What is computing?

ATLS 1300

Thurs, Jan 16

Announcements

- Recitations are on Friday (**1B31**), times are correct:
  - 8am, 9:30, 11am, 12:30
- **McNair Scholar Program**
  https://www.colorado.edu/mcnair/
  - Due Jan 24th
  - First generation students, URMs (unsure? apply!)
  - Grad school training, GRE course, GRE fee paid, waived grad application fees
  - Sophomores and Juniors

The Rundown

- **This week** - programming basics
  - *Tech topic*- computer and human languages
  - *Creative topic*- learning rules to break them
- **Next week** - Common operators, variables, drawing tools
- Monday:
  - Course outline, next week’s lectures, Week 2 & 3 assignments
  - Recorded lectures up

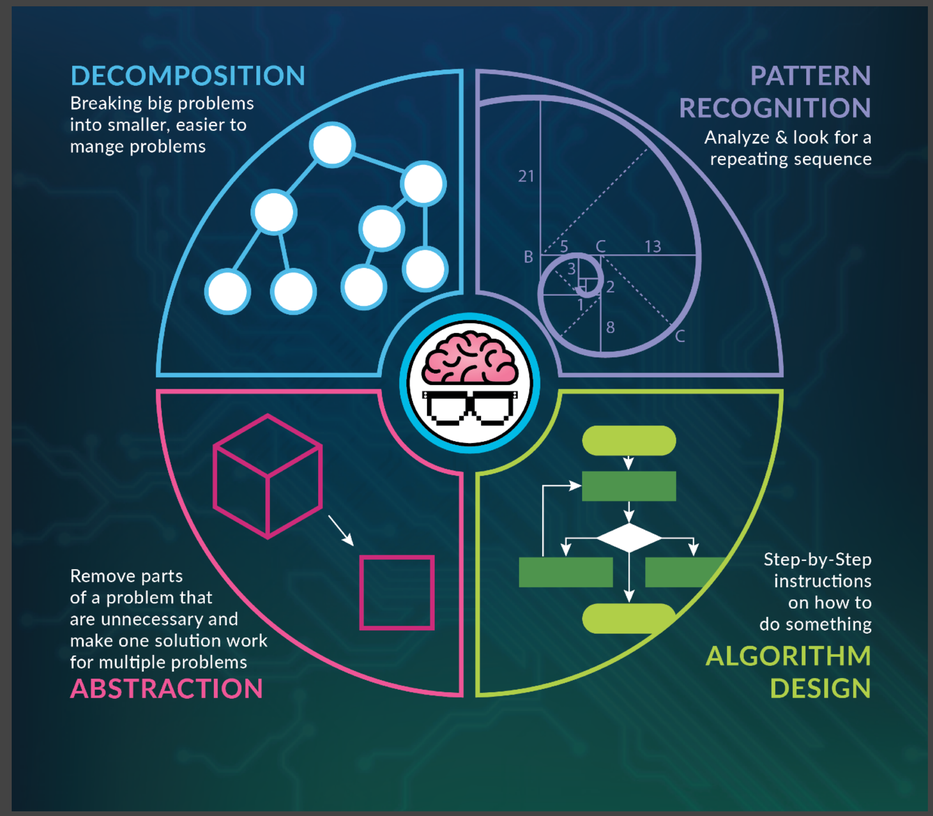
The Rundown

- Computer components
  - CPU
  - RAM
  - HD
- Machine language, human language, and the bridge between
- ID(L)Es
- The first program

What is computation?

Hint: What field (not CS) deals with all 4 of these sections of computational thinking?

- Math!
- At each step, you'll be turning your goals into mathematical representations:
  - Numbers
  - Mathematical equations
  - Sequences

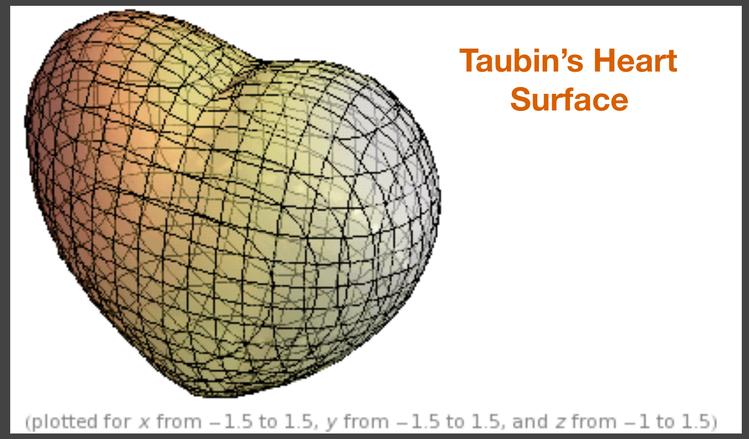


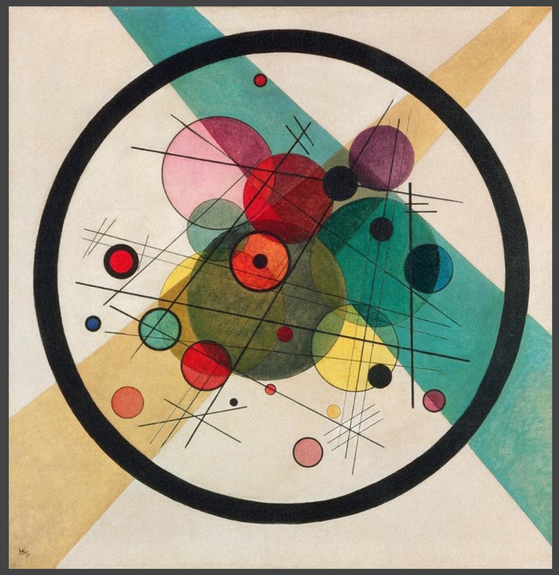
#ComputationalThinking

Brush up on ya math: https://youtu.be/6hjr4n4MCPA **<— Also on Canvas. That's weird.**

Math is art is math

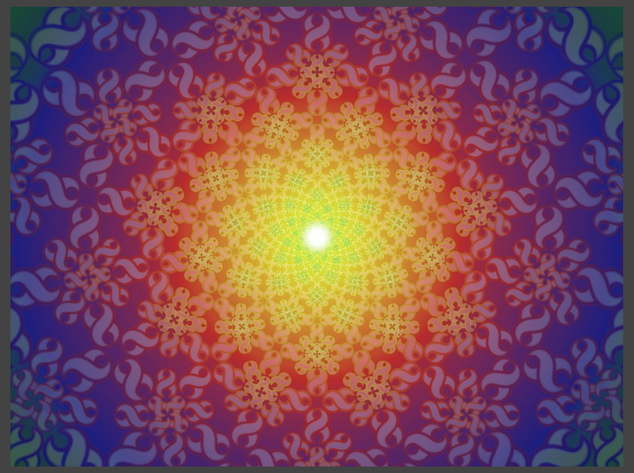
Recamán's Sequence
(Reddit)





Circles in a Circle
Wassily Kandinsky, 1923

Wall Drawing 51
Sol Lewitt, 1971 (Boston



Fibonacci spiral art
(chromatism.net)

How Do Computers Compute?

Computational Components

Memory

- Potentially long term data storage facility
- Data at a specific position (like an address)

Unpowered data storage

- Keep data *without* a current of electricity

Processor

- Do the tasks
  - Calculations
  - Throw and fetch data

Computational Components

Memory

- Potentially long term data storage facility
- Data at a specific position (like an address)

Unpowered data storage

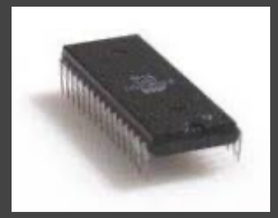
- Keep data *without* a current of electricity

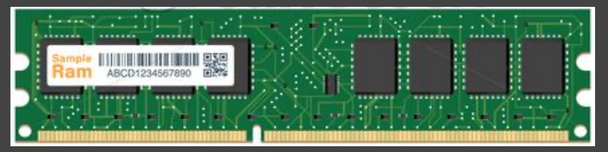
Processor

- Do the tasks
  - Calculations
  - Throw and fetch data

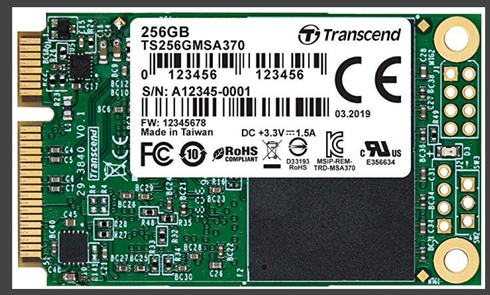
Computer Components

Random Access Memory





Hard Drive Disk | Solid State Drive



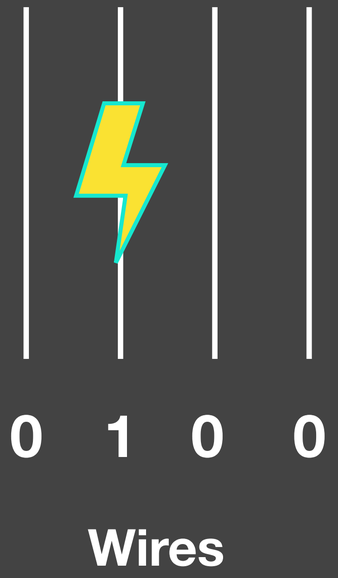
Central Processing Unit

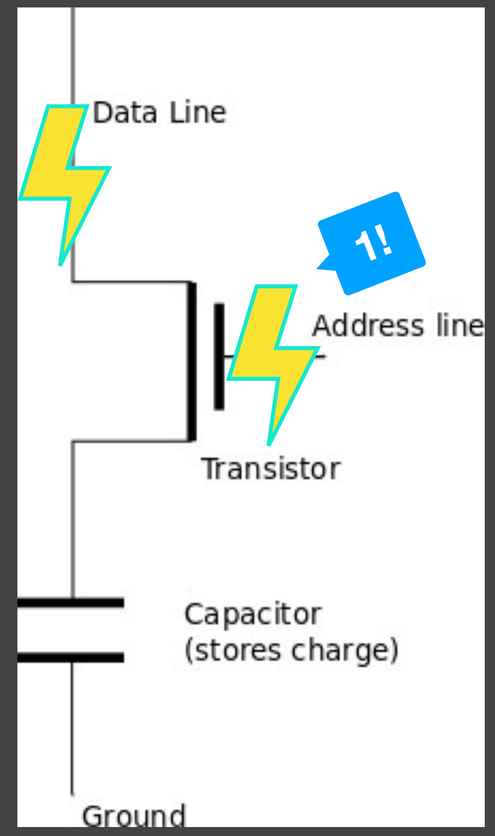
Binary is all RAM's fault.

"Wires" = transistor + capacitor component = **bit**

A bit has 2 possible values

- **ON = 1**
- **OFF = 0**





Binary is all RAM's fault.

If every **address** (bit location) can have 2 unique values…

…then computers ***count*** in increments of 2 (2^n).

- 1, 2, 4, 8, 16…

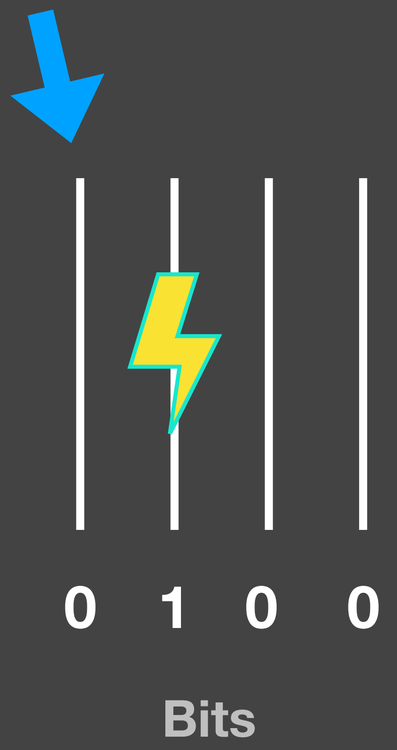
Humans reuse 10 unique values (10^n).

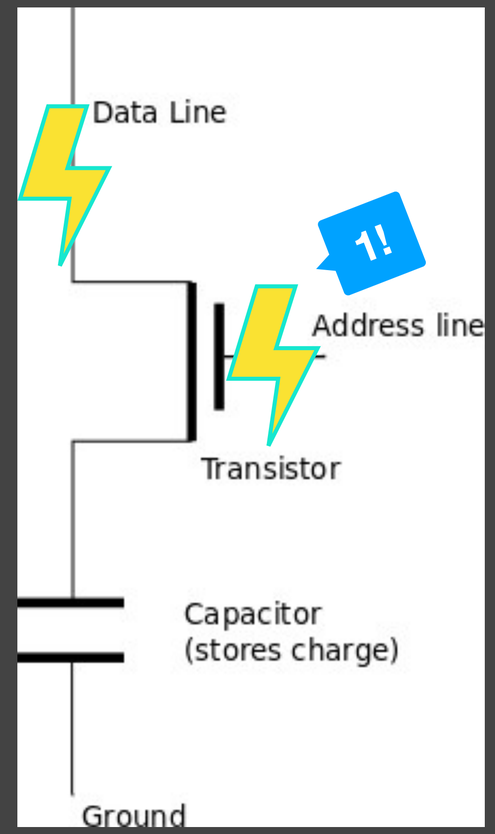
- 0,1,2,…,7,8,9

We arbitrarily arrange the bits in ***groups of 8***

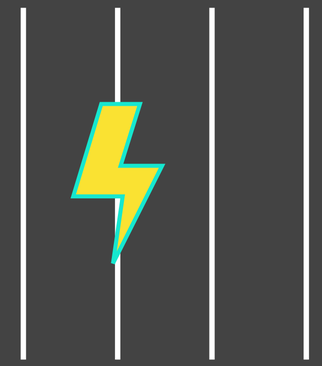
- This is called a **byte**

Thanks, IBM!





So what is binary?



0	9	0	0
10^3	10^2	10^1	10^0
1000	100	10	1

Value @ address

Position name

Non-notated #s

0	1	0	0
2^4	2^3	2^2	2^1
16	8	4	2

← By this

← Multiply this

(0 x 16)	(1 x 8)	(0 x 4)	(0 x 2)
0	8	0	0

Add these up → 8

Beefy RAM

We just worked out a 4-bit number…

8 bits = 1 byte

1024 bytes = 1 kilobyte (KB)

1024 KB = 1 megabyte (MB)

1024 MB = 1 gigabyte (GB)

So when you're computer has 16 "gigs" of RAM, it has

1,073,741,824 bits = transistor-capacitor pairs

Machine language is binary

- And very few people want to speak it
- Instead, we use **programming languages**

When you program…

High Level Languages

Easy to understand
Approach human language
Large memory footprint

Python

C++

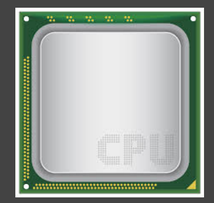
Java

Ruby

Javascript

Humans

Programming



Hardware

Low Level Languages

Fast processing
Require deeper knowledge
Low memory footprint

Assembly language
(drivers)

C

Integrated DeveLopment Environment

Software suite that uses Graphical User Interfaces (GUI) to consolidate tools needed to write and test programs.

- **Text editor** - a place to build code
    - **Command Line** - a place to explore and test individual lines of code
    - **Help** - a way to look up commands and functions
- Visualize images, graphs, animations
- **ATLAS IDEs:** SublimeText, Coda

Launch Spyder!

- Open Anaconda Navigator
- Double click Spyder

- Open terminal/command line
- Type `spyder`

- Search for Spyder and open it.

Spyder Layout

The Color Code

- `Numbers`
  - 0,1,2
  - Numerical values
  - Great for mathin’

- `Names:`
  - **Letters/words**
    - Represent variables (info stored in the RAM)

The Color Code

- **Comments:**
  - **#hashtag**
    - A way to add notes and citations
    - Descriptions of what a part of your program does.
- **Strings:**
- - “letters”/“words” A
    - Alphanumerical values.
    - You MUST add quotes (either ‘’ or “”) to get alphanumerical values!

The Color Code

- **Functions**
  - functionName()
  - Execute command with a **name** of a built-in function

- Built-in functions
  - `print()` - *displays* any value inside parentheses
  - `input()` - *displays* value inside parentheses AND *returns* the user's typed input
  - `abs()` - *returns* the absolute value of number inside parentheses
  - `len()` - returns the length of value inside parentheses
  - `id()` - returns the computer's identification number (address) for the bit of data stored in RAM

Try it out...

1. In the **command line**, type a number and press ENTER
2. Try adding and subtracting numbers, too.
3. Try typing a word and press ENTER. What happens?
4. Try using a **built-in function**

Built-in functions

- `print()` - *displays* any value inside parentheses
- `input()` - *displays* value inside parentheses AND *returns* the user's typed input
- `abs()` - *returns* the absolute value of number inside parentheses
- `len()` - returns the length of value inside parentheses
- `id()` - returns the computer's identification number (address)
  for the bit of data stored in RAM

Command Line is NOT the Text Area

- **Command Line**
  - Where you'll try new functions
  - Where you'll test your code
- **Text Area or Workspace**
  - Where you'll build your programs
  - Write stuff in here to save your programs!

Recitations

- You'll be reproducing the first computer program ever created:
  - Programming Challenge 00: Hello World (Canvas)
- Choose lab partner(s)
  - Keep partner for most of the semester (option to switch in Week 4)

Homework

1. Submit Programming Challenge 00 by **Wednesday, Jan 22nd, 5pm**
2. Read the assigned pdf by Tuesday - 9 pages
    - It will help to read it before tomorrow
    - NOTE: Images, not text, so dictation won’t work. Ask about accommodations.