Business Analysis Summer 2022

Class 1

Sri Akhil Jonnalagadda
sjonnala@stedewards.edu

About Me

- Masters/Bachelors Economics for The University of Texas at Austin
- Research interests are in healthcare and energy economics.
- Industry focus is in startups, data science and health economics.
- Big soccer guy, loves to cook and recently into endurance sports.
- Feel free to ask about my work as a Data Scientist, Economist, Consulting and Product Management

Student Learning Outcomes for the Semester

1. Understand the basics of computer hardware & software (I/O, memory, storage, logic).
2. Use python shell and learn basic commands and data structures.
3. Write and run python programs that take inputs and produce outputs.
4. Understand loops and functions in Python.
5. Learn how to ingest csv data into Python structures and export data into csv.
6. Learn how to manipulate data in Python and produce output.
7. Learn how to install and use Python advanced libraries.
8. Learn how to summarize data with statistics tools including Pandas.
9. Learn how to visualize data with Pandas and MatPlotLib.
10. Learn problem-solving techniques using data analysis.

Course Grading

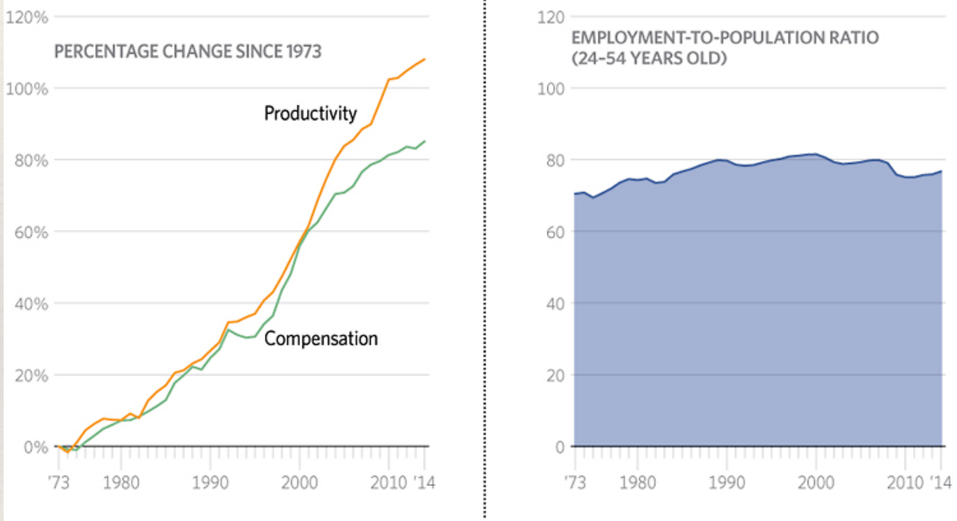
Graded Item	% of Final Grade
Discussions	10
Live Sessions	10
Quiz 1	15
Assignments (week 1-6)	40
Final Assignment (week 7)	25

What are computers so special?
What's the big deal?

CHART 1

Labor Productivity Aided by Technological Advances

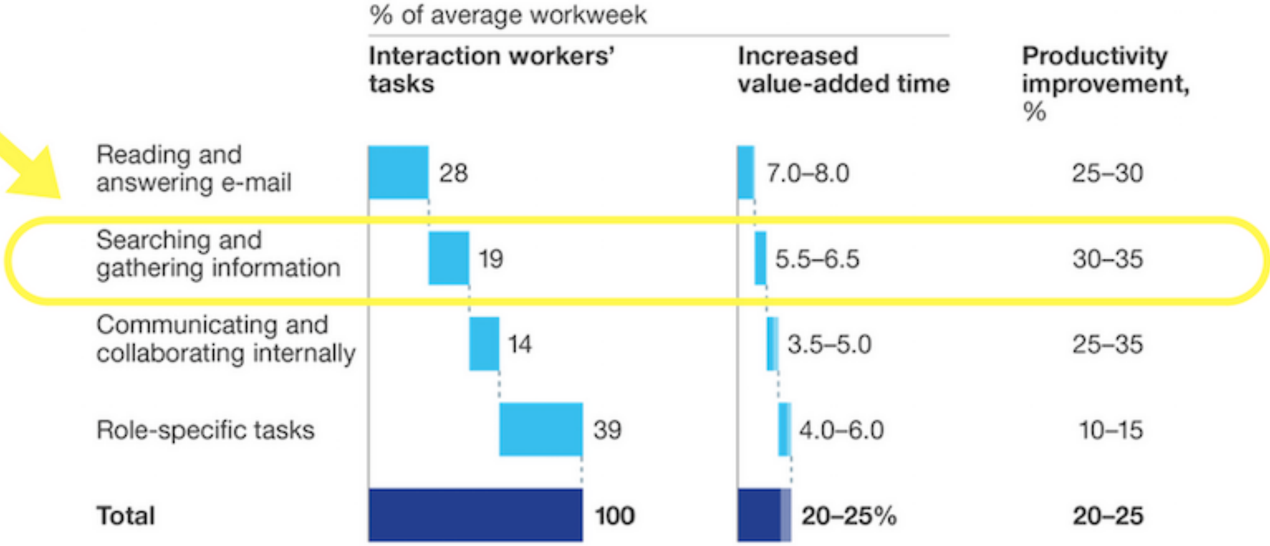
Between 1973 and 2014, technological advances have made workers more productive and sharply increased their pay. However, it has had little effect on overall employment rates.



Note: Productivity and compensation are adjusted for inflation using the implicit price deflator for non-farm businesses.
Source: U.S. Department of Labor, Bureau of Labor Statistics, and Haver Analytics, "Labor Force Statistics from the Current Population Survey," Employment to Population Ratio (25–54 years old) and "Productivity and Costs, "non-farm business sector.

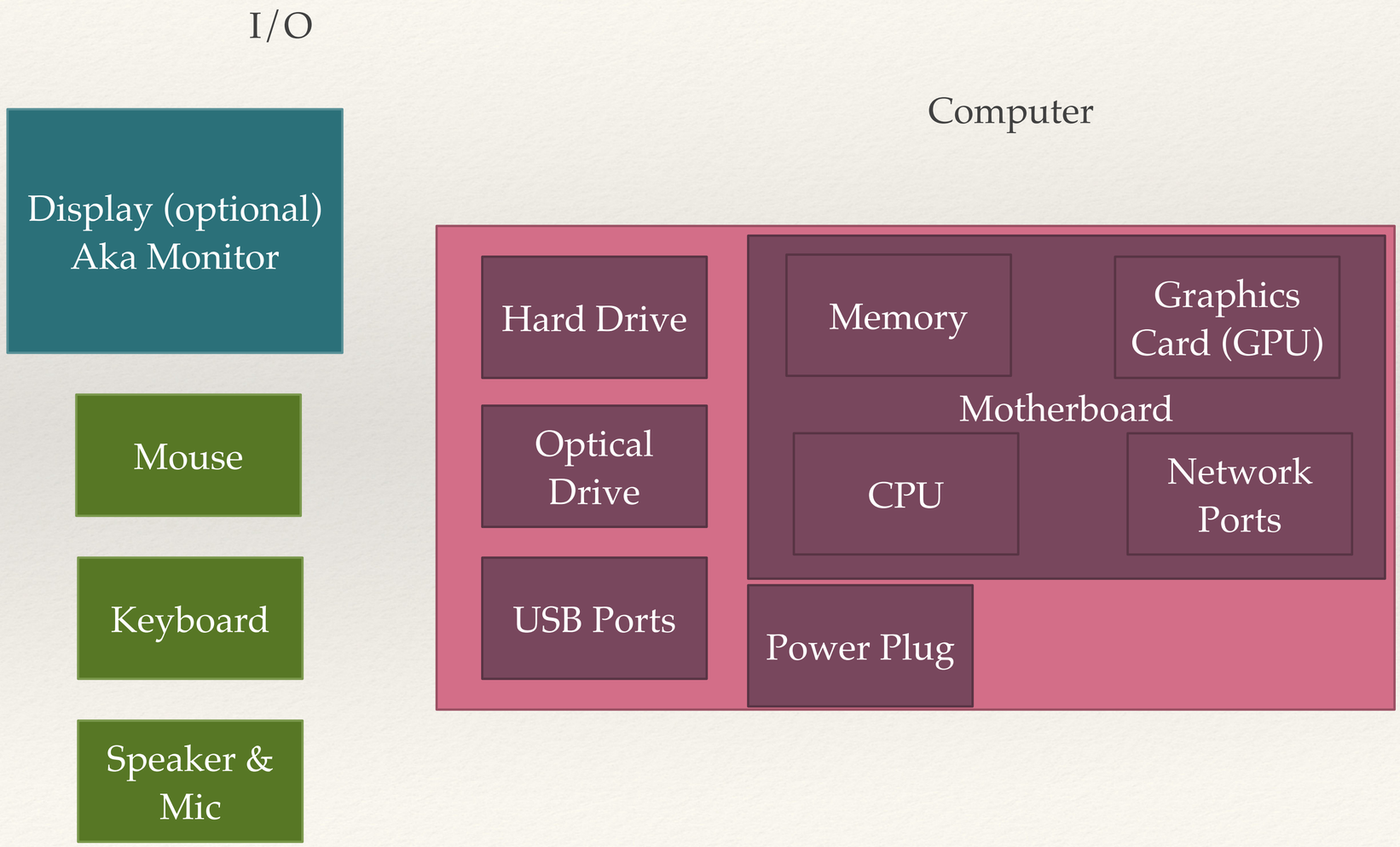
BG 3016 heritage.org

30-35% Productivity Improvement Using Technology Tool



Source: International Data Corporation (IDC); McKinsey Global Institute analysis

What is a computer?

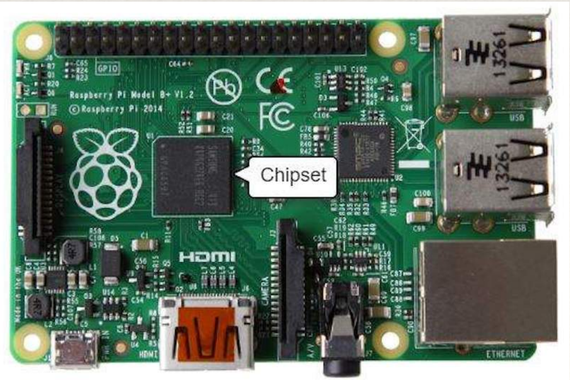


Motherboard

What it is: All components of a computer communicate through a circuit board called the motherboard, as was mentioned above.

What it does: Think of the motherboard as the glue that holds everything else together.

The motherboard's video card and Central Processing Unit are contained in an integrated (built-in) chipset, shown in the picture below:



CPU

What it is: The CPU is often called the "brain" of a computer, thanks to its direct plug connection to the motherboard, and communication with all of the computer’s other components.

What it does: Whenever you write a line of code (in Python, Java, C++, or any other programming language), it's broken down into assembly language—which is a language that the processor can understand. It fetches, decodes, and executes these instructions.

And that’s where the CPU comes in—all the processes a computer handles are taken care of by the CPU.

Graphics GPU

What it is: It's not uncommon to hear gamers obsess over the next new graphics card, as these graphic cards make it possible for computers to generate high-end visuals like those found in the many different types of video games.

In addition to video games, though, good graphics cards also come in handy for those who rely on images in order to execute their craft, like 3D modelers using resource-intensive software.

What it does: Graphics cards often communicate directly with the display monitor, meaning a $1,000 graphics card won't be of much use if there isn't a high-end monitor connected to it.

Random Access Memory RAM

What it is: RAM, also known as volatile memory, stores data regarding frequently accessed programs and processes. (It's called volatile memory because it gets erased every time the computer restarts.)

What it does: RAM helps programs and games start up and close quickly.

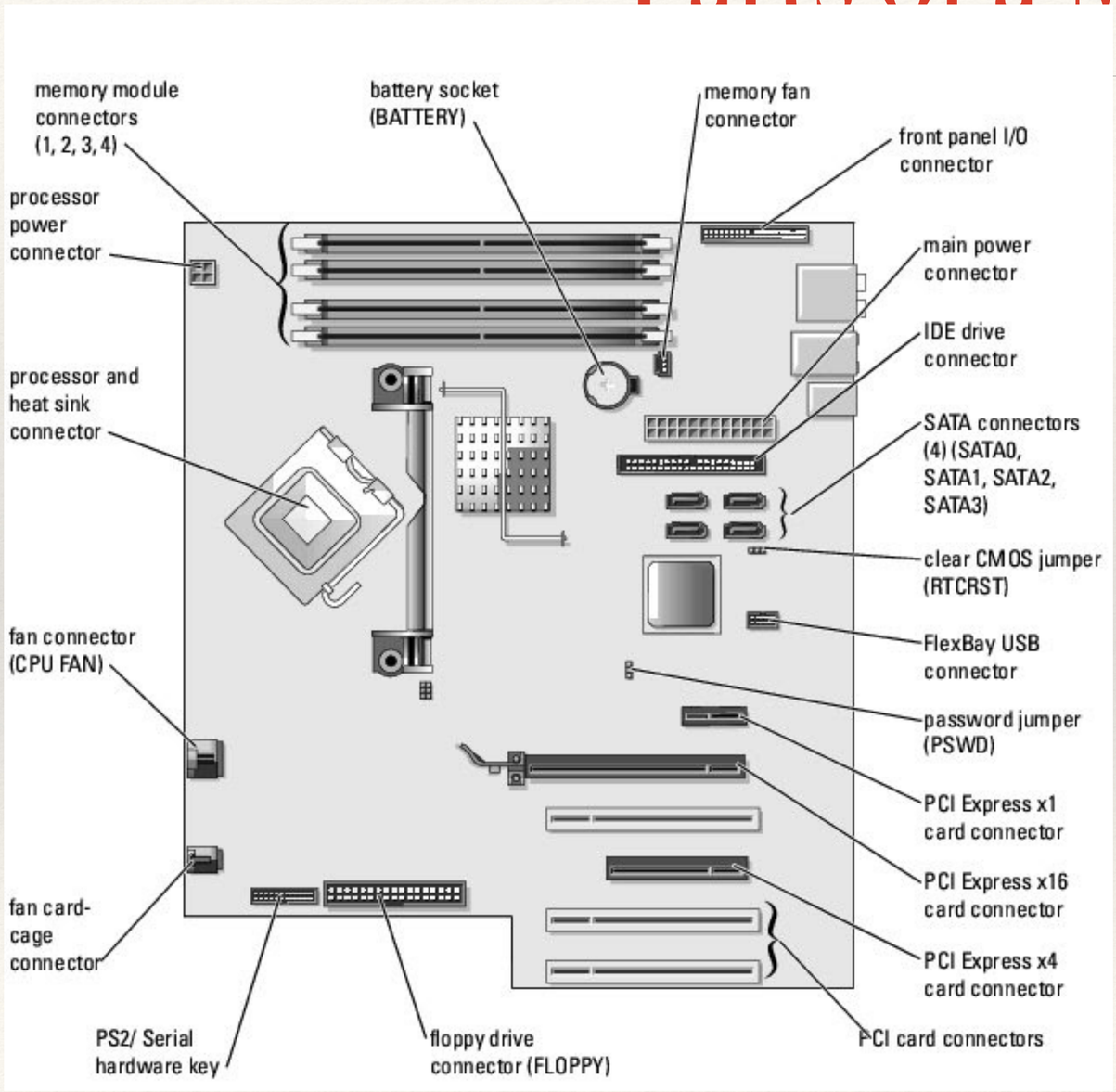
Storage

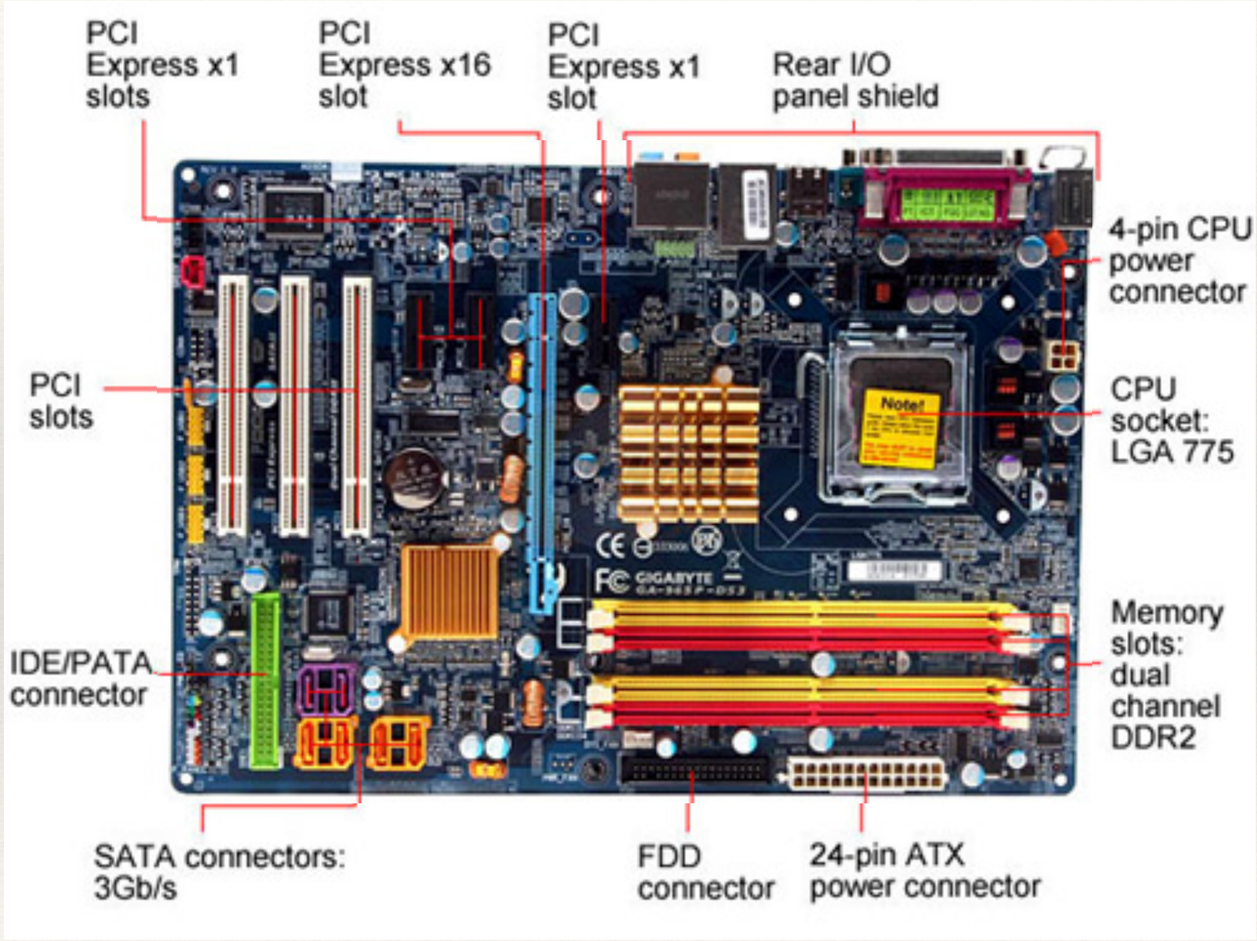
What it is: All computers need somewhere to store their data. Modern computers either use a Hard Disk Drive (HDD) or Solid State Drive (SSD).

What it does: HDDs are made of an actual disk onto which data is stored. The disk is read by a mechanical arm. (HDDs are cheaper than SSDs, but are slowly becoming more and more obsolete.)

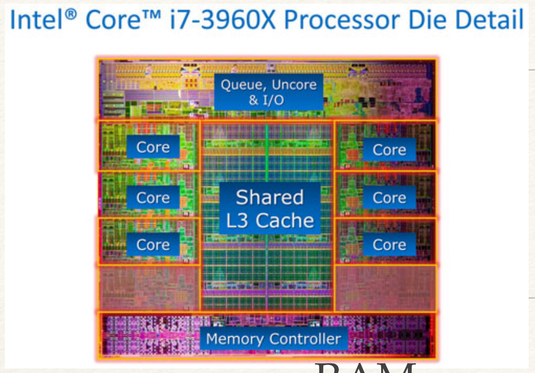
SSDs (think SIM cards) have no moving parts and are faster than a hard drive, because no time is spent waiting for a mechanical arm to find data on a physical location on the disk.

Parts of a Motherboard

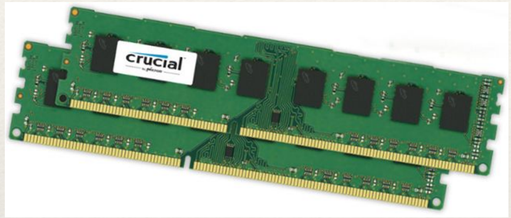




Types of Memory & Why?

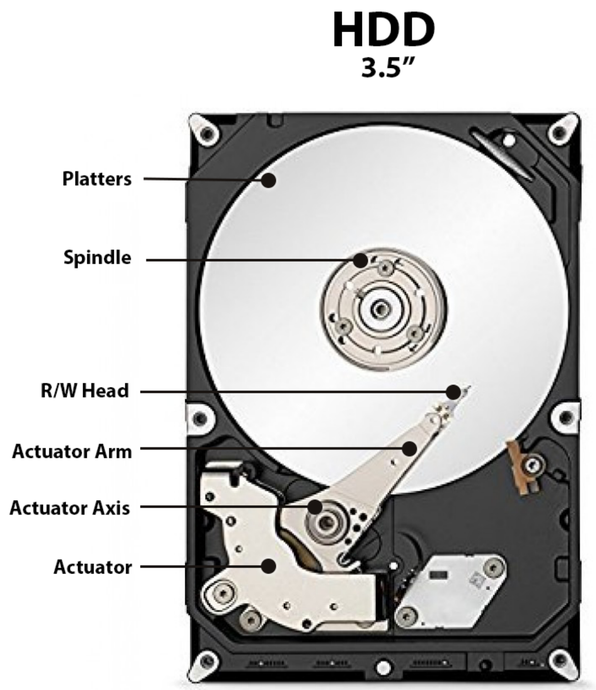


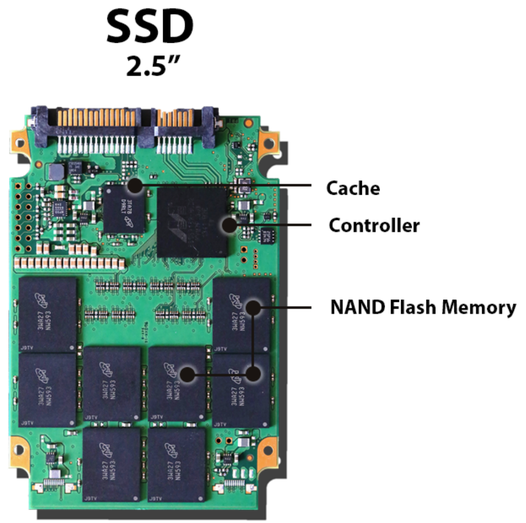
RAM



ROM

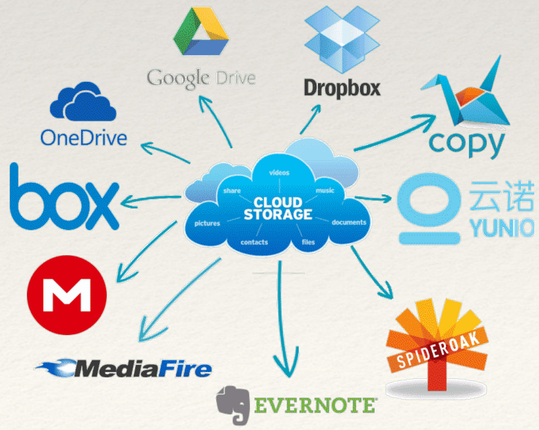
Storage



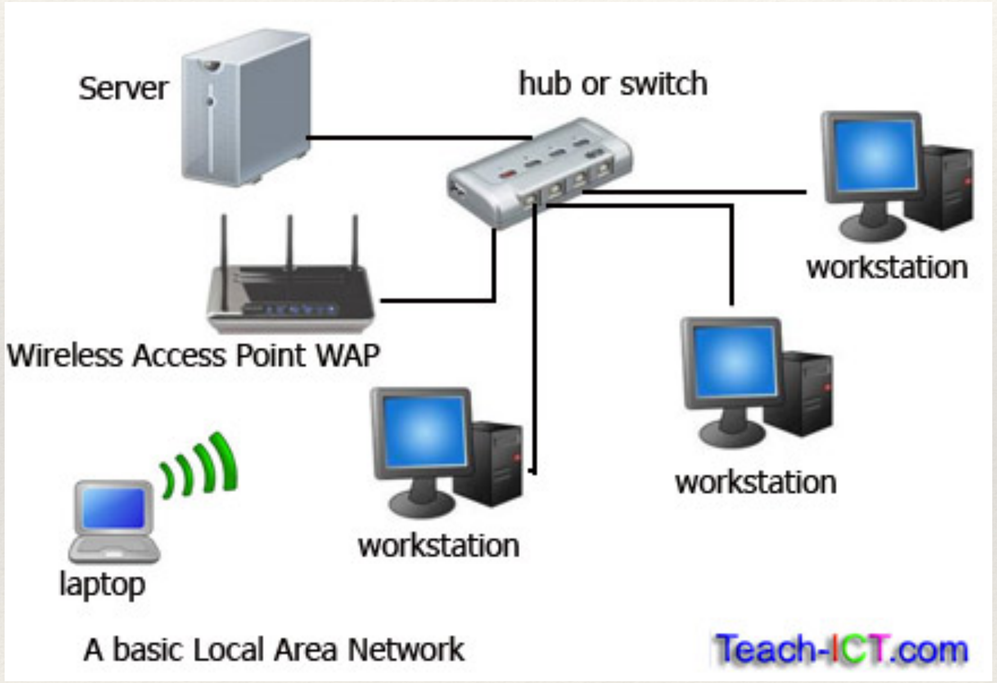


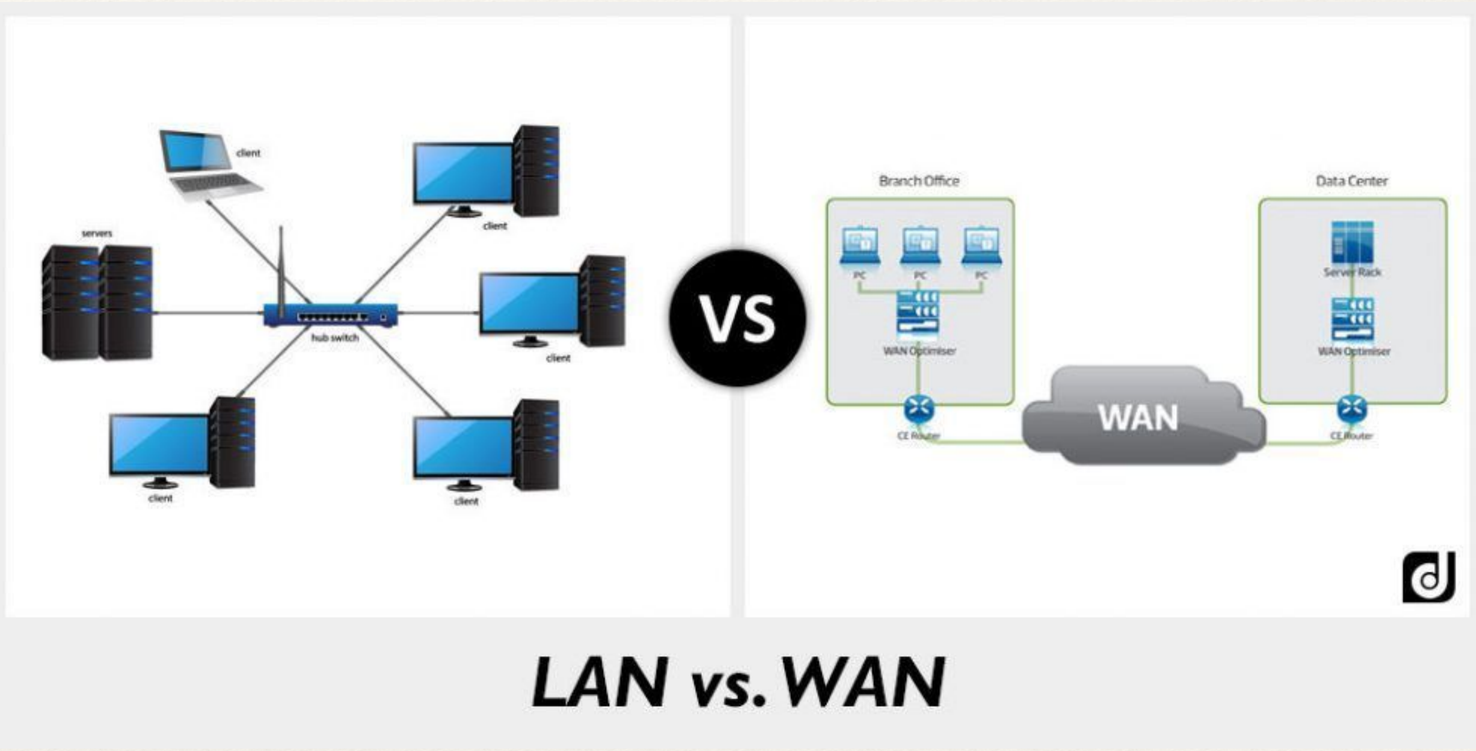
External Storage

External Storage

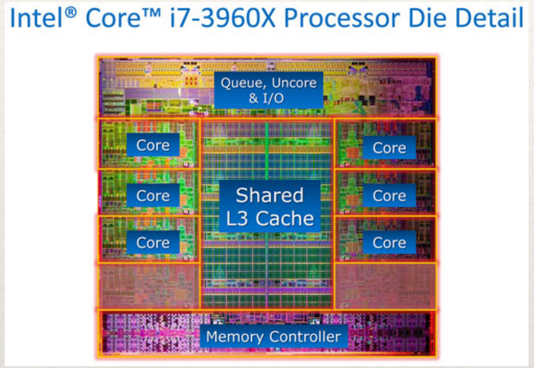


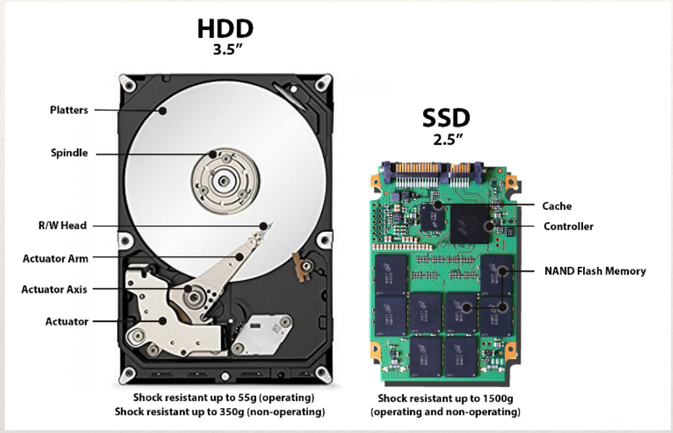
What is a network?
And why?



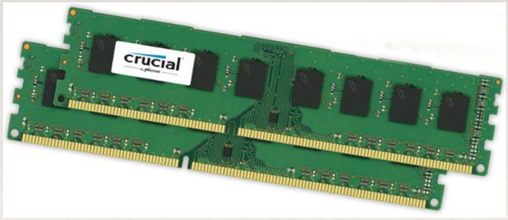


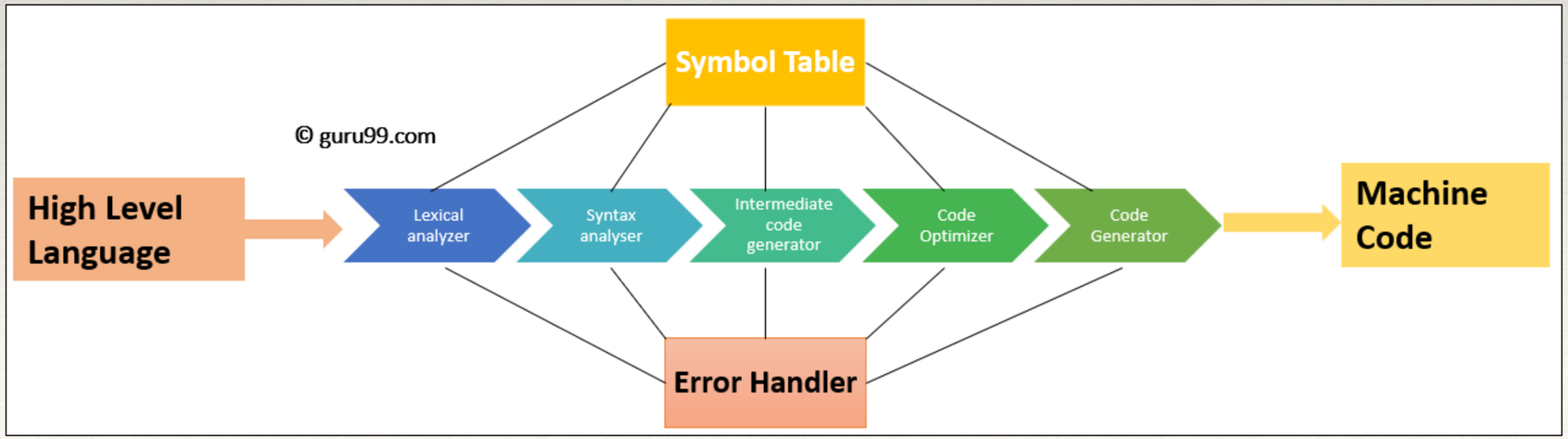
What is code? What is data? Where does Python run from?



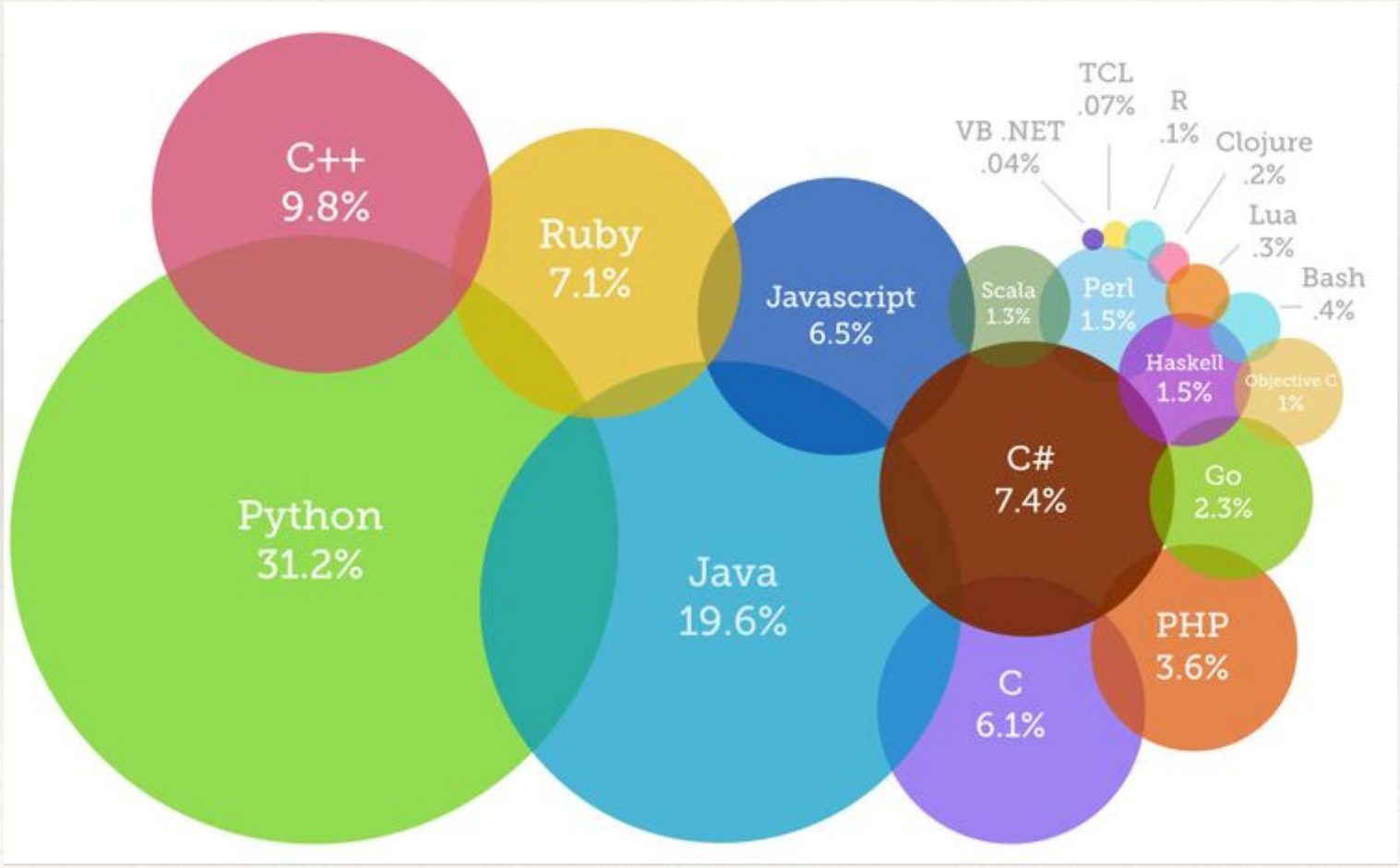


- Python is an interpreted language, but it still has to be converted into machine code, it just does it at run-time, not ahead of time.v





Programming Languages (by popularity)



Differences in programming languages:

1. Level of "abstraction" away from bytecode
2. Compiled vs Interpreted
3. Single-threaded vs. Multi-threaded
4. Strongly-Typed vs. Loosely Typed
5. Syntax (how you write stuff + things it can let you do)

Why Python?

1. Level of “abstraction” away from bytecode
    - Python is very abstract and therefor easier to learn and FAST to write.
2. Compiled vs Interpreted
    - Interpreted, so it is easy to see what is going on. Also makes code portable.
3. Single-threaded vs. Multi-threaded
    - Single-threaded means no need to deal with concurrency complexity (simultaneously running code)
4. Strongly-Typed vs. Loosely Typed
    - Loose typing means no need to declare types… makes it faster to write code and to understand.
5. Syntax (how you write stuff + things it can let you do)
    - Tabbed syntax makes code readable. Millions of libraries means there is nothing you cannot do in python.
    - Can make web apps, apps, and so much more!
6. Popularity!!!
    - Python is one of the most popular languages and is used in ALL industries!!!

Your first Python commands.

print(“ something “)

input(“How old are you? “)

Variable Assignment:

a = 5

b = “joe”

c = 3

Variable Comparison:

a == 5; True

b == “Sally”; False

Let's Code!
"Hello World!"

- Instal Spyder
- **SUGGESTION: make a folder under C:\ called pythonprograms to store your programs AND data! cd pythonprograms to get there from command prompt**
  - Backup if not able, is to run python in browser: https://pyfiddle.io/ (switch to 3.x)
- **Get Python to say "Hello World!"**
- **Get Python to ask for your name, and say "Hello ….!" (your name)**

Hints:
print("")
name = input("what's your name? ")

Interpret vs File
Visual Studio Code

- Run Spyder

Boolean Logic Basics
Why?

NOT

x	x'
0	1
1	0

AND

x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

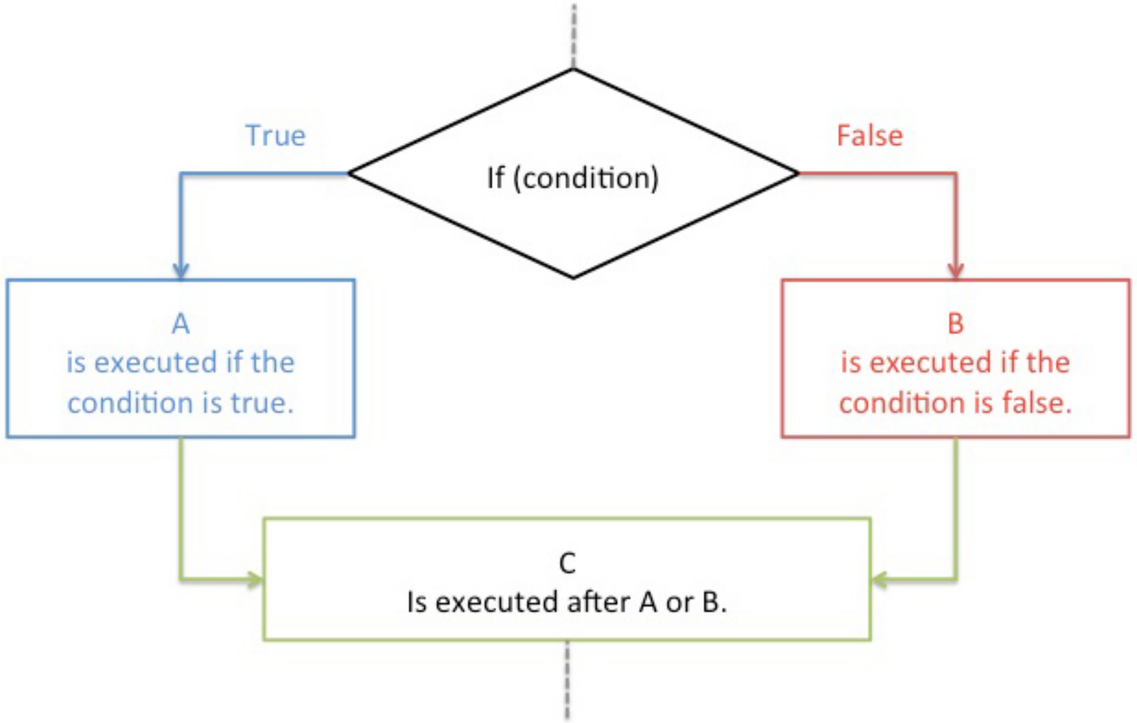
OR

x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

XOR

x	y	$x \oplus y$
0	0	0
0	1	1
1	0	1
1	1	0

Conditionals



Logical Comparison Operators

Six different comparison operators are used in mathematics and computer programming.

Condition	In Math	In Programming
A equals B	A = B	A = B or A == B
A is not equal to B	A ≠ B	A != B
A is less than B	A < B	A < B
A is greater than B	A > B	A > B
A is less than or equal to B	A ≤ B	A <= B
A is greater than or equal to B	A ≥ B	A >= B

Types

To change a type, use a typecast.
For example, To turn a variable age into an int do this:

int(age)

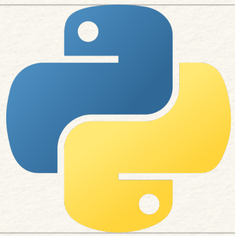
To turn a string or int into a floating point number:

float(age)

- What are types?
- What are some types?

In other languages, types are “declared” when you declare a variable, and it stays that TYPE forever. Python infers the type and lets you change it any time.. How does it do this? (hint: pointers)

Type	Example
• Numeric: Integer, Float	x = 10 x = 1.0
• String	x= ‘Mike’
• Boolean	y = True x = False
• List	my_list = [10, 20, 30]
• Tuple	my_tuple = (‘Brett’, ‘Cisco', ‘Cary’,2015)
• Dictionary	my_dict = {"one":1, "two":2}
• Lists in Lists	my_list2=[[10,20,30], [‘Cisco Live’, ‘May’, 2016]]



Let's Code!
Play with types

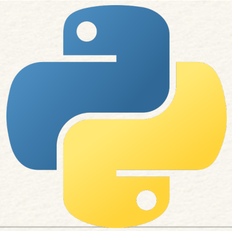
- https://www.learnpython.org/en/Variables_and_Types
- Go to the url above and do run and understand all the code there.
- Why can't you do this?

```
a = 5

b = "6"

print(a + b)
```

- Use typecast to Fix the above so that the print works correctly!



Let's Code!
"Hello [youngster/oldster]!"

- Get Python to ask for your age, and say "Hello oldster" or "Hello youngster" (youngster if you are 40 or less otherwise oldster).

Lists

A list is an ordered grouping of variables referenceable by their index.

- myintlist = [1,2,3]
- stringlist = [“one”, “two”, “three”]
- myintlist(0) #will print 1
- stringlist(1) #will print two

- What is special about lists?
- What can they hold?
- What is the “pointer” structure that you think a list has?