Icahn
School of
Medicine at
**Mount
Sinai**

Introduction to coding with R

Part I

Joselyn Chávez

01/10/2023

Let's recap

- R and RStudio
- RStudio panels
- What do we do in the console?
- How to create a script?
- Where to visualize existing variables?
- How to read/import a table?
- How to visualize the table?
- How to write/export a table?

Data structures in R

- Vectors
- Matrices
- Data frames
- Lists
- Functions

Choosing a good variable name

- Be clear and concise.
- Preferably using lowercase.
- Not contain special characters. Avoid dieresis and other accents (é, è, â, î or ô, tilde ñ, ü or ï)
- Use _ or Upper/Lower case to separate words, never space.
- Avoid conflict with any base R keywords (True, False, and, if, or else or other function names)

Let's try

What of these variable names follow good practices?

a) MY_FIRST_VARIABLE

b) OxygenLevel

c) patient_name

d) final.value

e) mean

Vectors

Creating a vector

Using the assignment operator

For one value

```
my_vector <- 10
my_vector <- "a"
```

Using the combine function

For two or more values

```
my_vector <- c(1,10,25,30)
my_vector
```

```
## [1]  1 10 25 30
```

```
my_vector <- c("a","b","c")
my_vector
```

```
## [1] "a" "b" "c"
```

Let's practice

- Create a variable called vector_1 that contains the number 500
- Create a variable called vector_2 that contains the numbers 1:500
- Create a variable called vector_3 that contains the letters "a" to "e"
- Create a variable called vector_4 that contains your name, age, and the city where you were born.

Using the seq() function

```
my_vector <- seq(1:10)
my_vector
```

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

```
my_vector <- seq(from = 0, to = 10, by = 2)
my_vector
```

```
## [1]  0  2  4  6  8 10
```

- Write a vector with the numbers 1 to 500 in steps of 10

Vector features

- Vectors have only one dimension (length)

```
my_vector <- c(1,2,3,4)
length(my_vector)
```

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

- All vector components must be the same type
  - Numeric
  - Integer
  - Double
  - Character
  - Factor
  - Logical

- Numeric

```
x_num <- c(1, 2, 3)
class(x_num)
```

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

- Integer

```
x_int <- c(1L, 2L, 3L)
class(x_int)
```

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

- Double

```
x_dbl <- c(1, 2.5, 3.1)
typeof(x_dbl)
```

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

- Character

```
x_chr <- c("a", "chair", "the window")
class(x_chr)
```

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

- Factor

```r
x_fct <- factor("mouse_a","mouse_b","mouse_c")
class(x_fct)
```

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

- Logical

```r
x_log <- c(TRUE, FALSE, TRUE)
class(x_log)
```

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

- R finds a way to unify data type when there is more than one per vector

```
x <- c(1, "a", TRUE)
x
```

```
## [1] "1"    "a"    "TRUE"
```

```
class(x)
```

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

What will be the class of these vectors?

y <- c(5, 7, "airplanes")

z <- c(10, 30.5, TRUE)

Converting one class to another using as. functions

```
x <- c(1.9, 2, 0, 0)
class(x)
```

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

```
as.double(x)
```

```
## [1] 1.9 2.0 0.0 0.0
```

```
as.integer(x)
```

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

Converting one class to another using as. functions

```r
as.character(x)
```

```
## [1] "1.9" "2"   "0"   "0"
```

```r
as.factor(x)
```

```
## [1] 1.9 2   0   0  
## Levels: 0 1.9 2
```

```r
as.logical(x)
```

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

Missing values

- NA

```r
x <- c(1, "a", TRUE, NA)
x
```

```
## [1] "1"    "a"    "TRUE" NA
```

- NaN

```r
x <- c(10, -1, NA)
log(x)
```

```
## [1] 2.302585      NaN       NA
```

How do we access the vector elements?

Using an integer as index

Vector index in R starts at 1

```
x <- c(10,20,30,40,50)
x
```

```
## [1] 10 20 30 40 50
```

```
x[3] # Extract the third element
```

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

```r
x <- c(10,20,30,40,50)

# Extract index from 3 to 5
x[3:5]
```

```
## [1] 30 40 50
```

```r
x <- c("a","b","c","d","e")

# Extract index 2 and 5
x[c(2,5)]
```

```
## [1] "b" "e"
```

Let's practice

- Create a vector with numbers 50 to 100 in steps of 5. Extract the first 7 numbers. Extract the last 8 numbers
- Create a vector with letters "a" to "k". Extract the letters c,d and j

Using the name as index

```
x <- c(1,3,10)
names(x)
```

```
## NULL
```

```
x <- c("number_a" = 10, "number_b" = 50, "number_c
x
```

```
## number_a number_b number_c
##       10       50      100
```

```
names(x)
```

```
## [1] "number_a" "number_b" "number_c"
```

```
x <- c(10, 50, 100)

names(x) <- c("number_a", "number_b", "number_c")
x
```

```
## number_a number_b number_c
##       10       50      100
```

```
x["number_b"]
```

```
## number_b
##       50
```

```
x[c("number_a","number_c")]
```

```
## number_a number_c
##       10      100
```

Using logical evaluation as index

```
x <- seq(1:10)
x
```

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

```
x < 5
```

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

```
x[x < 5]
```

```
## [1] 1 2 3 4
```

```r
x <- c("a","a","b","c","c","c")

x == "c"
```

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

```r
x[x == "c"]
```

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

Your turn!

The following vector shows the temperature in New York during the last week:

```r
temperature_nyc <- c("Monday" = 55,
                     "Tuesday" = 55,
                     "Wednesday" = 66,
                     "Thursday" = 50,
                     "Friday" = 48,
                     "Saturday" = 45,
                     "Sunday" = 47)
```

- Select the temperatures from the first 3 days.
- Select the temperatures from Tue, Thu and Sat.
- What days registered temperatures above the mean?

How to modify a vector?

- Adding a new element

```
x <- c("a","b","c")
x
```

```
## [1] "a" "b" "c"
```

```
x[4] <- "d"
x
```

```
## [1] "a" "b" "c" "d"
```

- Removing an element

```
x
```

```
## [1] "a" "b" "c" "d"
```

```
x[-2]
```

```
## [1] "a" "c" "d"
```

```
x <- x[-2]
x
```

```
## [1] "a" "c" "d"
```

Your turn!

Using the vector:

```
fruits <- c("apples" = 1,
            "cherries" = 10,
            "mangos" = 7)
```

- Add a new fruit to the vector
- Remove the cherries

- Replacing an element by index

```
x
```

```
## [1] "a" "c" "d"
```

```
x[1] <- "m"
x
```

```
## [1] "m" "c" "d"
```

- Replacing an element by logical evaluation

```
x
```

```
## [1] "m" "c" "d"
```

```
x[x == "d"] <- "e"
x
```

```
## [1] "m" "c" "e"
```

Excercise

Using the vector:

```r
fruits <- c("apples" = 1,
            "cherries" = 10,
            "mangos" = 7)
```

- Select all fruits with values bigger than 5
- Replace the apples number with 4

Matrices

Creating a matrix

Matrices are objects with elements arranged in a two-dimensional layout.

```r
my_matrix <- matrix(data = 1:12, nrow = 4)
my_matrix
```

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

- rows and columns
- All elements must be the same type

```r
my_matrix <- matrix(data = 1:12, ncol = 4)
my_matrix
```

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

Operations with matrices

Arithmetic operations

```
my_matrix + 10
```

```
##      [,1] [,2] [,3] [,4]
## [1,]   11   14   17   20
## [2,]   12   15   18   21
## [3,]   13   16   19   22
```

```
my_matrix * 2
```

```
##      [,1] [,2] [,3] [,4]
## [1,]    2    8   14   20
## [2,]    4   10   16   22
## [3,]    6   12   18   24
```

Arithmetic operations

```
matrix1 <- matrix(1:6, nrow = 2, ncol = 3)
matrix2 <- matrix(7:12, nrow = 2, ncol = 3)
```

```
matrix1
```

```
##      [,1] [,2] [,3]
## [1,]    1    3    5
## [2,]    2    4    6
```

```
matrix2
```

```
##      [,1] [,2] [,3]
## [1,]    7    9   11
## [2,]    8   10   12
```

```
matrix1 + matrix2
```

```
##      [,1] [,2] [,3]
## [1,]    8   12   16
## [2,]   10   14   18
```

Your turn!

- Create a matrix with numbers 101:125, arrange them in 5 rows.
- Add 10 units, then multiply by 3

How do we access matrices elements?

```
matrix1 <- matrix(101:125, nrow = 5)
matrix1
```

```
##      [,1] [,2] [,3] [,4] [,5]
## [1,]  101  106  111  116  121
## [2,]  102  107  112  117  122
## [3,]  103  108  113  118  123
## [4,]  104  109  114  119  124
## [5,]  105  110  115  120  125
```

```
matrix1[1:7]
```

```
## [1] 101 102 103 104 105 106 107
```

How do we access matrices elements?

```
matrix1[1,2]
```

```
## [1] 106
```

```
matrix1[1:2,1:2]
```

```
##      [,1] [,2]
## [1,]  101  106
## [2,]  102  107
```

Your turn!

- How would you select the number 120 from the matrix?
- How would you select the last two columns?

Some extra information about packages

Two main repositories

CRAN

- More than 19000 packages
- Topics: Statistics, machine learning, plotting, economy, spatial data, databases, phylogenetics, natural language processing, ...

- https://cran.rstudio.com > Packages

- How do we install packages from CRAN?

```
install.packages("ggplot2")
```

Two main repositories

Bioconductor

- Software, Annotation, and Experiment packages
- 2183 software packages

- https://bioconductor.org

- How do we install packages from Bioconductor?

```
install.packages("BiocManager")
BiocManager::install("Biostrings")
```

Installing packages from the source

- Main source of code: GitHub
- Under development packages, small packages, in preparation for submitting to CRAN or Bioconductor...

- How do we install packages from GitHub?

```
install.packages("remotes")
remotes::install_github("nstrayer/datadrivencv")
```

Thanks!

Ilustration by Allison Horst