Day 1: Introduction to Computing

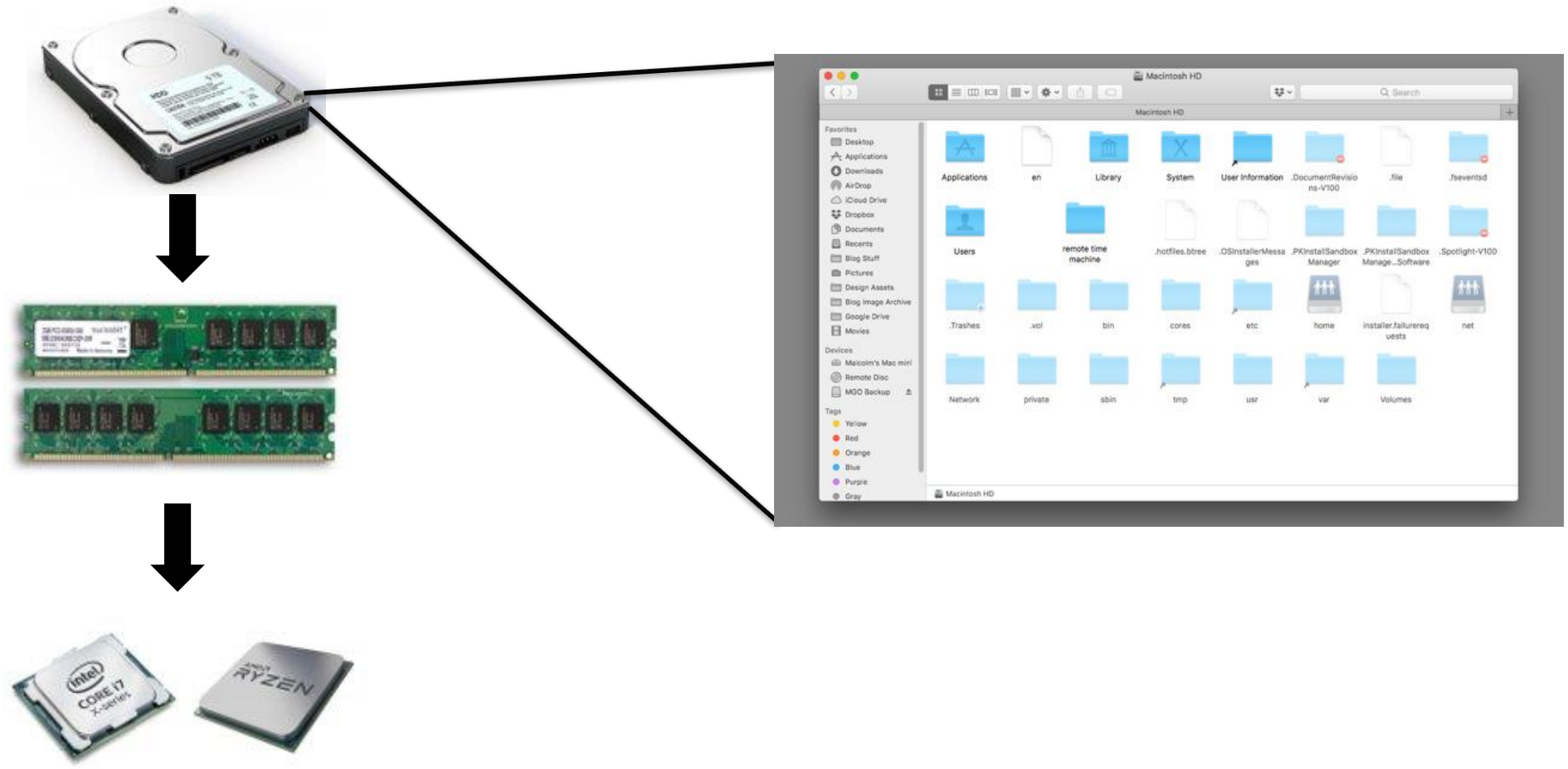
Why learn computing?

- Data collection
- Analysis
- Visualization
- Replication

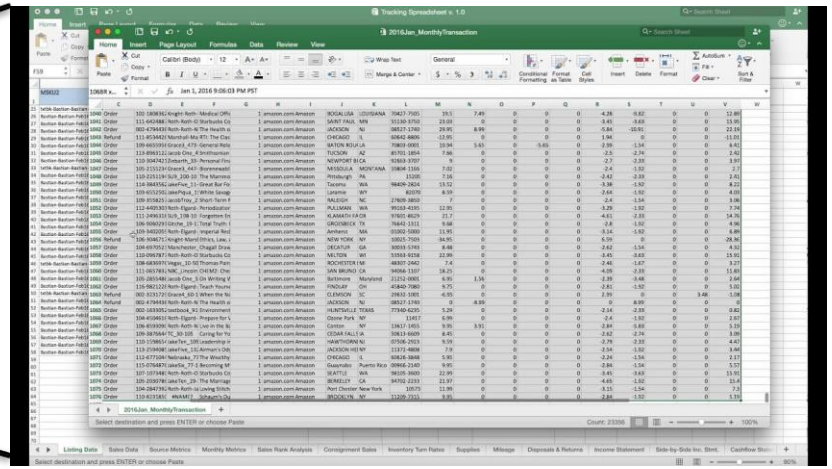
Computing is a force multiplier for scientists. You'll be able to do more, better, faster.

How your computer works

A familiar example…



A familiar example…



A familiar example...

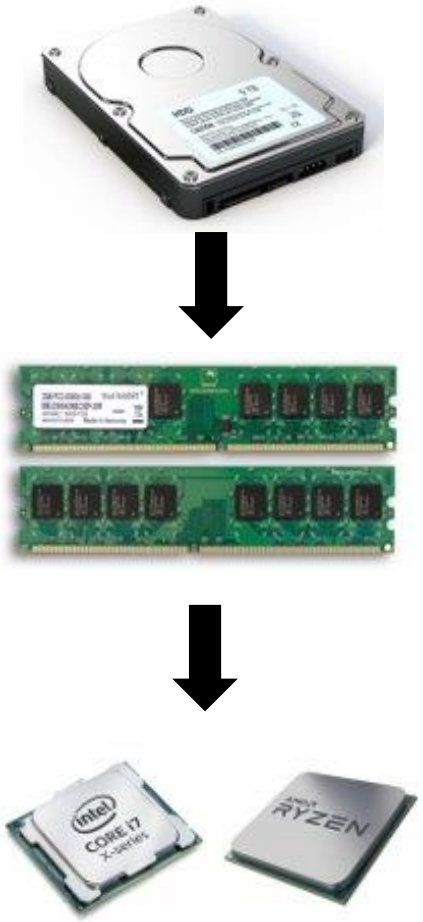
SEARCH =SUM(B2:B5)

	A	B	C	D
1	Name	Week 1	Week 2	Total
2	Bob	$7,894	$6,942	$14,836
3	Rishna	$4,897	$9,375	$14,272
4	Sue	$7,835	$2,845	$10,680
5	Mo	$9,584	$6,458	$16,042
6	Total	=SUM(B2:B5)		$55,830
7				

Intellisense function guide SUM(**number1**, [number2], ...)

This is most of what you will do with R. Instead of point and click, we write the process out in R for speed and reproducibility.

The three types of errors you will most frequently encounter revolve around this process



1. I can't find the data or package you told me to find
2. There's too much information
3. I don't understand what you want me to do

Example in R studio

- Run my_first_program.R section and demonstrate process in htop
- Give quick tour of R studio including scripts, notebooks, the console, environment

What is a programming language and how do they work?

Setting expectations

- Vocabulary, grammar, and syntax for instructing a computer to execute a program or algorithm.
  - It's not like learning a language
  - It's not like learning math
- You will never write R the same way you write your native language. Not because R is exceptionally difficult, but because programming is a fundamentally different task. Programming is about solving problems and automating work flows, natural languages are about expression.
- Expect to have questions constantly. You'll never stop seeking help or troubleshooting bugs.
- You will get much better at diagnosing and solving problems.
- You will get more aware of what tools are available to help you accomplish your goals.

Data types

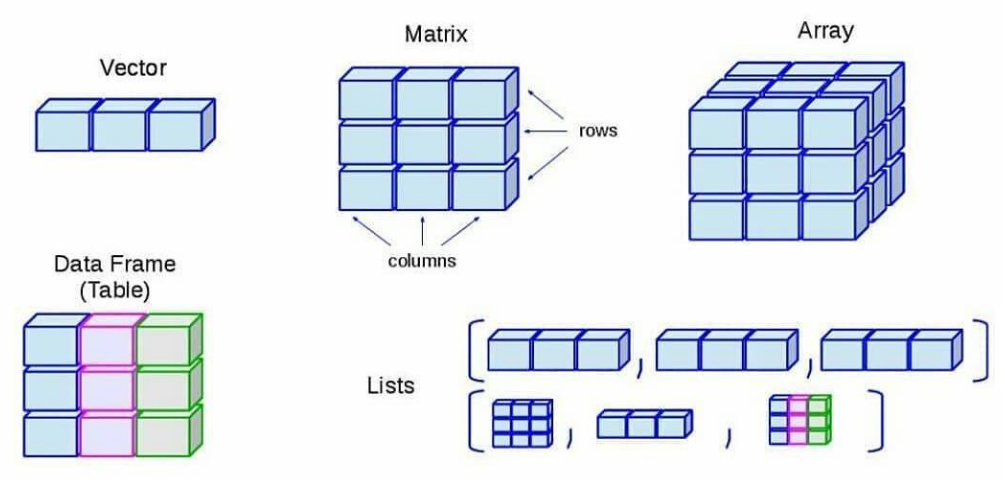
Why is this important?

- Different data types mean different things to the computer
- Some data types are more efficient (strings are slow)
- Numeric has precision limits (rarely a problem, but sometimes relevant e.g. twitter IDs)

Six basic types:

- Character (strings)
- Numeric (doubles, all real numbers. You will often hear them called floats as well)
- Integer
- Logical (True/False, Boolean)
- Complex (imaginary numbers)
- Raw (don't worry about this one)

Data Structures



- + Factors (categoricals)

<u>Important points:</u>

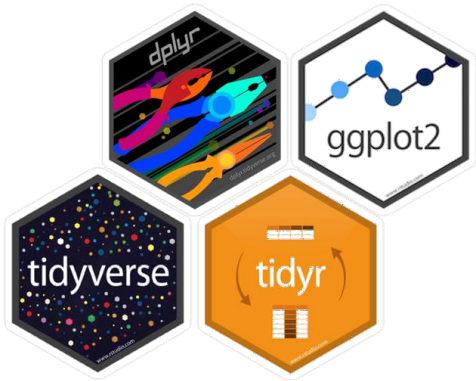
- The Dataframe will be your best friend. Think of it like an excel table
- Vectorized operations by default
- R will import strings as factors by default unless you specify otherwise

Data Manipulation

R is what's known as a functional programming language. All you need to know for now is that data is…

- Stored in **objects** and **variables**
- Manipulated with **operators** and **functions**

Functions accept **arguments.** New functions and operators are imported with **packages,** or can be created by the user.



General advice

- Avoid point and click, keep your hands on the keyboard as much as possible
- Read and think about error messages. It won't be obvious what they mean at first, but trying to understand them will help you learn R
- Comment on any line of code that isn't immediately obvious. This is for you as much as it is for others replicating your work.

Concepts in computing vocab

- RAM
- CPU
- Storage, hard-drive, SSD
- Float and double (numeric)
- Integer
- String (character)
- Bool (logical)
- Syntax error
- Semantic error
- Exception
- Variable
- Object
- Function
- Argument
- Object
- IDE
- GUI
- Script/notebook
- R Markdown