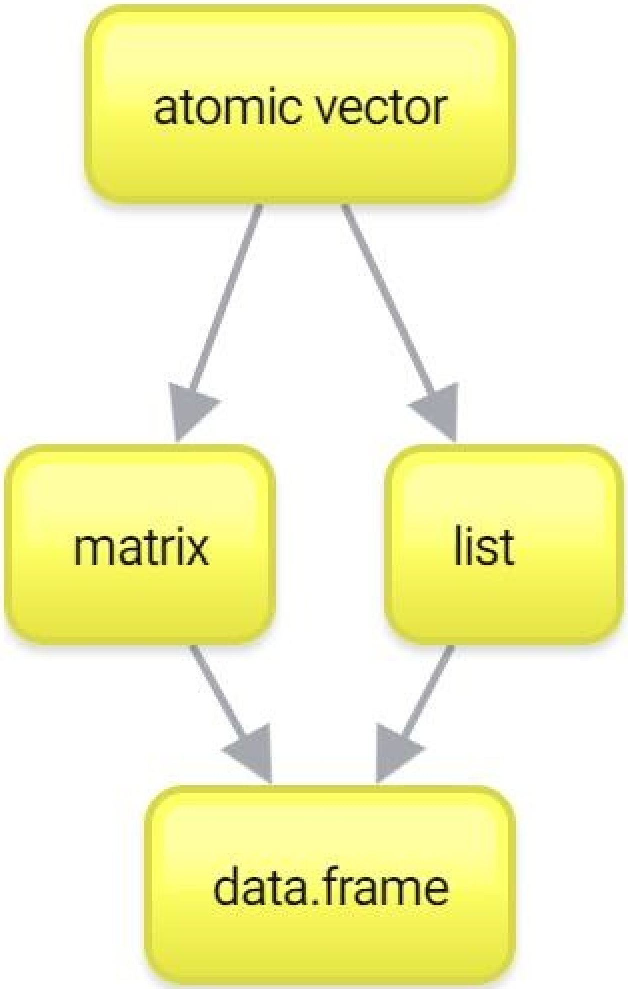


Figure 2.10:

2.4 (matrix)

, . — “ ” : , , . matrix() , .

```
A <- matrix(1:20, nrow=5,ncol=4)
A
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    6   11   16
## [2,]    2    7   12   17
## [3,]    3    8   13   18
## [4,]    4    9   14   19
## [5,]    5   10   15   20
```

, :

```
A <- matrix(1:20, nrow=5)
A
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    6   11   16
## [2,]    2    7   12   17
## [3,]    3    8   13   18
## [4,]    4    9   14   19
## [5,]    5   10   15   20
```

: . — , . .

```
A[2,3]
```

```
## [1] 12
```

```
A[2:4, 1:3]
```

```
##      [,1] [,2] [,3]
## [1,]    2    7   12
## [2,]    3    8   13
## [3,]    4    9   14
```

— , — . , / , :

```
A[, 1:3]
```

```
##      [,1] [,2] [,3]
## [1,]    1    6   11
## [2,]    2    7   12
## [3,]    3    8   13
## [4,]    4    9   14
## [5,]    5   10   15
```

```
A[2:4, ]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    7   12   17
## [2,]    3    8   13   18
## [3,]    4    9   14   19
```

```
A[, ]
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    6   11   16
## [2,]    2    7   12   17
## [3,]    3    8   13   18
## [4,]    4    9   14   19
## [5,]    5   10   15   20
```

, :

```
A[2:4, 2:4] <- 100
A
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    6   11   16
## [2,]    2  100  100  100
## [3,]    3  100  100  100
## [4,]    4  100  100  100
## [5,]    5   10   15   20
```

, , . R , , , MATLAB. : .

, — , : R — dim ( ) dimnames. — , “ ”. . : dim . . 99-101 “R in a Nutshell” (Adler, 2010).

2.5 (list)

. !

```
l <- list(42, "    ", T)
l
```

```
## [[1]]
## [1] 42
##
## [[2]]
## [1] "     "
##
## [[3]]
## [1] TRUE
```

, , !

```
lbig <- list(c("Wow", "this", "list", "is", "so", "big"), "16", l)
lbig
```

```
## [[1]]
## [1] "Wow"  "this" "list" "is"   "so"   "big"
##
## [[2]]
## [1] "16"
##
## [[3]]
## [[3]][[1]]
## [1] 42
##
## [[3]][[2]]
## [1] "     "
##
## [[3]][[3]]
## [1] TRUE
```

, , , , str():

```
str(lbig)
```

```
## List of 3
##  $ : chr [1:6] "Wow" "this" "list" "is" ...
##  $ : chr "16"
##  $ :List of 3
##   ..$ : num 42
##   ..$ : chr "     "
##   ..$ : logi TRUE
```

, - . - - .

:

```
namedl <- list(age = 24, PhDstudent = T, language = "Russian")
namedl
```

```
## $age
## [1] 24
##
## $PhDstudent
## [1] TRUE
##
## $language
## [1] "Russian"
```

, . :

```
namedl$age
```

```
## [1] 24
```

, . , :

```
namedl[1]
```

```
## $age
## [1] 24
```

, , — , . . :

```
class(namedl)
```

```
## [1] "list"
```

```
class(namedl[1])
```

```
## [1] "list"
```

(-) ,
:

```
namedl[[1]]
```

```
## [1] 24
```

```
class(namedl[[1]])
```

```
## [1] "numeric"
```

Indexing lists in #rstats. Inspired by the Residence Inn pic.twitter.com/YQ6axb2w7t

— Hadley Wickham (**?**) September 14, 2015

, .

```
namedl[['age']]
```

```
## [1] 24
```

— , $.

R, , Python. R,
, —

. , —
, .
.

2.6 Data.frame

, . - . (**data.frames**). , ,
.

. , .

```
name <- c("Ivan", "Eugeny", "Lena", "Misha", "Sasha")
age <- c(26, 34, 23, 27, 26)
student <- c(F, F, T, T, T)
df <- data.frame(name, age, student)
df
```

```
##      name age student
## 1    Ivan  26   FALSE
## 2 Eugeny  34   FALSE
## 3    Lena  23    TRUE
## 4   Misha  27    TRUE
## 5   Sasha  26    TRUE
```

```
str(df)
```

```
## 'data.frame':    5 obs. of  3 variables:
##  $ name   : chr  "Ivan" "Eugeny" "Lena" "Misha" ...
##  $ age    : num  26 34 23 27 26
##  $ student: logi  FALSE FALSE TRUE TRUE TRUE
```

, , ? , — - ,
atomic . , “ ”.
, “ ” 90 . , ,
. (!),
, . , :
— character, — numeric, — logical. ,
, :

```
df$age[2:3]
```

```
## [1] 34 23
```

age $. ,
, 2 3.

$:

```
df$lovesR <- T #     recycling -     ?
df
```

```
##      name age student lovesR
## 1    Ivan  26   FALSE   TRUE
## 2 Eugeny  34   FALSE   TRUE
## 3    Lena  23    TRUE   TRUE
## 4   Misha  27    TRUE   TRUE
## 5   Sasha  26    TRUE   TRUE
```

, :

```
df[3:5, 2:3]
```

```
##   age student
## 3  23    TRUE
## 4  27    TRUE
## 5  26    TRUE
```

, , — .

:

```
df[1:2,"age"]
```

```
## [1] 26 34
```

! , R , :

```
df[df$age < mean(df$age), 4]
```

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

:

```
df$lovesR[df$age < mean(df$age)]
```

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

```
df[df$age < mean(df$age), 'lovesR']
```

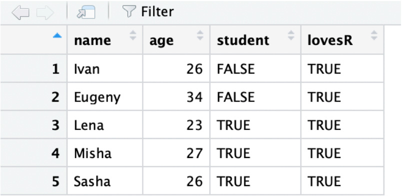
```
## [1] TRUE TRUE TRUE TRUE
```

— , .

RStudio. View(df) (Environment). , Excel . , ...[5]

, , , . . .

[5] , , .
.



	name	age	student	lovesR
1	Ivan	26	FALSE	TRUE
2	Eugeny	34	FALSE	TRUE
3	Lena	23	TRUE	TRUE
4	Misha	27	TRUE	TRUE
5	Sasha	26	TRUE	TRUE

Figure 2.11:

.

2.7

, . ( ) , , “ ” . . , , ...

2.7.1

, , “ ”. (, read.csv() —), , :

```
read.csv("character-deaths.csv")
```

```
## Warning in file(file, "rt"):                  'character-deaths.csv': No
## such file or directory
```

```
## Error in file(file, "rt"):
```

, R . - , . - , R . .

- : .

, getwd() ( ), . :

```
got <- read.csv("character-deaths.csv")
```

- : .

setwd() , , . :

```
got <- read.csv("character-deaths.csv")
```

. , .

- : .

```
got <- read.csv("/Users/Username/Some_Folder/character-deaths.csv")
```

, , .

Windows : / R, //.

- : *Import Dataset.*

  Environment RStudio *Import Dataset.* , — , , .

- .

  R, , , . :

```
got <- read.csv("https://raw.githubusercontent.com/Pozdniakov/stats/master/data/charac
```

- : *RStudio.*

`File - New Project...`, `New Directory`, `New Project`, `Directory Name` `Create Project`.

- , . . _, , , . — , . — , , `data` . .

  , , Git. RStudio . — `File - New Project...` `Version Control`, Git Subversion . RStudio . Git RStudio.

2.7.2

, — R. , — . — , .

. .
. , , .
? : .

, “ ” :
, . .
: , - -
.

read.table().
: ! read.csv(), read.csv2()
— , read.table(), ,
, . .csv, *Comma Separated*
Values (,). , “ ” :
, . : (. .
) , ,
. “ ” .csv, (;), —
(,). read.csv() read.csv2() —
“ ” , — ( ) “ ”.

— , read.csv()
read.csv2() .

, . file =, ,
stringsAsFactors = FALSE:

```
got <- read.csv("data/character-deaths.csv", stringsAsFactors = FALSE)
```

, read.table() *character*
(*factor*). , — , character,
. - : ,
“male” “female”, 1 2, ,
1 2. ,
.
character, *factor*, .
.
.

View(got): ! -
— . *Help*.

.csv . , .tsv — , .csv,
— . read.delim() read.delim2().
, , read.table().
(, fread() data.table — !),
“ ” — . .
.

, .
— *Microsoft Excel*. .xlsx ,

. ,
.csv.

: *readxl, xlsx, openxlsx.*
SPSS, Stata, SAS *foreign.*
.

Bibliography

Adler, J. (2010). *R in a nutshell: A desktop quick reference.* " O'Reilly Media, Inc.".