R for Data Analysis in Agrobiodiversity 1

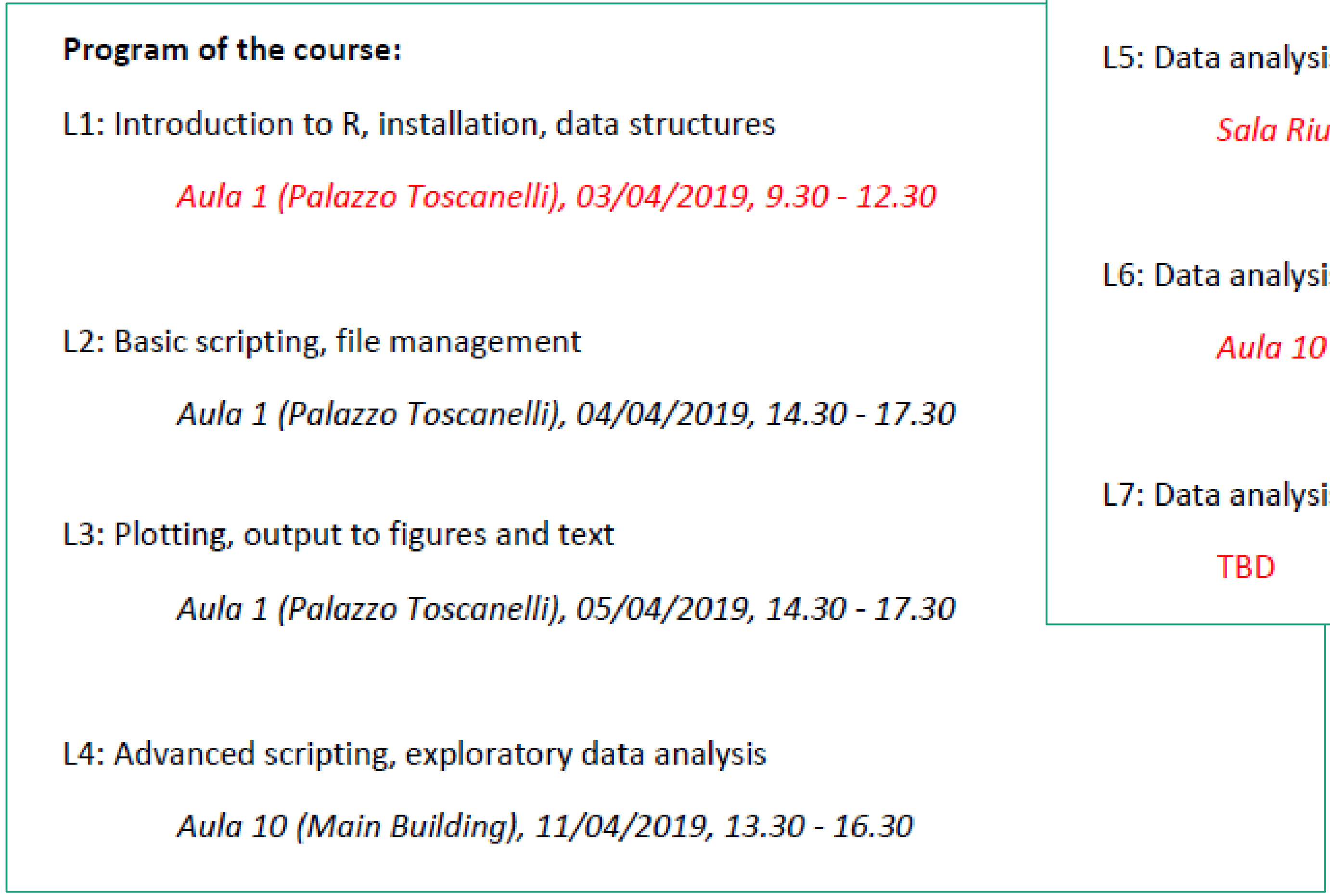
Intro and data types

Introduction to the course

- The course is organized in 7 lessons (+exam)
- 20 hours total (2 CFU)
- Attendance to 70% of the lectures is required

- Theory + practice; we will start from the basics, and aim to data analysis. Let's see how it goes

Planned programme



Program of the course:

L1: Introduction to R, installation, data structures

Aula 1 (Palazzo Toscanelli), 03/04/2019, 9.30 - 12.30

L2: Basic scripting, file management

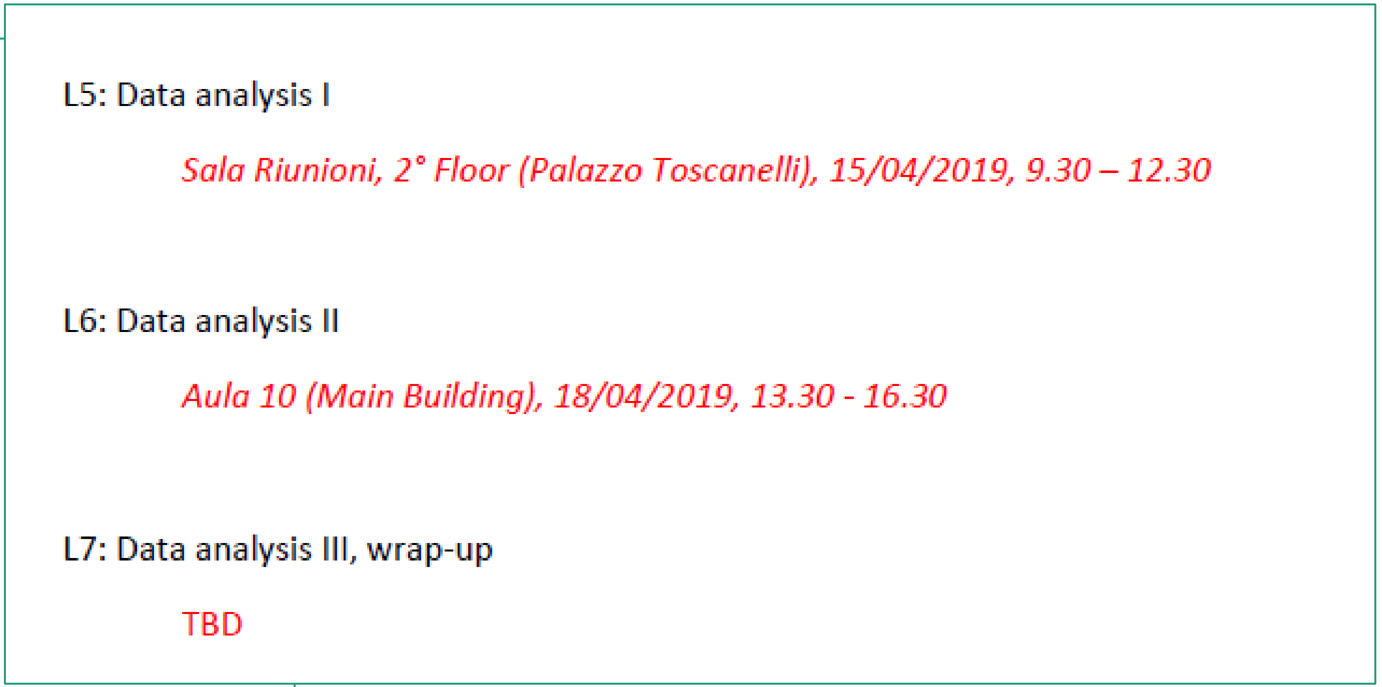
Aula 1 (Palazzo Toscanelli), 04/04/2019, 14.30 - 17.30

L3: Plotting, output to figures and text

Aula 1 (Palazzo Toscanelli), 05/04/2019, 14.30 - 17.30

L4: Advanced scripting, exploratory data analysis

Aula 10 (Main Building), 11/04/2019, 13.30 - 16.30



L5: Data analysis I

Sala Riunioni, 2° Floor (Palazzo Toscanelli), 15/04/2019, 9.30 – 12.30

L6: Data analysis II

Aula 10 (Main Building), 18/04/2019, 13.30 - 16.30

L7: Data analysis III, wrap-up

TBD

Some instructions

- Always bring your laptop, fully charged
- The program is not written in stone; let me know what you are looking for and we will try to arrange it
- Be flexible; many different levels of preparation in the room
- You will learn much more R by exercising on your own than sitting in this room
- Don’t be afraid if you loose something; scripts are there for this reason, and will be shared
- Google is your best friend; learn to ask the right questions

DON'T PANIC!

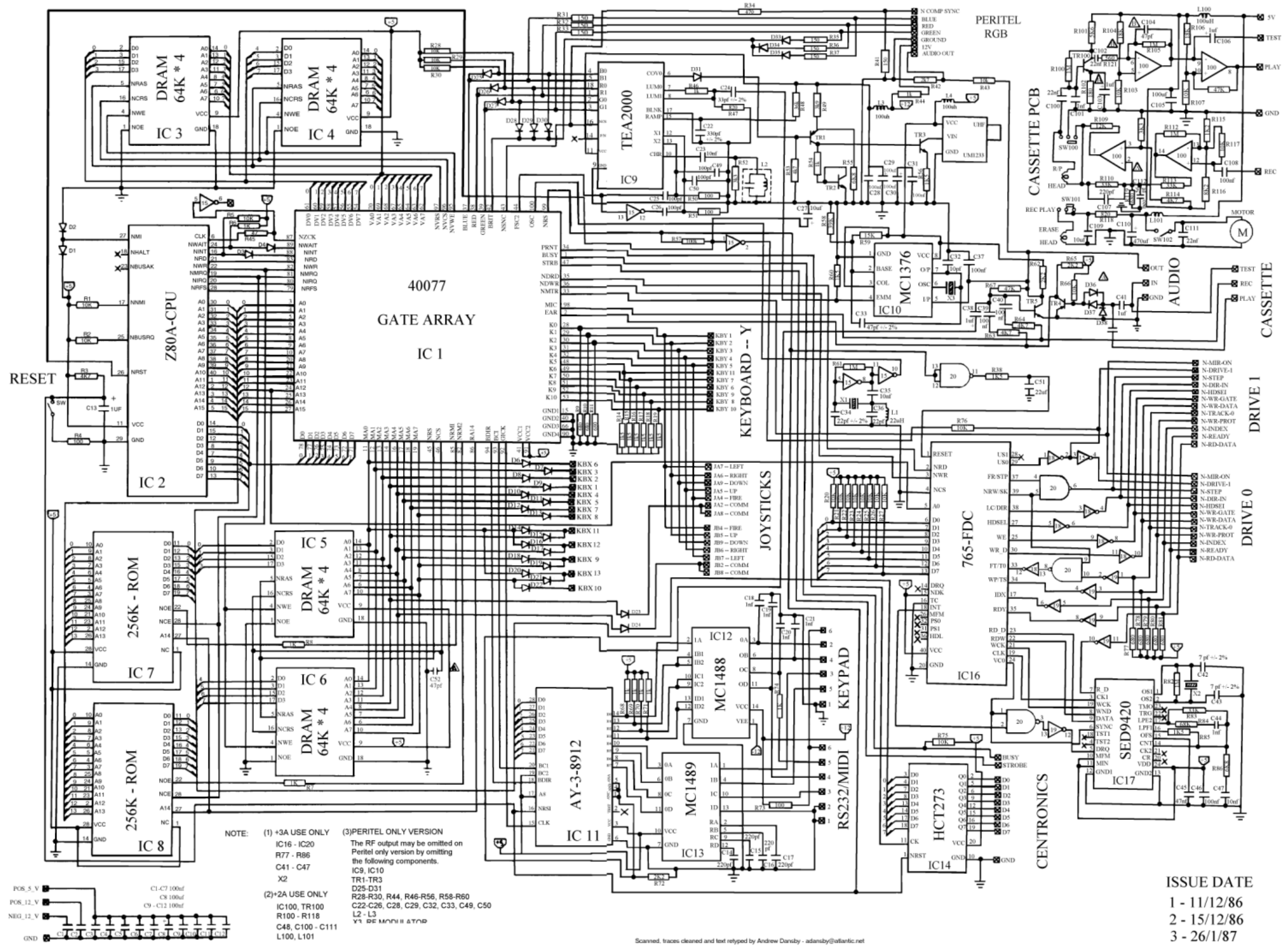
Who am I (to teach you R)?

- NOT an informatician
- Assistant Professor in crop genetics here at the Scuola
- R user for some time now, mainly using R in genomics and genetic data analysis
- Pro Google user

What I use R for

- Data manipulation, quality check
- Exploratory data analysis
- Statistical analyses
- Plotting for papers and presentations

A few notes on computers

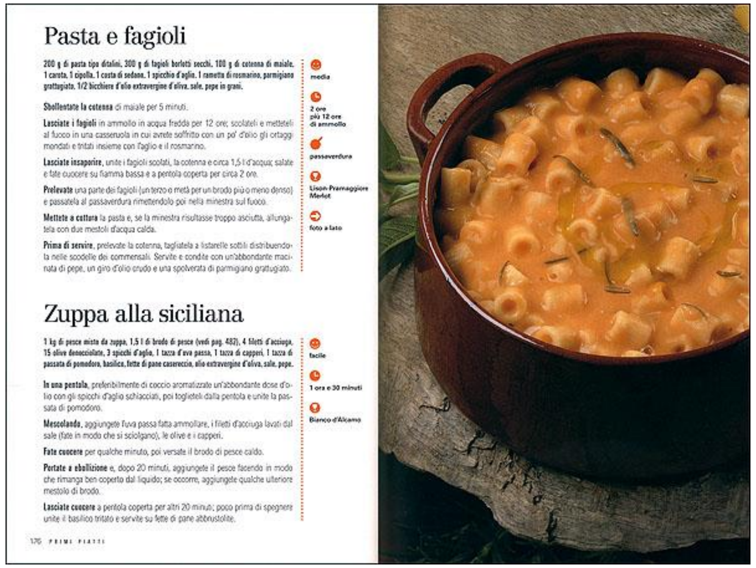


Hardware & Software

- Computer hardware are physical devices used in or with your machine
- Computer software is a collection of code run on the hardware
- SW&HW are interconnected; hardware is useless without software, software cannot be run without hardware

HW

SW



Pasta e fagioli

200 g di pasta tipo ditalini, 300 g di fagioli borlotti secchi, 100 g di cotenna di maiale, 1 carota, 1 cipolla, 1 costa di sedano, 1 spicchio d'aglio, 1 rametto di rosmarino, parmigiano grattugiato, 1/2 bicchiere d'olio extravergine d'oliva, sale, pepe in grani.

Sbollentate la cotenna di maiale per 5 minuti.

Lasciate i fagioli in ammollo in acqua fredda per 12 ore; scolateli e metteteli al fuoco in una casseruola in cui avrete soffritto con un po' d'olio gli ortaggi mondati e tritati insieme con l'aglio e il rosmarino.

Lasciate insaporire, unite i fagioli scolati, la cotenna e circa 1,5 l d'acqua; salate e fate cuocere su fiamma bassa e a pentola coperta per circa 2 ore.

Prelevate una parte dei fagioli (un terzo o metà per un brodo più o meno denso) e passatela al passaverdura rimettendolo poi nella minestra sul fuoco.

Mettete a cottura la pasta e, se la minestra risultasse troppo asciutta, allungatela con due mestoli d'acqua calda.

Prima di servire, prelevate la cotenna, tagliatela a listarelle sottili distribuendola nelle scodelle dei commensali. Servite e condite con un'abbondante macinata di pepe, un giro d'olio crudo e una spolverata di parmigiano grattugiato.

media

2 ore più 12 ore di ammollo

passaverdura

Lison-Pramaggiore Merlot

foto a lato

Zuppa alla siciliana

1 kg di pesce misto da zuppa, 1,5 l di brodo di pesce (vedi pag. 482), 4 filetti d'acciuga, 15 olive denocciolate, 3 spicchi d'aglio, 1 tazza d'uva passa, 1 tazza di capperi, 1 tazza di passata di pomodoro, basilico, fette di pane casereccio, olio extravergine d'oliva, sale, pepe.

In una pentola, preferibilmente di coccio aromatizzate un'abbondante dose d'olio con gli spicchi d'aglio schiacciati, poi toglieteli dalla pentola e unite la passata di pomodoro.

Mescolando, aggiungete l'uva passa fatta ammollare, i filetti d'acciuga lavati dal sale (fate in modo che si sciolgano), le olive e i capperi.

Fate cuocere per qualche minuto, poi versate il brodo di pesce caldo.

Portate a ebollizione e, dopo 20 minuti, aggiungete il pesce facendo in modo che rimanga ben coperto dal liquido; se occorre, aggiungete qualche ulteriore mestolo di brodo.

Lasciate cuocere a pentola coperta per altri 20 minuti; poco prima di spegnere unite il basilico tritato e servite su fette di pane abbrustolite.

facile

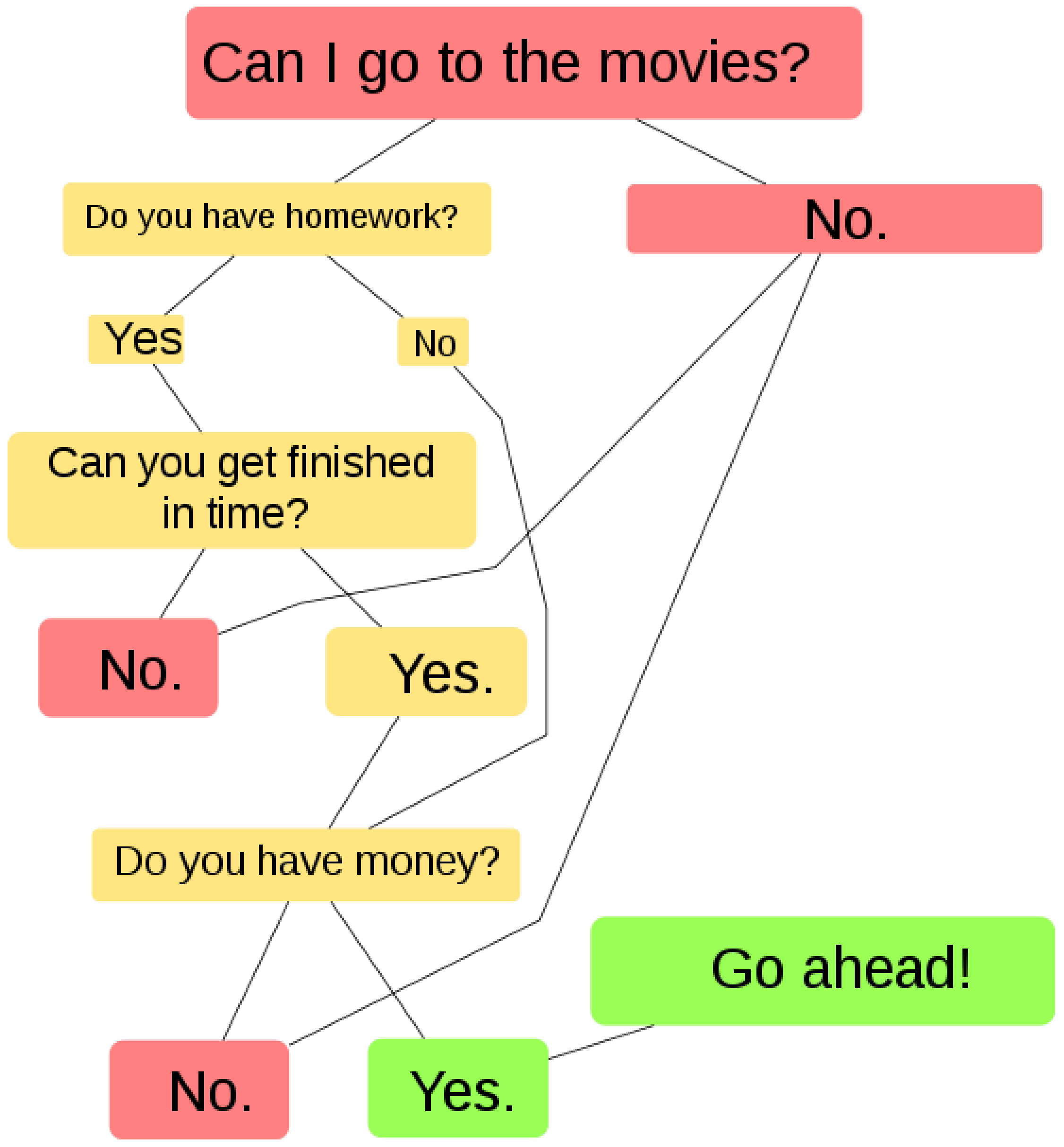
1 ora e 30 minuti

Bianco d'Alcamo

170 PRIMI PIATTI

Algorithms

- Computers are dumb; they can't do anything without being told what to do
- Algorithms are step-by-step instructions given to computers in order to solve problems and/or perform tasks



Input devices

- Hardware devices that you use to feed information to the computer

Hard disk drive (HDD)

- Stores all the data necessary to run processes
- It can be magnetic (HDD) or solid state (SSD)
- It is the same found in pen drives

Its capacity determines the **amount of data** that you can store

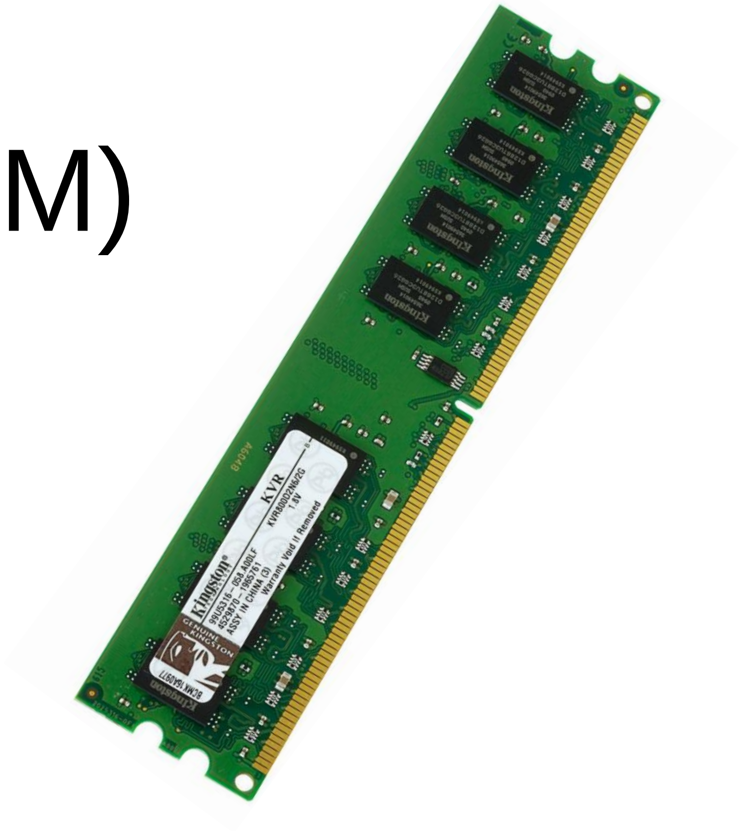
Think of it as your cabinet. The bigger it is, the more stuff you can put into it

Random access memory (RAM)

- Temporarely stores data, allowing you to work on it
- Provides quick read and write access to the storage device

Its size determines the **amount of data** you can work on **simultaneously**

Think of it as your desk; the bigger it is, the more data you can simultaneously work on



Central processing unit (CPU)

- Comes in many flavours (GPUs are also a thing)
- Responsible for processing information
- Works by logic gates: either there is electric current (1) or there isn't (0)
- It compiles any information in binary code (0/1)
- Might be installed in parallel to increase computing power

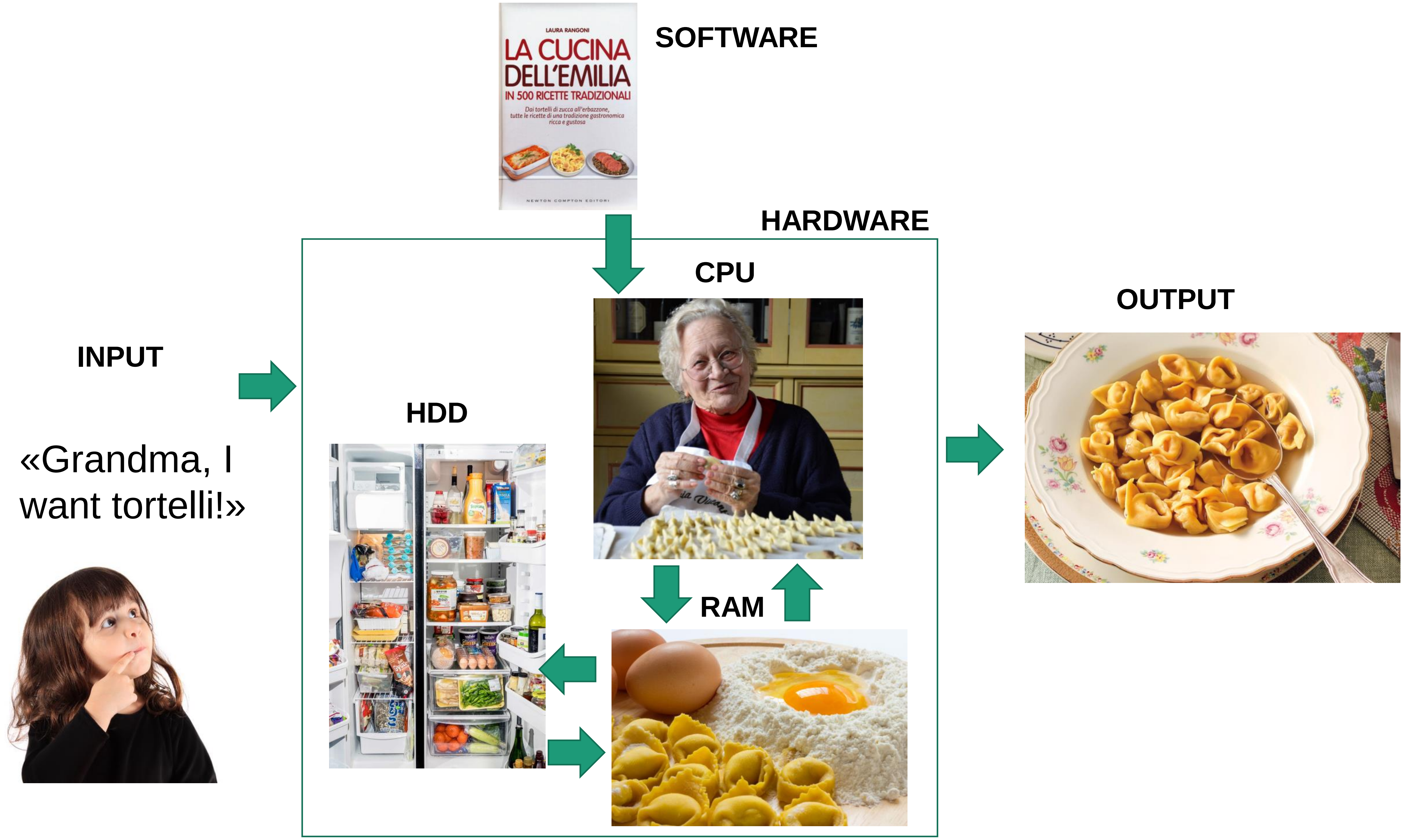
The CPU controls the **amount** of simultaneous processes that you can run and their **speed**

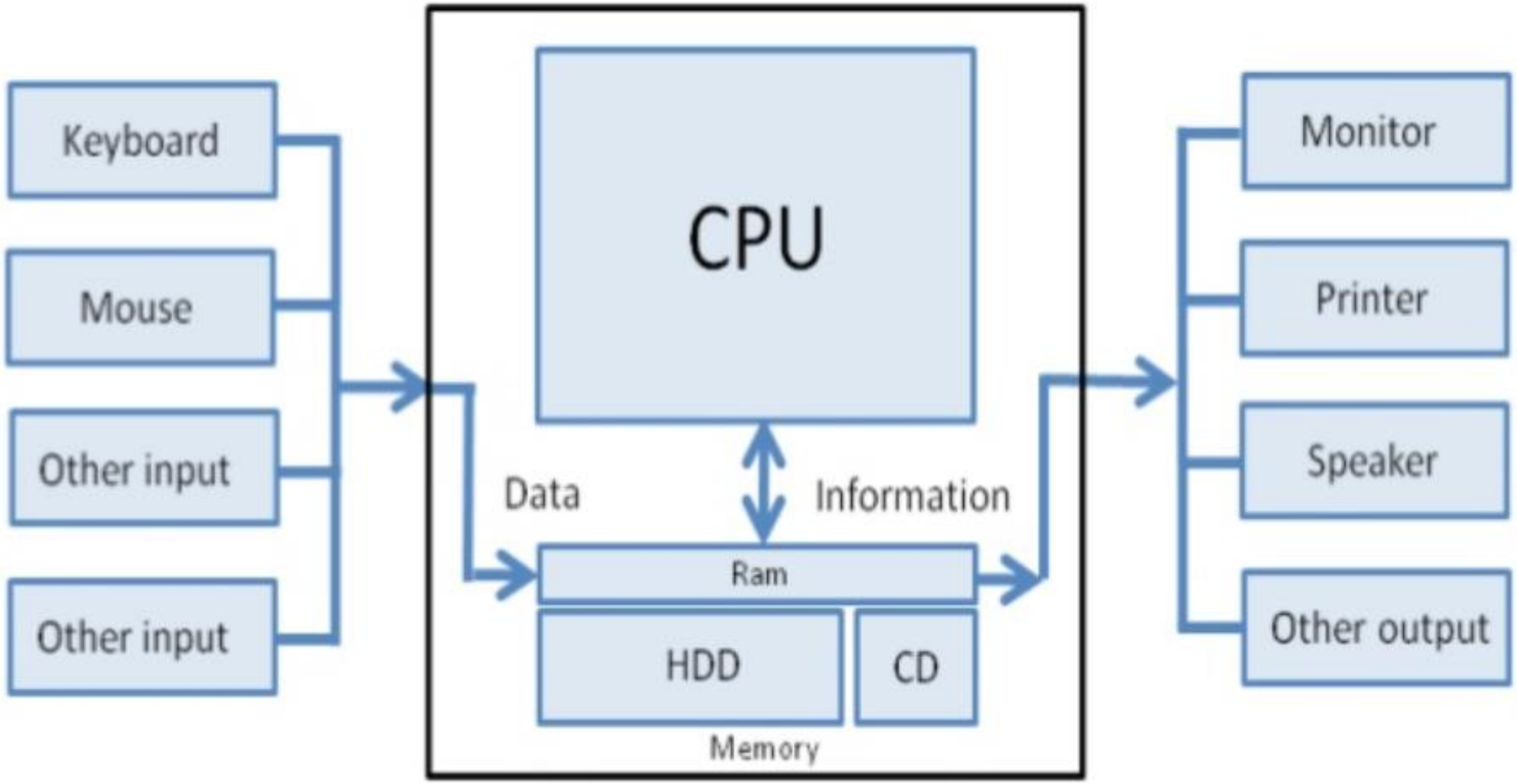
Output devices

- Devices (peripherals) that receive data from the computer and sendi it to the outer world

Motherboard

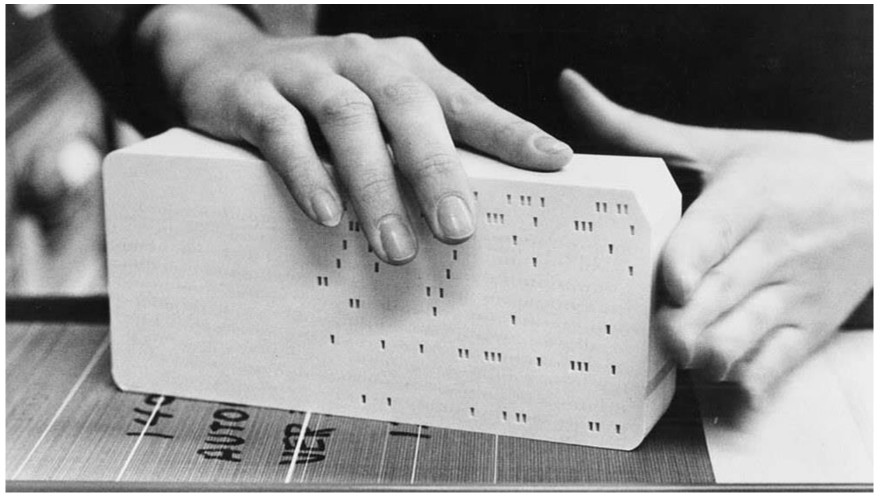
- The backbone of your computer
- Hosts all main components and puts them in relation
- Host circuits supplying power and circulating information


SOFTWARE
LAURA RANGONI
LA CUCINA DELL'EMILIA
IN 500 RICETTE TRADIZIONALI
Dai tortelli di zucca all'erbazzone, tutte le ricette di una tradizione gastronomica ricca e gustosa
NEWTON COMPTON EDITORI
HARDWARE
CPU
INPUT
«Grandma, I want tortelli!»
HDD
RAM
OUTPUT


Keyboard
Mouse
Other input
Other input
CPU
Data
Information
Ram
HDD
CD
Memory
Monitor
Printer
Speaker
Other output

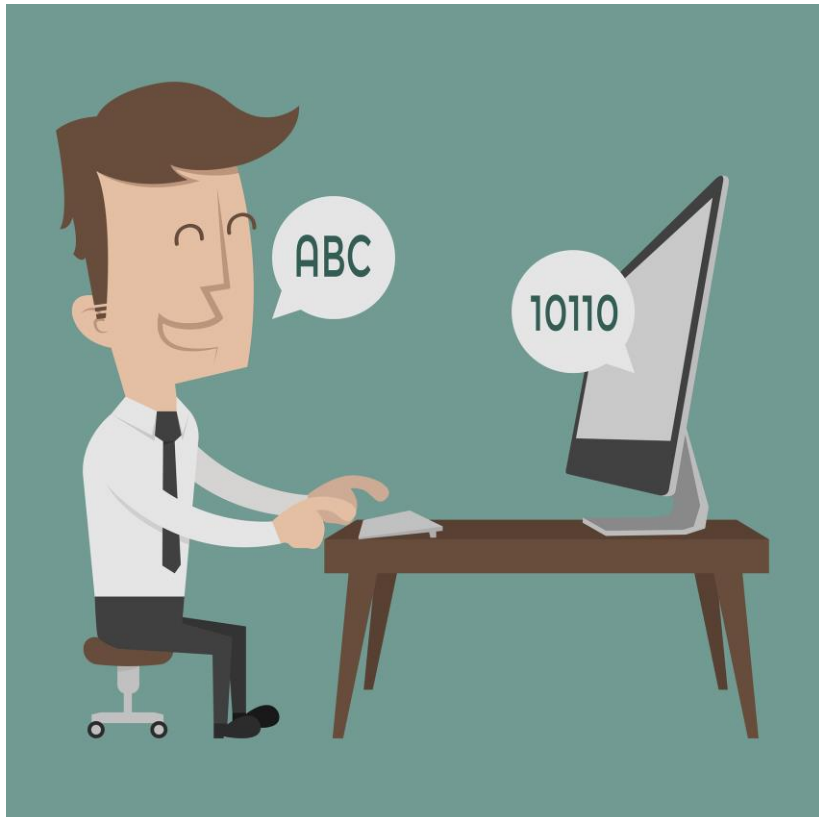
A few words on programming languages

- Akin to idioms from all over the world, programming languages follow their own grammar and synthax rules
- Different languages share some degree of compatibility depending on their relatedness
- Languages cannot be easily translated in one another



Programming languages are developed to communicate with computers

Computers and humans speak (and **think**) in very different ways

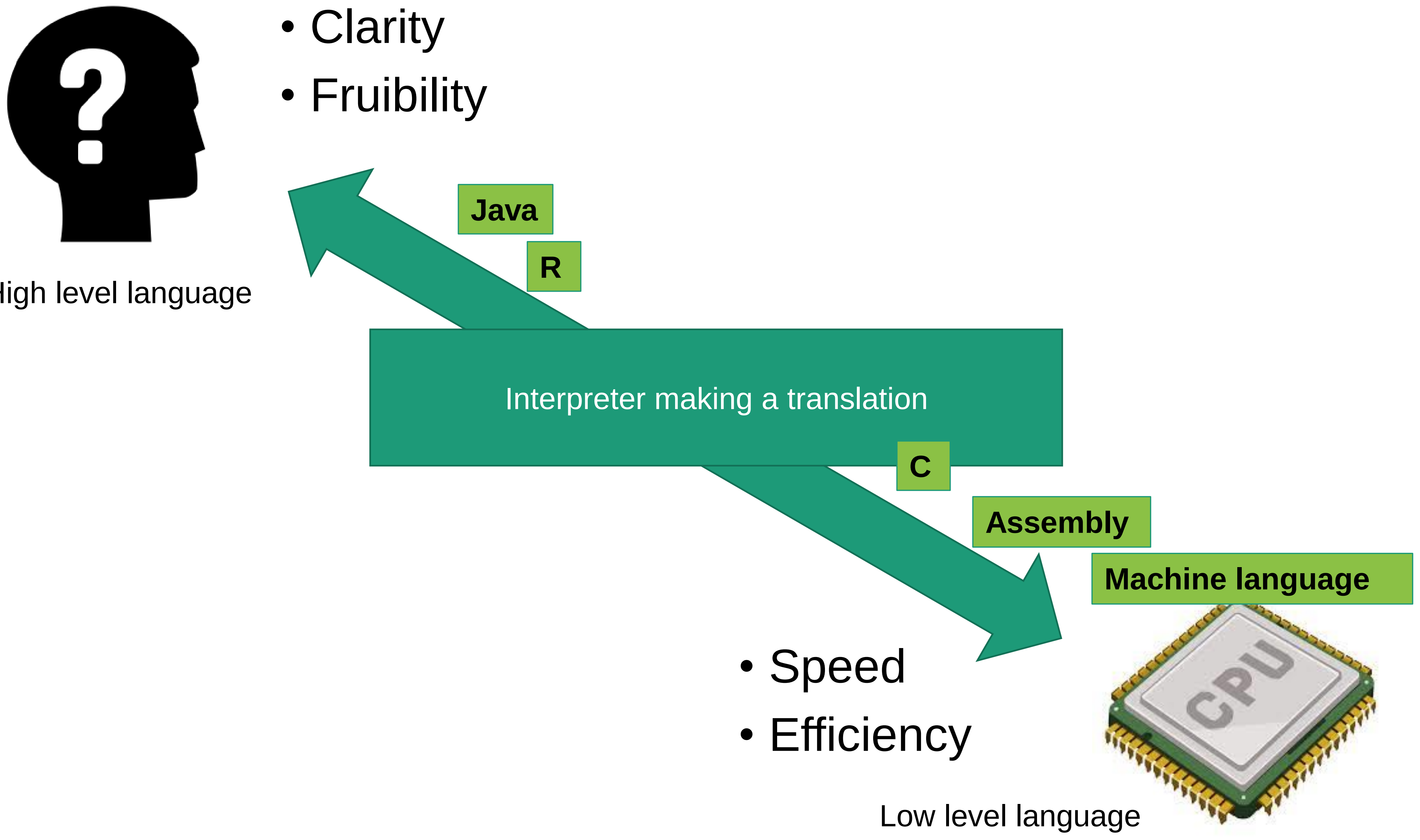


Programming languages can be sorted by their closeness to human language in:

- High level (close to human)
- Low level (close to machine)

Eventually, every instruction must be provided in 0s and 1s

- Clarity
- Fruibility

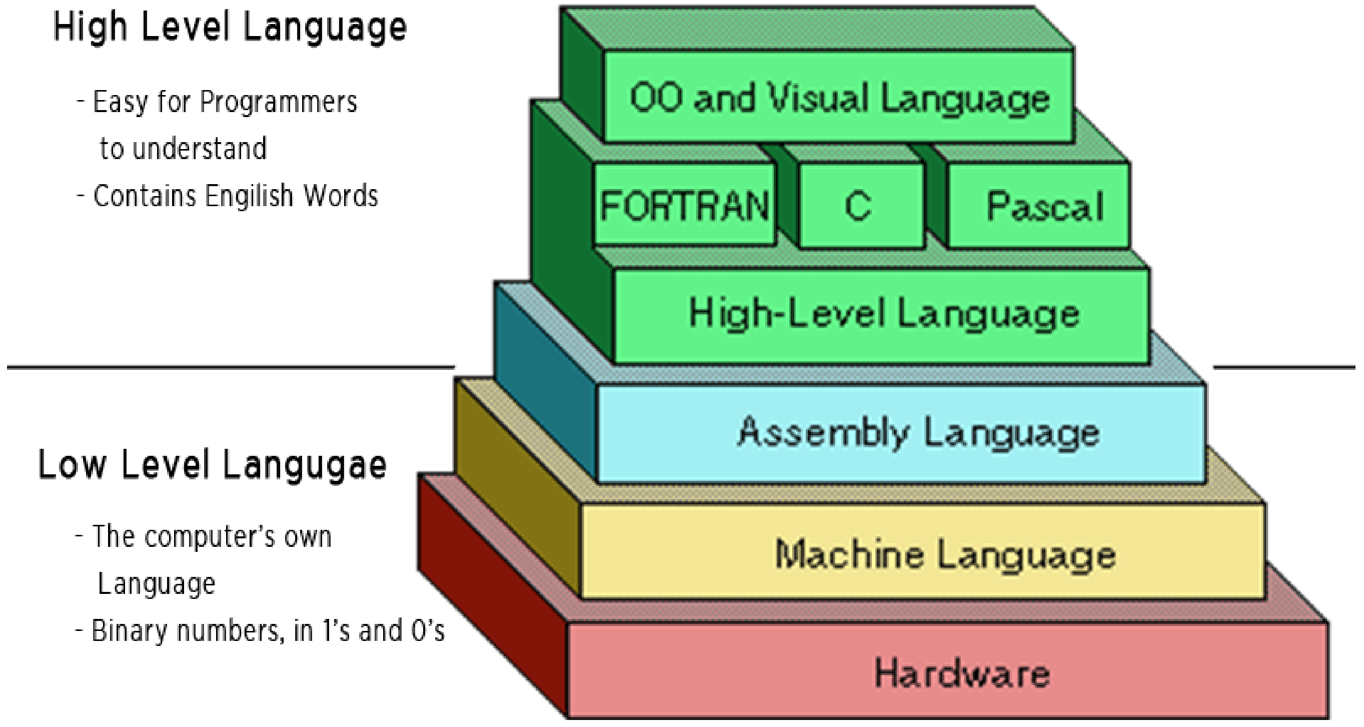


- Speed
- Efficiency

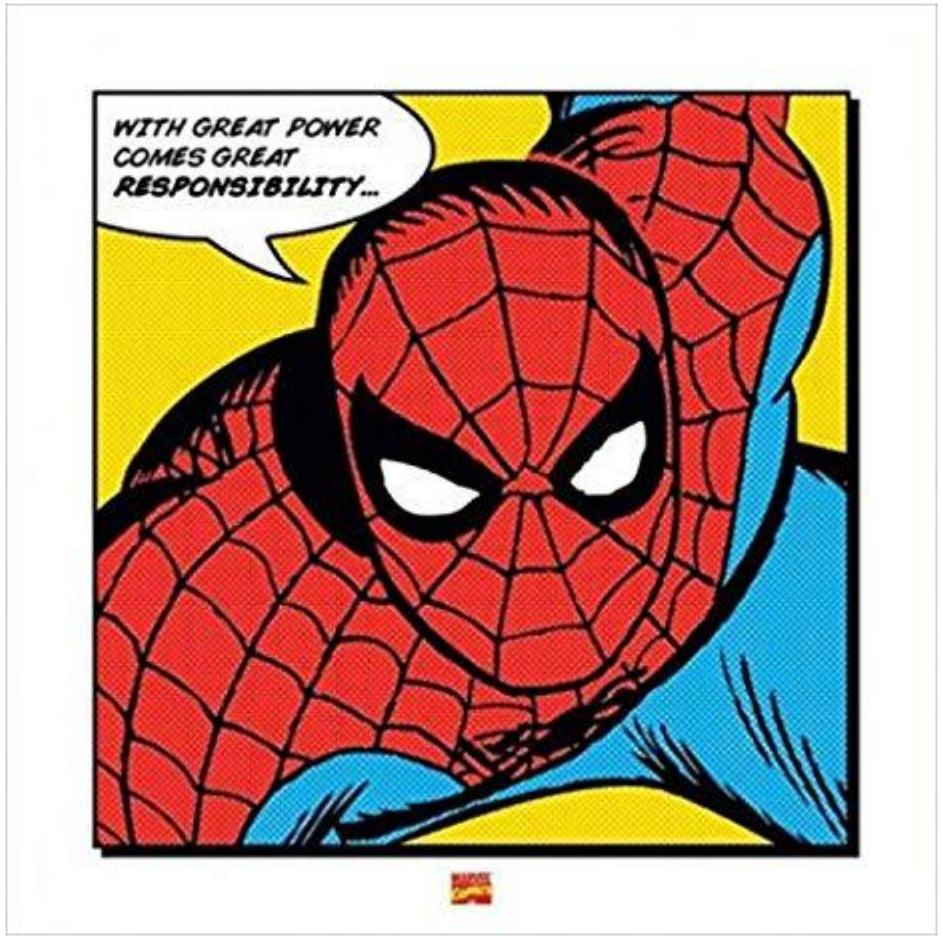
Computer languages come with interpreters

The three components are needed

$$\begin{tikzcd}
\text{Software (script)} \arrow[d] \\
\text{Interpreter} \arrow[d] \\
\text{Machine}
\end{tikzcd}$$

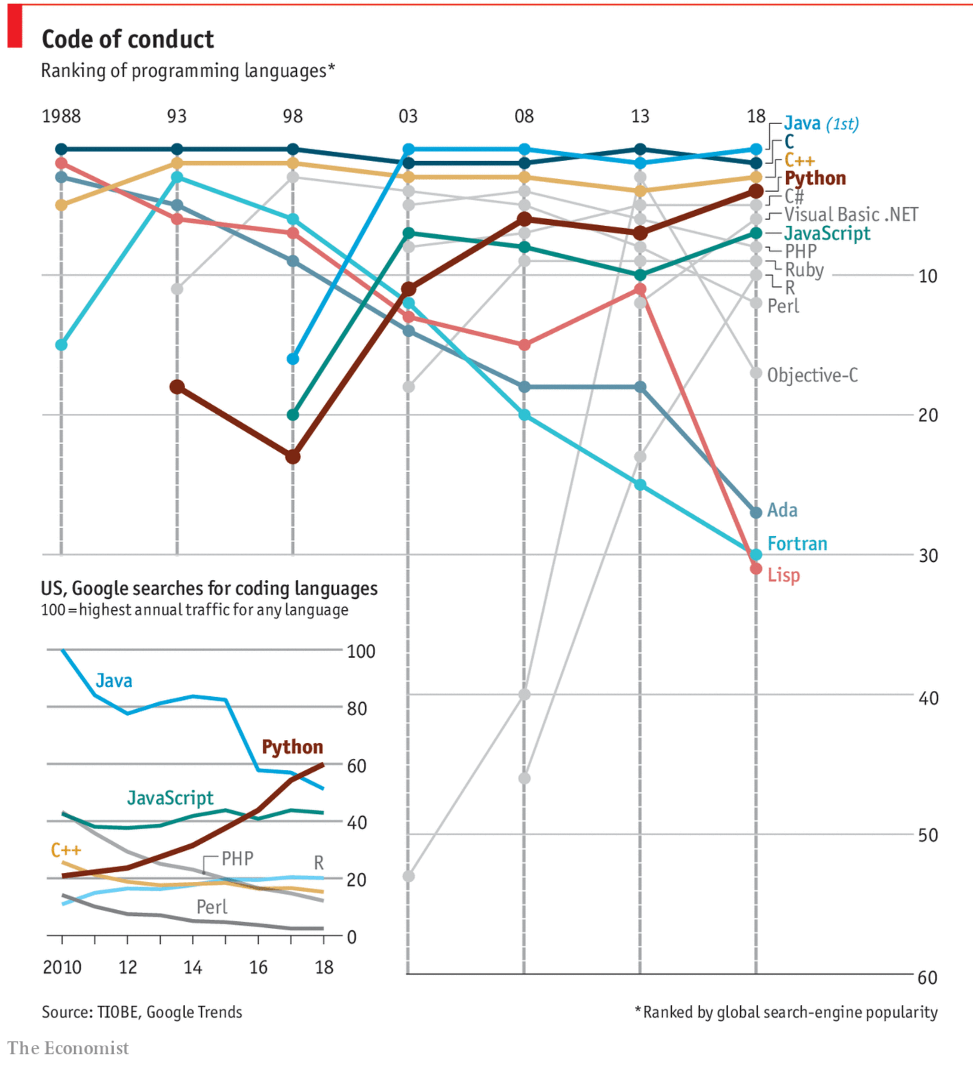


A friendly reminder



Why are we learning R?

- R is a programming language and an enrivornment for statistical computing
- It is a free software contributed by a large community
- It is one among many programming languages



- It is NOT the fastest
- It is NOT the most efficient
- It is NOT the easiest

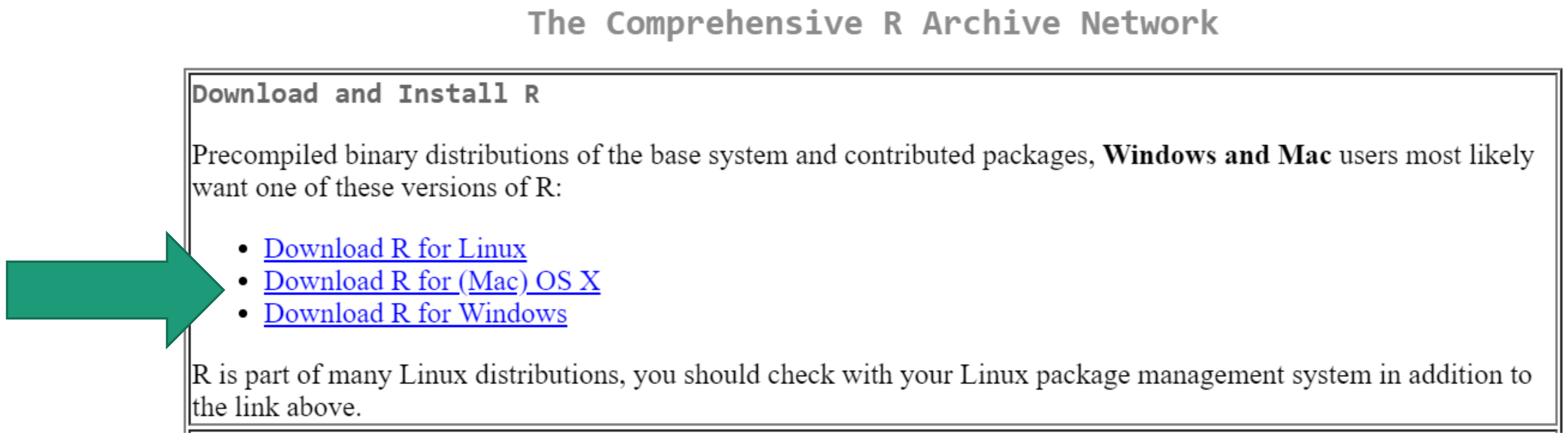
However, R is perhaps the language most used by the scientific community (at least in life sciences). For good reasons:

- It is great to conduct statistical analyses
- It is great to produce plots
- It is supported by a large community providing a number of tools aiding data analysis and supporting reproducibility

R installation

Go to R website and download the installation package compatible with your platform (Win, OS, Linux)

https://cran.r-project.org



The Comprehensive R Archive Network

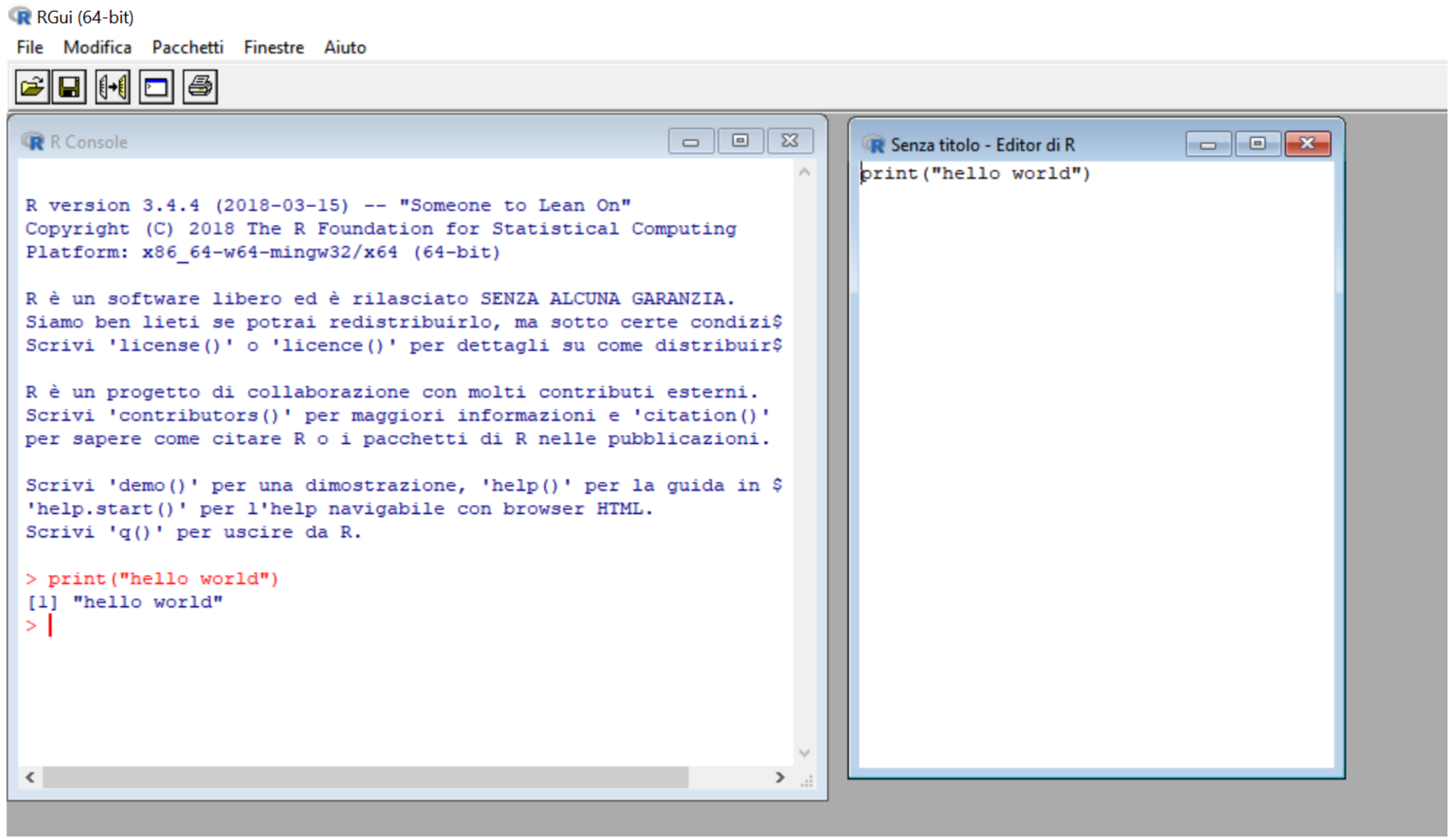
Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

R comes in a basic interpreter


RGui (64-bit)
File Modifica Pacchetti Finestre Aiuto
R Console
Senza titolo - Editor di R

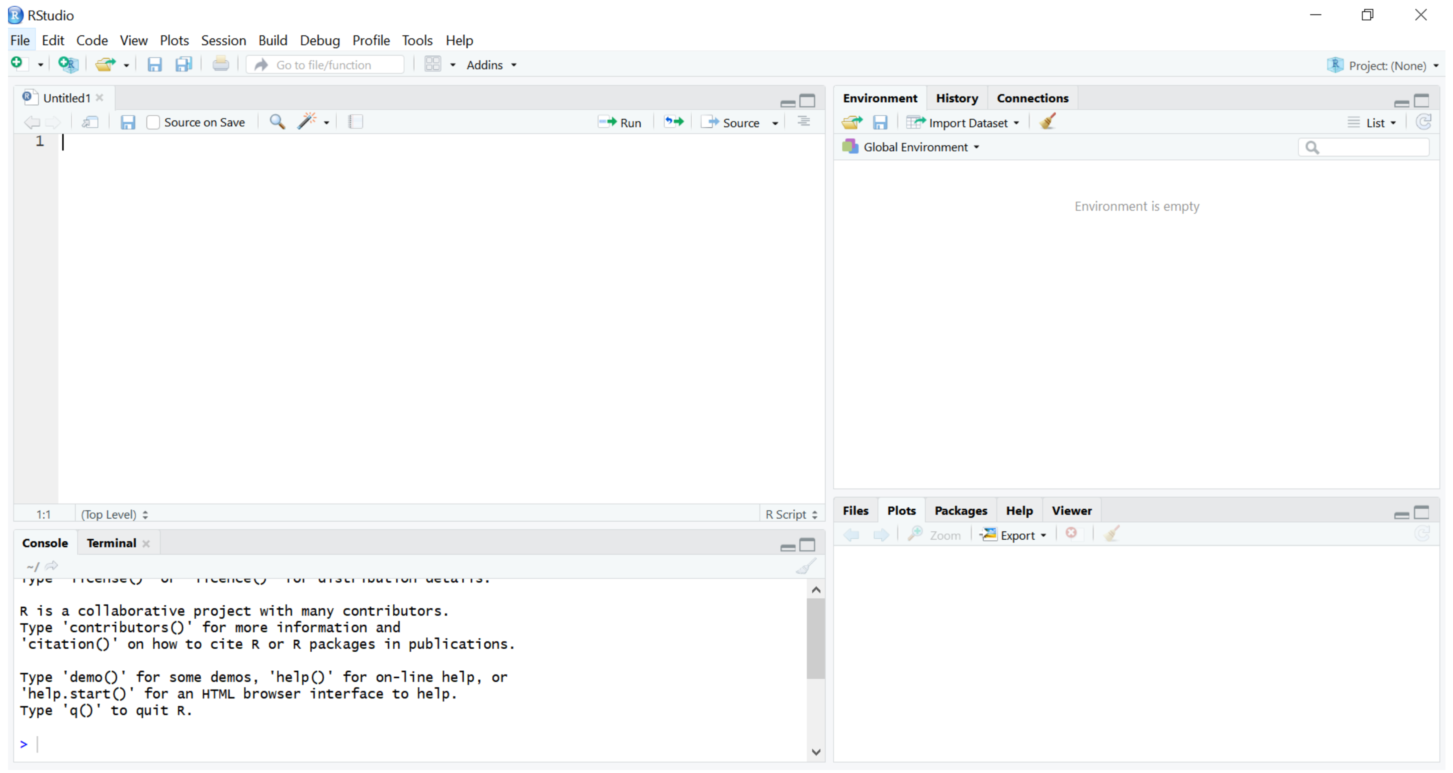
```
R version 3.4.4 (2018-03-15) -- "Someone to Lean On"
Copyright (C) 2018 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R è un software libero ed è rilasciato SENZA ALCUNA GARANZIA.
Siamo ben lieti se potrai redistribuirlo, ma sotto certe condizi$
Scrivi 'license()' o 'licence()' per dettagli su come distribuir$

R è un progetto di collaborazione con molti contributi esterni.
Scrivi 'contributors()' per maggiori informazioni e 'citation()'
per sapere come citare R o i pacchetti di R nelle pubblicazioni.

Scrivi 'demo()' per una dimostrazione, 'help()' per la guida in $
'help.start()' per l'help navigabile con browser HTML.
Scrivi 'q()' per uscire da R.

> print("hello world")
[1] "hello world"
> 
```

```
print("hello world")
```

- Not really user friendly
- Not really fancy
- Quite flexible
- Cross-platform
- **Good for learning**

I suggest installing also an advanced text editor such as **Notepad++** (Win) or **SublimeText** (OS)

For the sakes of this course, we will use Rstudio, as it provides additional functionalities

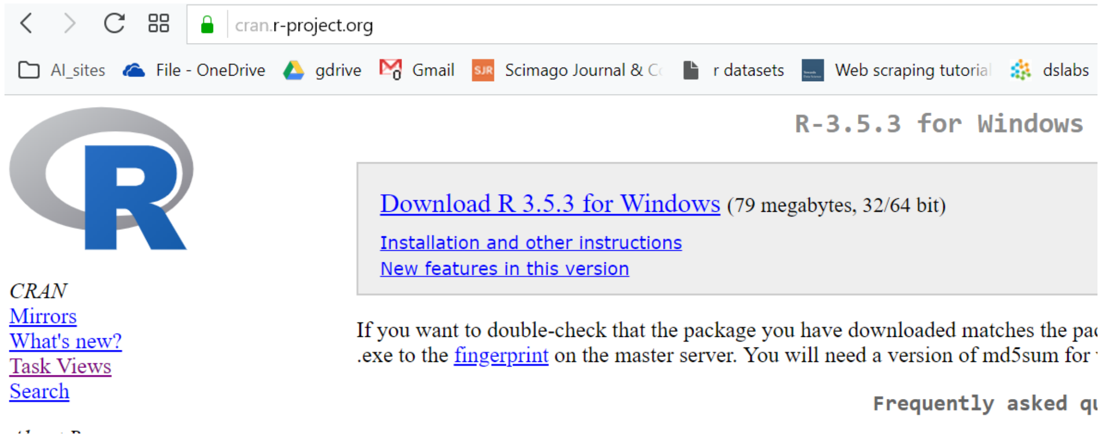


```
R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

>
```

Note: Rstudio requires the installation of the R interpreter to work

Move on to install Rstudio and an advanced text editor

https://cran.r-project.org



cran.r-project.org

AI_sites File - OneDrive gdrive Gmail Scimago Journal & C r datasets Web scraping tutoria dslabs

CRAN
Mirrors
What's new?
Task Views
Search

R-3.5.3 for Windows

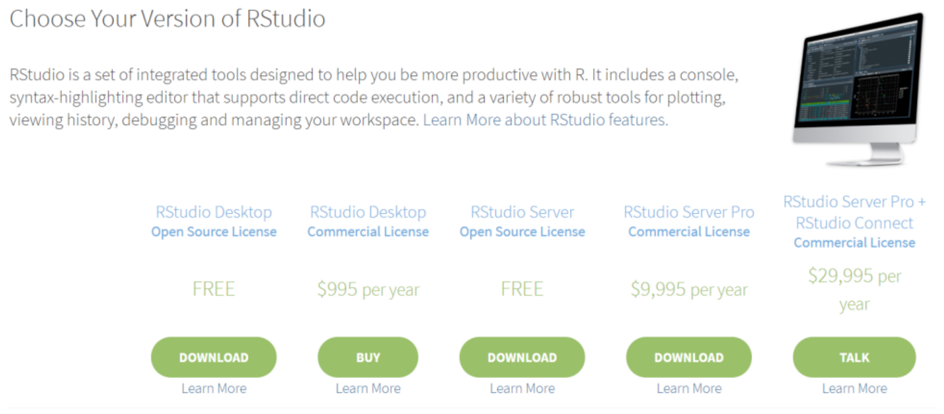
Download R 3.5.3 for Windows (79 megabytes, 32/64 bit)

Installation and other instructions
New features in this version

If you want to double-check that the package you have downloaded matches the pa
.exe to the fingerprint on the master server. You will need a version of md5sum for

Frequently asked qu

https://www.rstudio.com

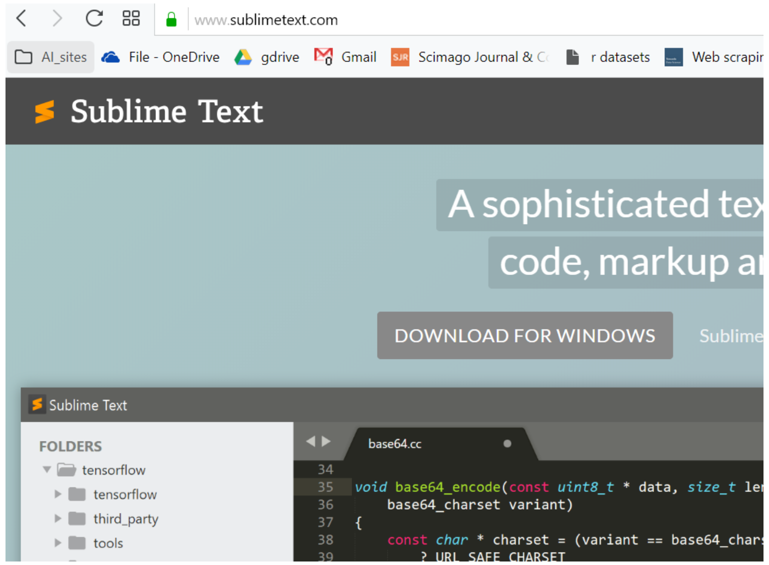


Choose Your Version of RStudio

RStudio is a set of integrated tools designed to help you be more productive with R. It includes a console, syntax-highlighting editor that supports direct code execution, and a variety of robust tools for plotting, viewing history, debugging and managing your workspace. Learn More about RStudio features.

RStudio Desktop Open Source License	RStudio Desktop Commercial License	RStudio Server Open Source License	RStudio Server Pro Commercial License	RStudio Server Pro + RStudio Connect Commercial License
FREE	$995 per year	FREE	$9,995 per year	$29,995 per year
DOWNLOAD	BUY	DOWNLOAD	DOWNLOAD	TALK
Learn More	Learn More	Learn More	Learn More	Learn More

https://www.sublimetext.org



www.sublimetext.com

AI_sites File - OneDrive gdrive Gmail Scimago Journal & C r datasets Web scrapir

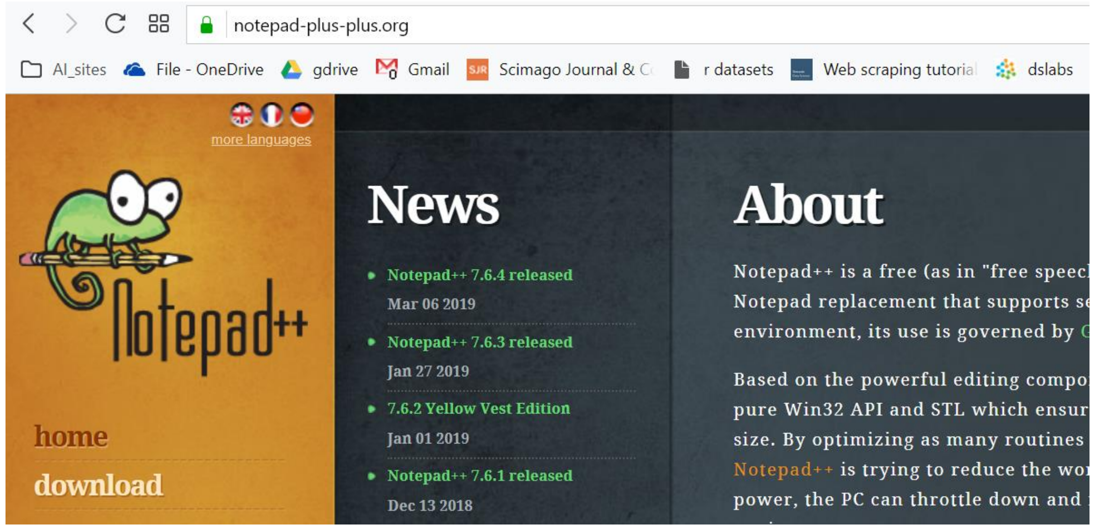
Sublime Text

A sophisticated tex
code, markup ar

DOWNLOAD FOR WINDOWS Sublime

```
void base64_encode(const uint8_t * data, size_t le
    base64_charset variant)
{
    const char * charset = (variant == base64_char
        ? URL_SAFE_CHARSET
```

https://www.notepad-plus-plus.org



notepad-plus-plus.org

AI_sites File - OneDrive gdrive Gmail Scimago Journal & C r datasets Web scraping tutoria dslabs

News

- Notepad++ 7.6.4 released
  Mar 06 2019
- Notepad++ 7.6.3 released
  Jan 27 2019
- 7.6.2 Yellow Vest Edition
  Jan 01 2019
- Notepad++ 7.6.1 released
  Dec 13 2018

About

Notepad++ is a free (as in "free speec
Notepad replacement that supports s
environment, its use is governed by

Based on the powerful editing compo
pure Win32 API and STL which ensur
size. By optimizing as many routines
Notepad++ is trying to reduce the wo
power, the PC can throttle down and

Let’s get to coding

When working in an R console, “>” is the command prompt; you will find it at the beginning of each line of code, provided that the console is ready to accept commands

The hashtag character “#” is used to denote comments; any line of text after it is not considered code

Each line of code is stand-alone, provided that special characters are not used to extend it in subsequent lines

Lower case letters and uppercase letter are considered different: Hello is not hello

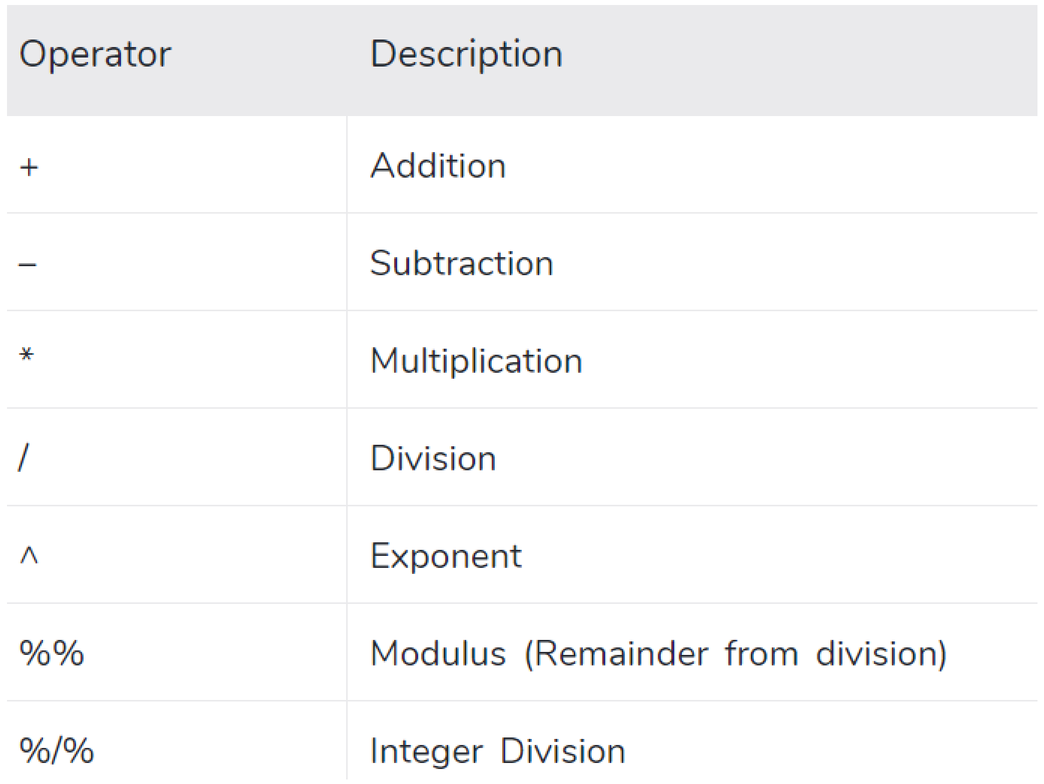
You may use how many white spaces as you want in between operators

R is a powerful calculator

```
> 2*3
[1] 6

> 7*8
[1] 56

> 5+7*8
[1] 61

> (5+7)*8
[1] 96
```

```
> 4^2
[1] 16

> 100%%10
[1] 0

> 100%%11
[1] 1
```

Arithmetic operators

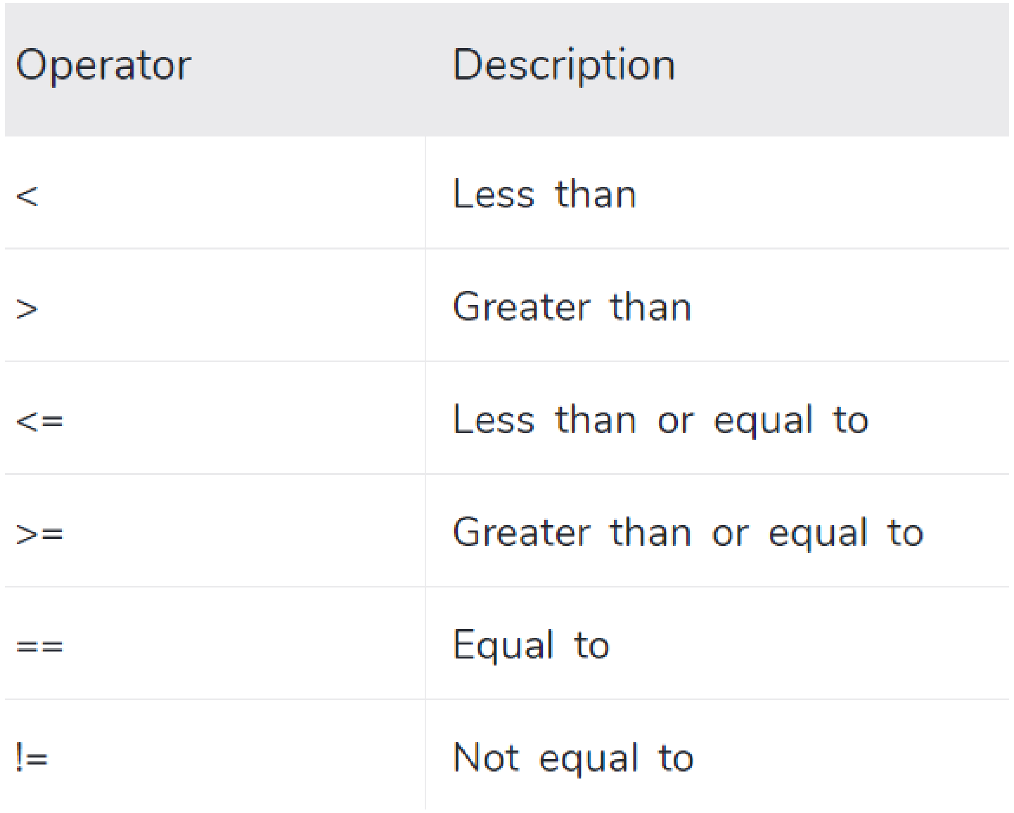


Operator	Description
+	Addition
–	Subtraction
*	Multiplication
/	Division
^	Exponent
%%	Modulus (Remainder from division)
%/%	Integer Division

Be ALWAYS aware of operator precedence

```
> 3 > 2
[1] TRUE

> 3 != 3
[1] FALSE

> 3*2 == 2*3
[1] TRUE

> 3^2 <= 2^3
[1] FALSE
```

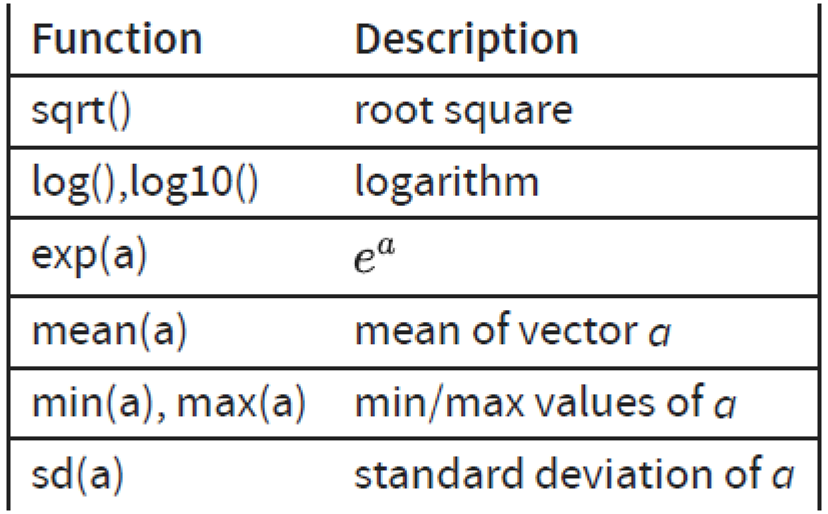
Relational operators



Operator	Description
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
==	Equal to
!=	Not equal to

Functions

- The cool thing is that you can assemble basic operators in what are called functions
- This helps in wrapping simple procedures in more complex tools, that can be nested one within the other (we will see more about that)
- Functions are identified by the round bracket operators



Function	Description
sqrt()	root square
log(),log10()	logarithm
exp(a)	e^a
mean(a)	mean of vector a
min(a), max(a)	min/max values of a
sd(a)	standard deviation of a

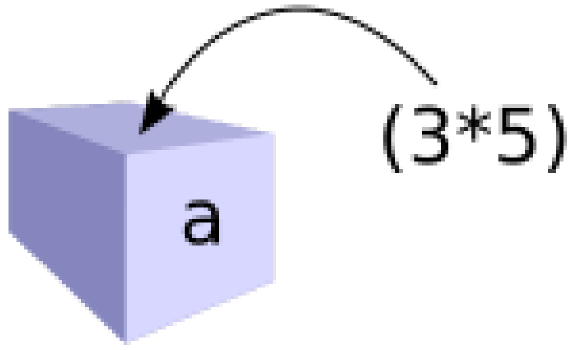
function.name(*argument1*, *argument2*, …)

R is object-oriented programming (OOP)

Values are assigned to objects through assignment operators

```
> 3*5
[1] 15

> a <- 3*5

> a
[1] 15
```

Operator	Description
<-, <<-, =	Leftwards assignment
->, ->>	Rightwards assignment



Operators and functions may then be used on objects

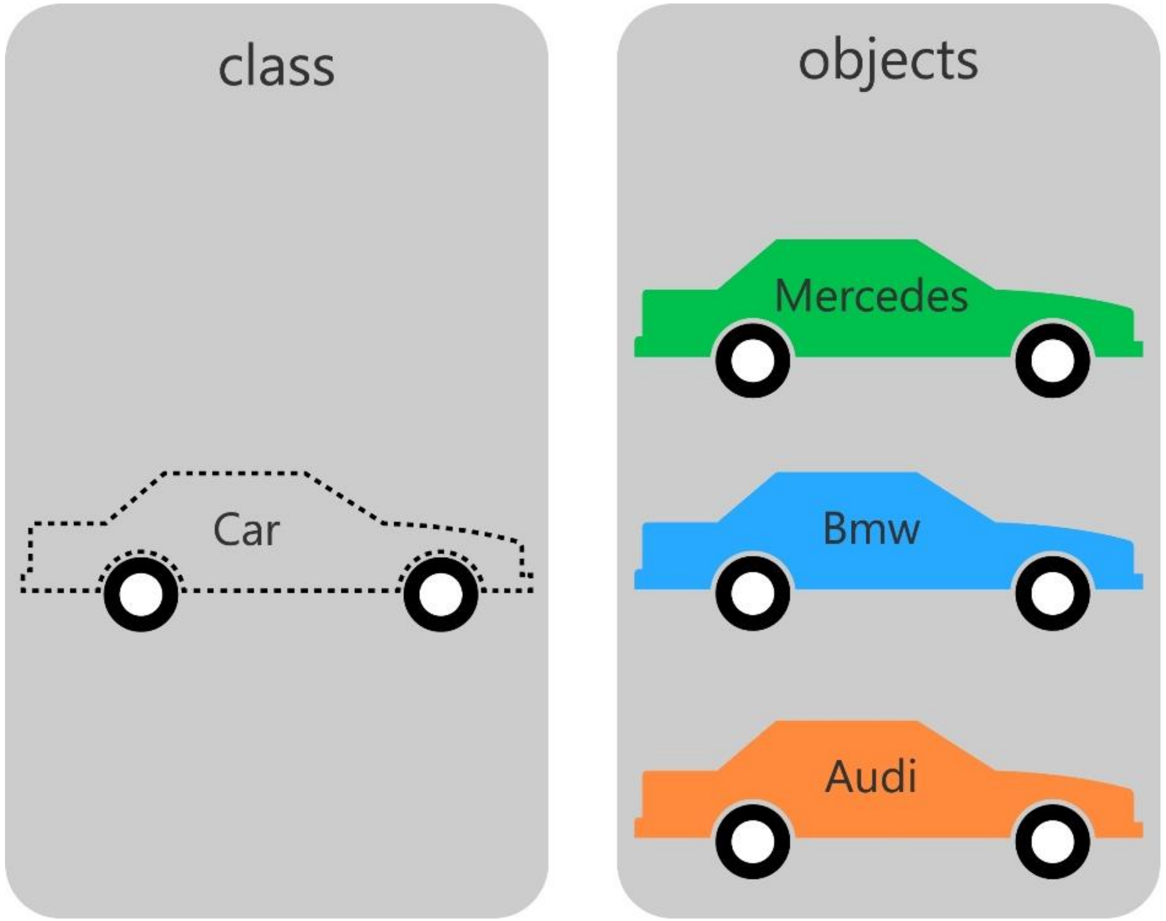
```
> a <- 3+3

> b <- 3*3

> a==b
[1] FALSE

> c<- a*b

> c < b
[1] FALSE
```

As a matter of fact, we will use objects all the time

Object names are **case-sensitive**:

“Result” is different from “result”

There are a few reserved words that is better not to use to name objects, especially those used by functions (e.g. “mean”)



- Depending on their features, objects may be traced back to classes

- Classes are characterized by specific attributes

- Classes define the data format (and features) of objects

R objects may belong to a number of different classes

«integer»
«logical»
«character»
«numeric»
«factor»
[…]

Classes dictate what you can do on a specific object; you cannot multiply characters, you cannot search for strings in logicals, etc.

Watch out for the representation of values

Integer	Floating number	Character
3	**3.0**	**"3"**

Coercion of data types

- Data may be converted from one type to the other with the appropriate functions: as.numeric(), as.character(), …

```
> a<-"3"
> anum<as.numeric(a)
> anum
[1] 3
```

- Be EXTRA careful and double check that everything is as intended

```
> is.numeric(anum)
[1] TRUE
```

Data structures

Data loaded in R may assume different structures, dipending on their dimensionality and features

- Vectors
- Matrices
- Dataframes
- Lists
- Arrays

Atomic vectors

- Ordered collections of objects of the same type
- Represent one-dimensional data
- May be created using the c() function that concatenates all its arguments

```
> vector <- c(1,4,7,2)
> vector
[1] 1 4 7 2
> class(vector)
[1] "numeric"
```

```
> vector <- c(1,4,7,"2")
> vector
[1] "1" "4" "7" "2"
> class(vector)
[1] "character"
```

Be careful with the dynamic assignment of object classes

Some functions for vectors

length()	number of elements in vector
sum()	sum of all elements of vector
cumsum()	cumulative sum
prod()	product of all elements of vector
order()	order vector elements
head()	print the beginning of the vector
unique()	extract unique values

Try it

Make a vector named "vec" with the following elements:

21, 23, 24, 21, 55, 32

- Calculate length, maximum, product, sum, cumulative sum
- Get unique values in a new vector named "u.vec"

Matrices

- Rectangular set of elements with n rows and m columns
- It contains data of the same type
- Can be initialized with the matrix() function

```
> mat <- matrix("value", nrow=4, ncol=5, byrow=T)
> mat
     [,1]    [,2]    [,3]    [,4]    [,5]
[1,] "value" "value" "value" "value" "value"
[2,] "value" "value" "value" "value" "value"
[3,] "value" "value" "value" "value" "value"
[4,] "value" "value" "value" "value" "value"
```

Some functions for matrices

nrow() / ncol()	number of rows/columns in matrix
colSums() / rowSums	sum by columns/rows
dim()	give dimensions of matrix
rbind() / cbind()	adds rows or columns to matrix
head()	print the first few rows of the matrix
t()	transpose matric
colnames() / rownames ()	get (and set) names of columns and rows

Try it

Make a matrix "mat" with values from 1 to 25 sorted in 5 columns, 5 rows

- Get dimensions, calculate sums by columns
- Create a new object "tmat" transposing mat and calculate sums by rows
- Join "tmat" ad "mat" by rows in a new "mat2" object, then compute again sums by columns

Data frames

- Similar to matrices, but able to contain different classes of data at once
- You may initialize them with the data.frame() function

```
> vector1<-c("A", "A", "B", "B")
> vector2<-c(32,43,67,65)
> df<-data.frame(type=vector1, count=vector2)
> df
  type count
1    A    32
2    A    43
3    B    67
4    B    65
```

Try it

Make a df joining the vector "u.vec" (see above) and a vector with the elements "cat", "cat", "dog", "cat", "dog"

- Get number of rows, columns
- Create an object "bigdf" joining "df" and "mat" by rows
- fix row names using a vector with elements "r1", "r2", "r3", "r4", "r5"
- fix column names using numbers from 1 to 7

Lists

- Lists are ordered sets of elements of different classes
- They can contain anything, including other lists
- You may initilize lists with the list() function

```
> ls<-list("A", c(1,2,5), list("nested", "list"))
```

```
> ls
[[1]]
[1] "A"

[[2]]
[1] 1 2 5

[[3]]
[[3]][[1]]
[1]
"nested"

[[3]][[2]]
[1] "list"
```

Try it

Make a list "weirdlist" joining mat, mat2, bigdf, and vec

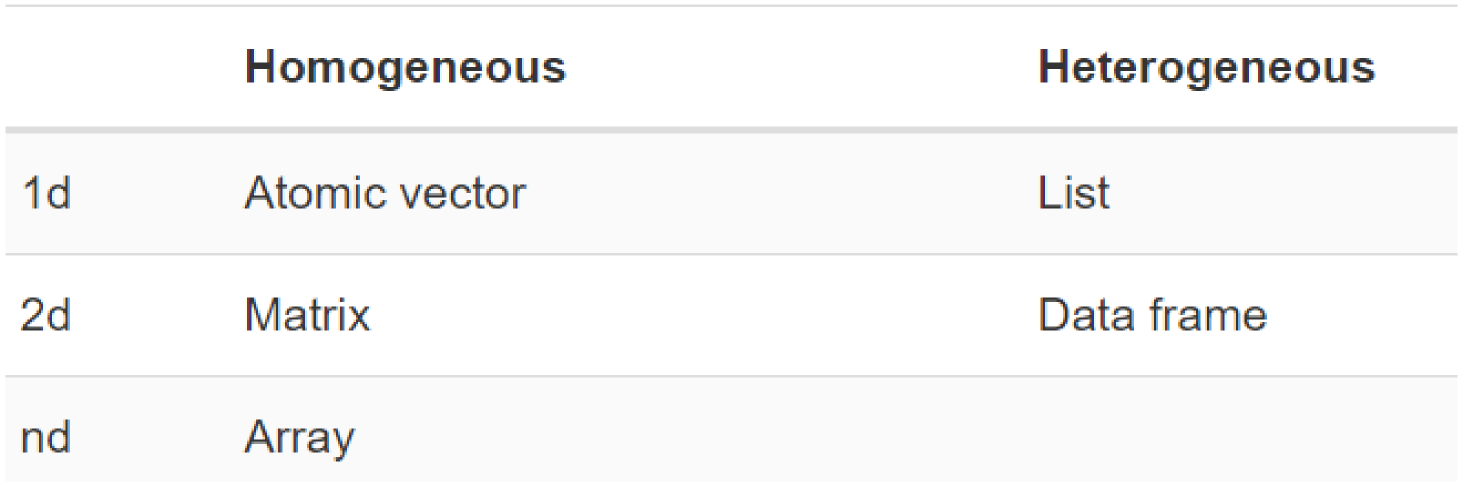
- Get length of list
- Print head of list elements and inspect them

Arrays

- Objects capable to store data in any number of dimensions
- Data stored in arrays must be homogenous
- Arrays might be initialized with the function array()

```
>vector1 <- c(5,9,3)
>vector2 <- c(10,11,12,13,14,15)
>arr <- array(c(vector1,vector2),dim = c(3,3,2))
>arr
, , 1

     [,1] [,2] [,3]
[1,]    5   10   13
[2,]    9   11   14
[3,]    3   12   15

, , 2

     [,1] [,2] [,3]
[1,]    5   10   13
[2,]    9   11   14
[3,]    3   12   15
```

In short…



	Homogeneous	Heterogeneous
1d	Atomic vector	List
2d	Matrix	Data frame
nd	Array	

Structure of objects at a glance

If you feel lost, you may know the features of your objects with a couple of very useful functions

- class(), to give you the class of your object
- str(), to give you the structure of your object

```
> df
  type count
1    A    32
2    A    43
3    B    67
4    B    65
```

```
> str(df)
'data.frame':   4 obs. of  2 variables:
 $ type : Factor w/ 2 levels "A","B": 1 1 2 2
 $ count: num  32 43 67 65
```

Data indexing

- To access individual elements of data structures, we may use the square brackets operators: [ ]
- Depending on dimensionality, brackets may be used to slice different components of data using the comma separator «,», the colon operator «:», the dollar symbol «$», the dash symbol «-»

1. Numeric indexing
2. Name indexing
3. Logical indexing

Indexing vectors

vector[index**]**

```
> vec<-c(2,5,7,8,9,12,14)
```

```
> vec[5]
[1] 9
```

Single element

```
>vec[2:5]
[1] 5 7 8 9
```

Range

```
>vec[c(2,5)]
[1] 5 9
> vec[-c(2,5)]
[1]  2  7  8 12 14
```

Specific elements

```
> vec[vec>5]
[1]  7  8  9 12 14
```

Logical statement

Indexing Matrices/Data Frames

dataframe[row index**,** col index**]**

```
> df
  var count
1   A    50
2   B    44
3   C    53
```

```
> df[2,]
  var count
2   B    44
```

By row

```
> df[,2]
[1] 50 44 53
```

By column

```
> df$var
[1] A B C
Levels: A B C
```

By name

```
> df[3,2]
[1] 53
```

Specific element

```
> df[df[,2]>=50,]
  var count
1   A    50
3   C    53
```

Nested logical

```
>ls<-list("A", df, c(1:6))
> ls
[[1]]
[1] "A"

[[2]]
  var count
1   A    50
2   B    44
3   C    53

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

Indexing Lists

list[[index**]]**

```
> ls[[1]]
[1] "A"
```

Single element

```
> ls[[2]][1:3,2]
[1] 50 44 53
```

Nested subset

Indexing Arrays

array[dimension 1, dimension 2, dimension n**]**

Indexing may also be used in negatives

```
> vec
[1]  2  5  7  8  9 12 14

> vec[-c(1:3)]
[1]  8  9 12 14
```

```
> df
  var count
1   A    50
2   B    44
3   C    53

> df[-1,]
  var count
2   B    44
3   C    53
```

All of the above is VERY useful to create new objects and manipulate data

Handling missing data

- Missing data may be stated in any object with the special value **NA** (not available)
- **NA** values are treated differently from other values and may be independentely targeted

```
> x<-c(1,5,NA,3)
> is.na(x)
[1] FALSE FALSE  TRUE FALSE
```

- Other special values are **Inf** (infinity) and **NaN** (Not a Number)

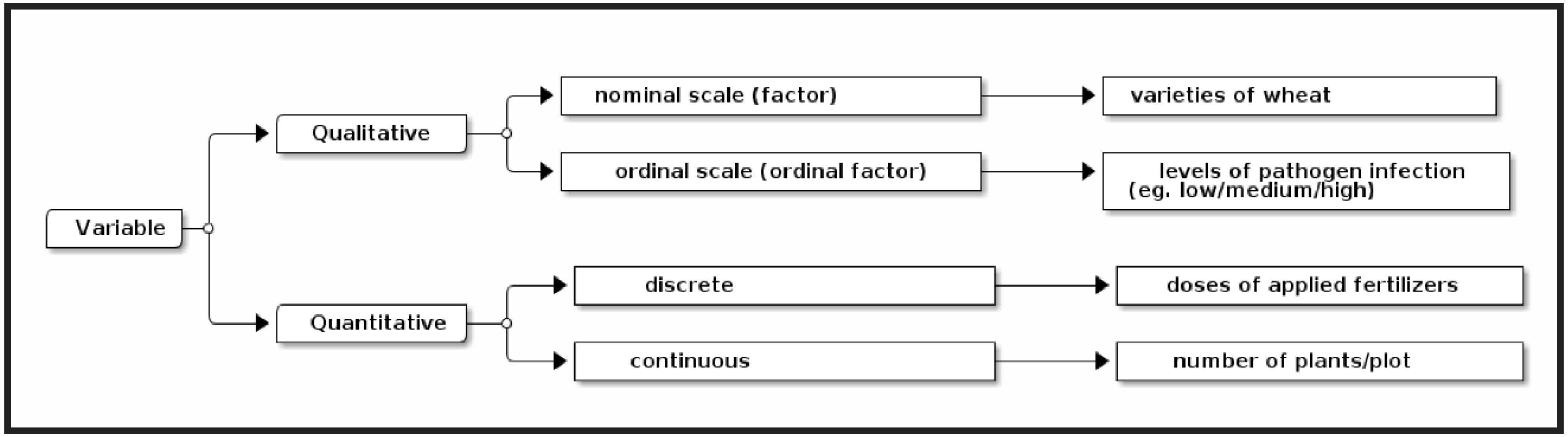
Be careful, as some functions and processes will need special instructions to deal with missing data!!

In whichever software you are coding your data, there should not be any other missing data value

- ?
- *
- Missing
- 0 Especially bad
- ...

These will affect the class of the data imported by R, most notably converting numbers in strings or factors

In statistics, variables come in many flavours



They are each best represented by an object class

The factor class: handle with care

- Variables that take only a limited number of different values
- Good for representing categorical variables and conducting the appropriate statistics (e.g. ANOVA)
- Factors are stored as vectors with a corresponding set of values (factors) printed when the vector is printed

```
> vec<-c(1,2,3,1,2,3,3,3,3,2,1,2,4)
> vec
 [1] 1 2 3 1 2 3 3 3 3 2 1 2 4
> fvec<-factor(vec)
> fvec
 [1] 1 2 3 1 2 3 3 3 3 2 1 2 4
Levels: 1 2 3 4
```

Watch out, as levels are embedded in the object and will remain there also after filtering

```
> fvec
 [1] 1 2 3 1 2 3 3 3 3 2 1 2 4
Levels: 1 2 3 4
> fvec[13]<-2
> fvec
 [1] 1 2 3 1 2 3 3 3 3 2 1 2 2
Levels: 1 2 3 4
```

In any moment, you may know which are the levels included in a factor object with the fucntion *levels(object)*

If you are not planning to use factors, for good use it may be good to set *option(stringsAsFactors=F)* at the beginning of your script; otherwise, character strings may be interpreted as factor levels

Recap excercise

Load some flower information by typing *data(iris)*

Rookies:

1. What type is this data?
2. What class is data in column 1? Column 5?
3. How many flowers have a width greater than 1?
4. Create a new matrix with sepal lenght, sepal witdh, and species name

Pros:

1. What is the mean petal lenght in the versicolor species?

Recap excercise 2

Load data from an experiment testing the relation between vitamin C and tooth growth in guinea pigs --> *data(ToothGrowth)*

Rookies:

1. Get the structure of the dataset
2. Create two datasets based on unique values in column 2
3. Get dimensions of each
4. What is the mean tooth length in each dataset?
5. What is the mean tooth length in each dataset but only for doses of 2.0?

Pros:

1. Create as many datasets as there are supp, and doses in each; compute mean tooth length in each, and assemble them in a new dataframe with 2 rows (each for supp) and three columns (each for a dose)