Lecture 20: **Computer Architecture**

Tom Yeh
he/him/his

Outline

- What is a computer
- What is memory
- Memory hierarchy

Old School Computer

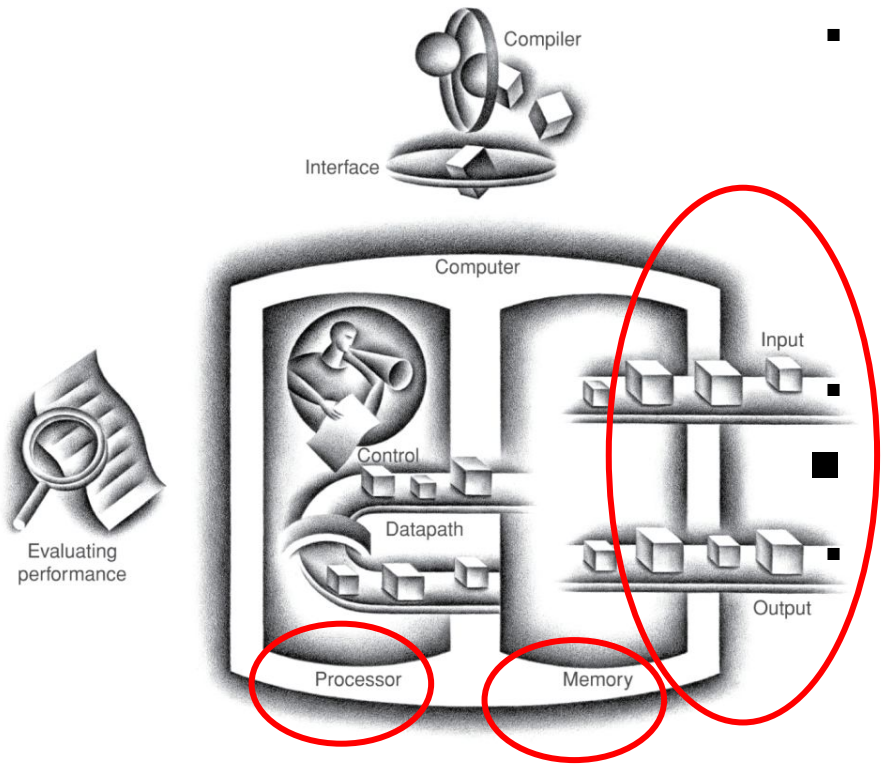
New School Computer

[Parkour Video]

Data Center is the Computer

Inside the warehouse data center

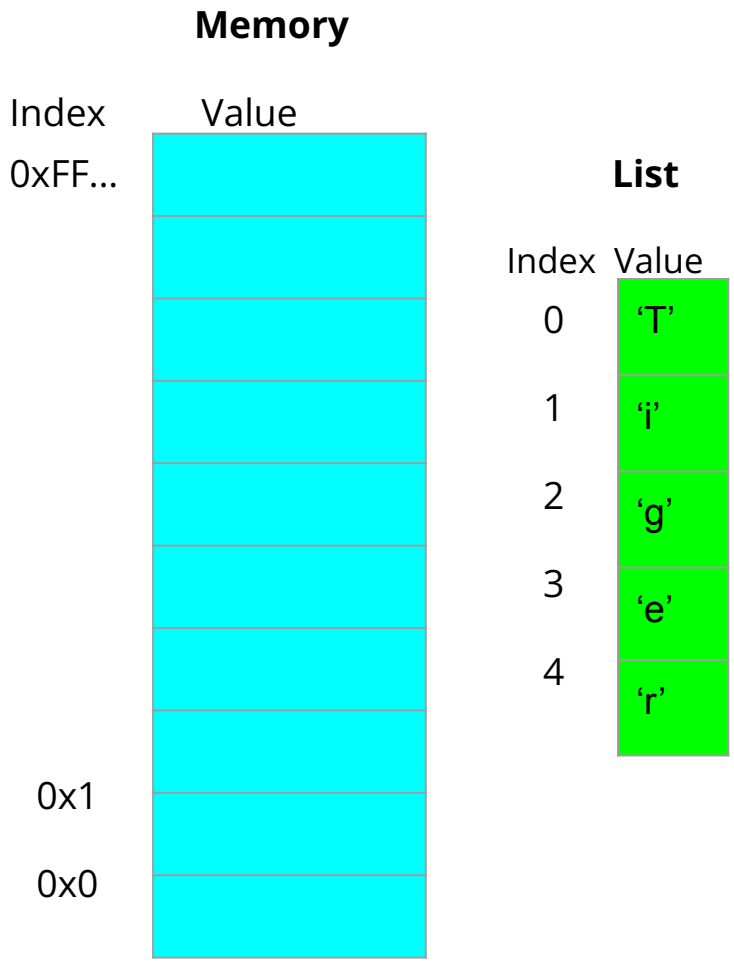
Components of a Computer



- Same 3 components for all kinds of computers
  - Processor (CPU)
  - Memory
  - I/O
- Processor executes instructions
- Memory holds data (inst)
- I/O transfers data to and from
  - Keyboard, mouse, network
  - Screen, printer, speaker
  - Flash drive, RAM,
-

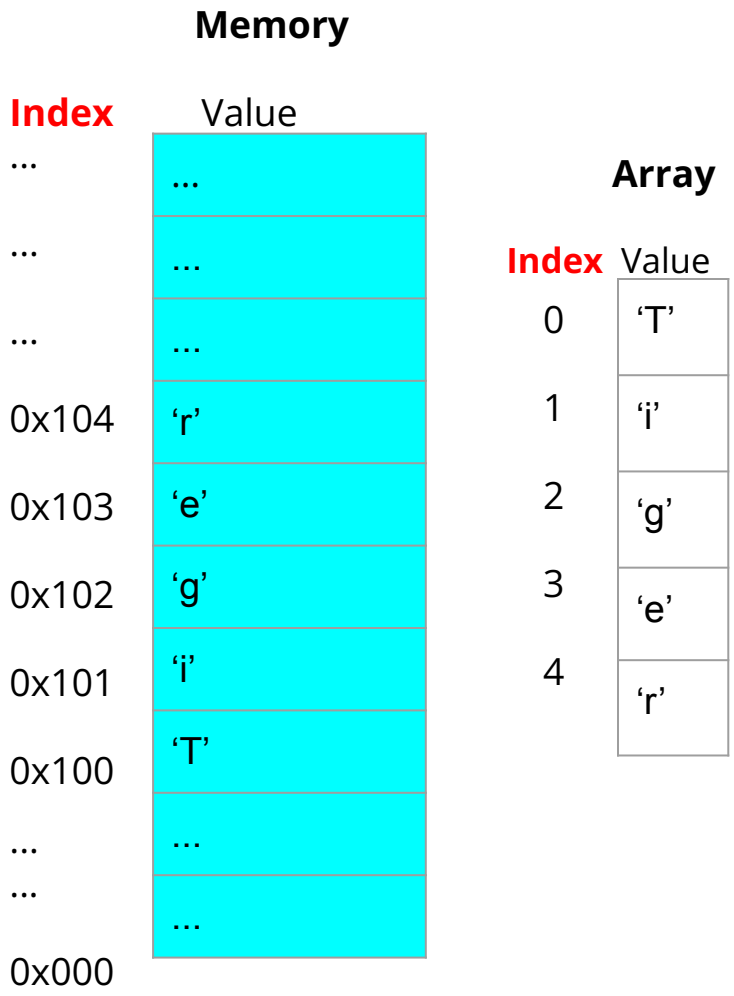
Memory

- Memory is like a big list of **bytes**
- **1 byte is 8 bits**
- **1 bit simply stores 0 or 1**



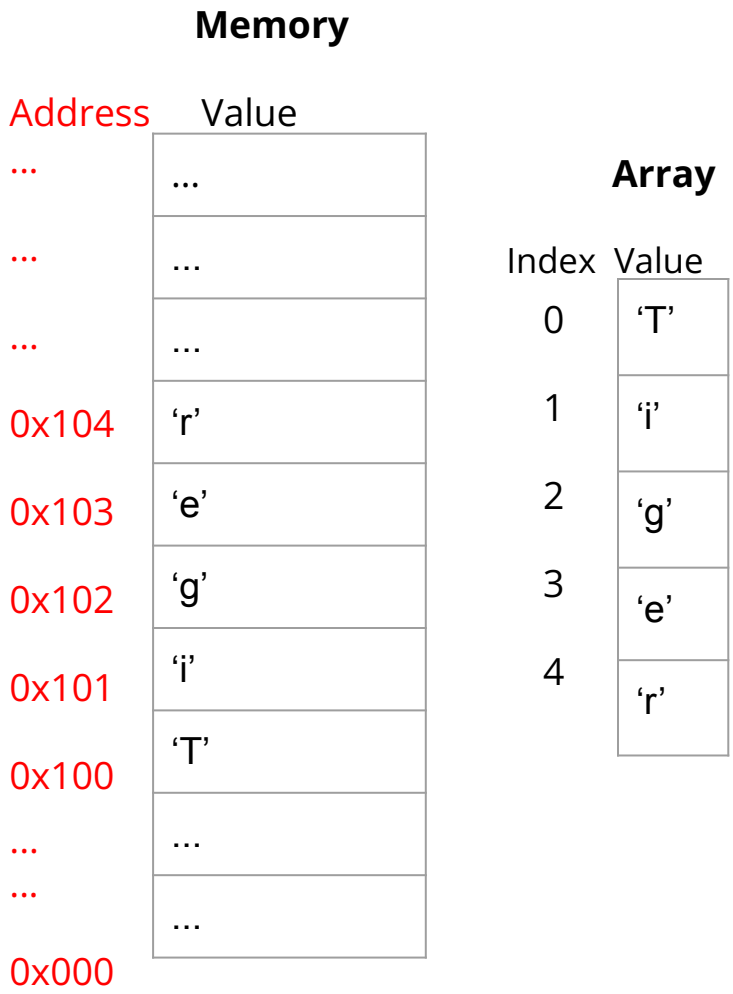
Memory

- Memory is a big list of bytes
- Each byte of memory has a **unique index**



Memory Address

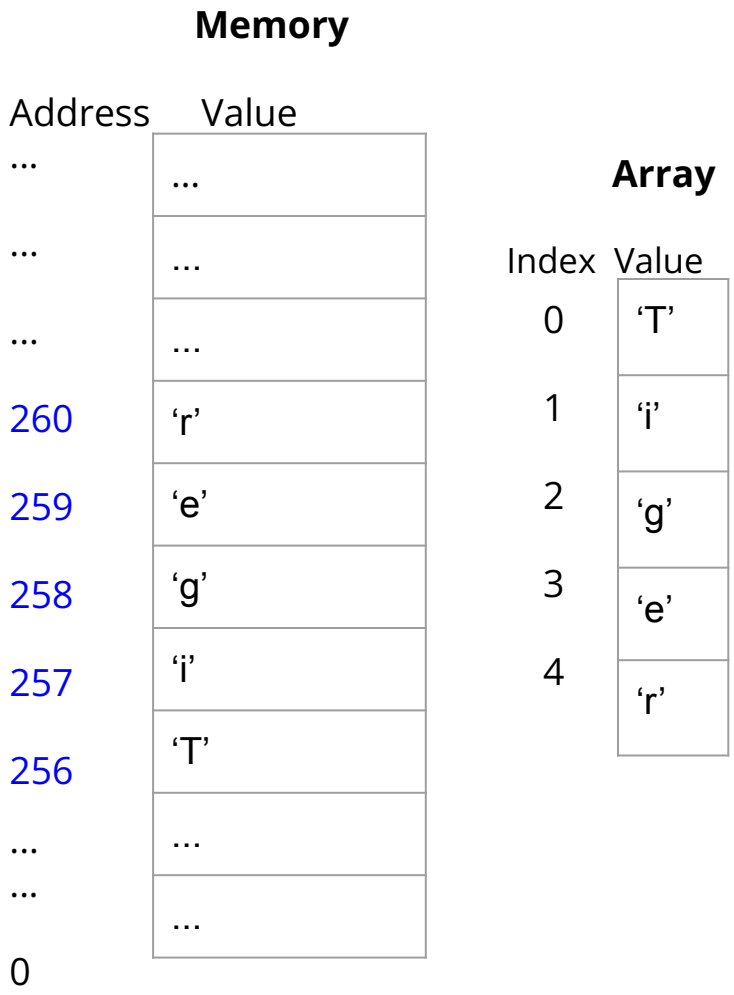
- Memory is a big list of bytes
- Each byte of memory has a unique index
- The unique index that points to different bytes is called the **memory address** (commonly written in hexadecimal)



Memory Address

- Memory is a big array of bytes
- Each byte has a unique index that is commonly written in hexadecimal
- The unique index that points to different bytes is called the memory address

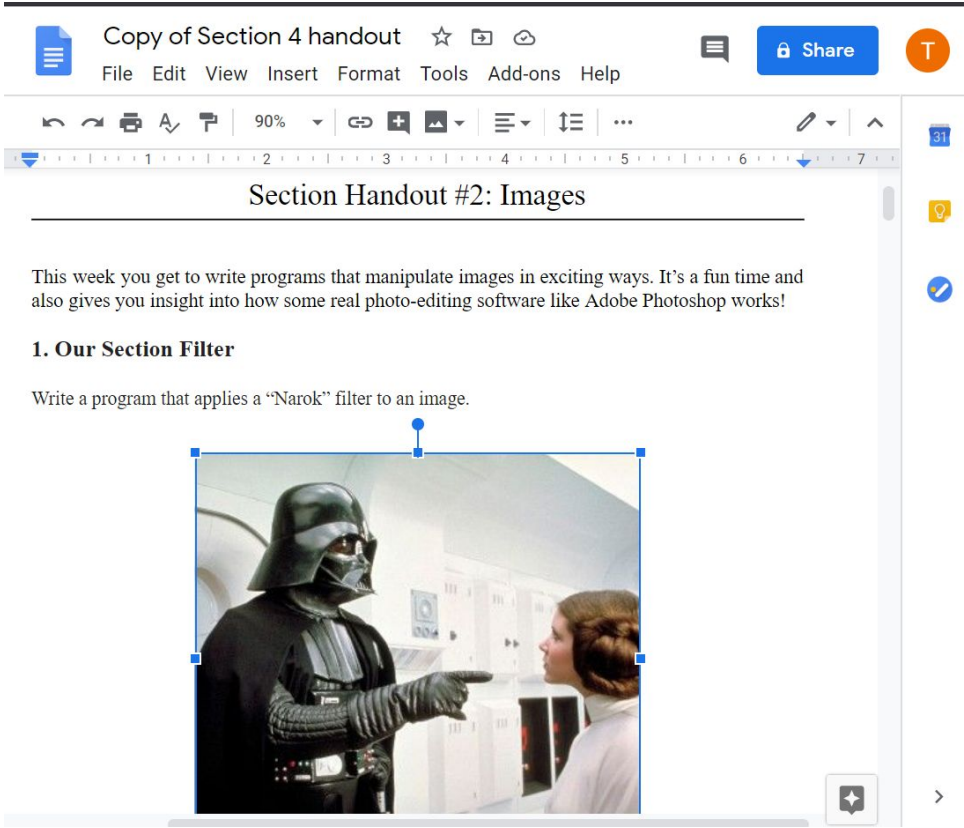
Key: A memory address is an index to each byte of memory



What is a Reference (pointer)?

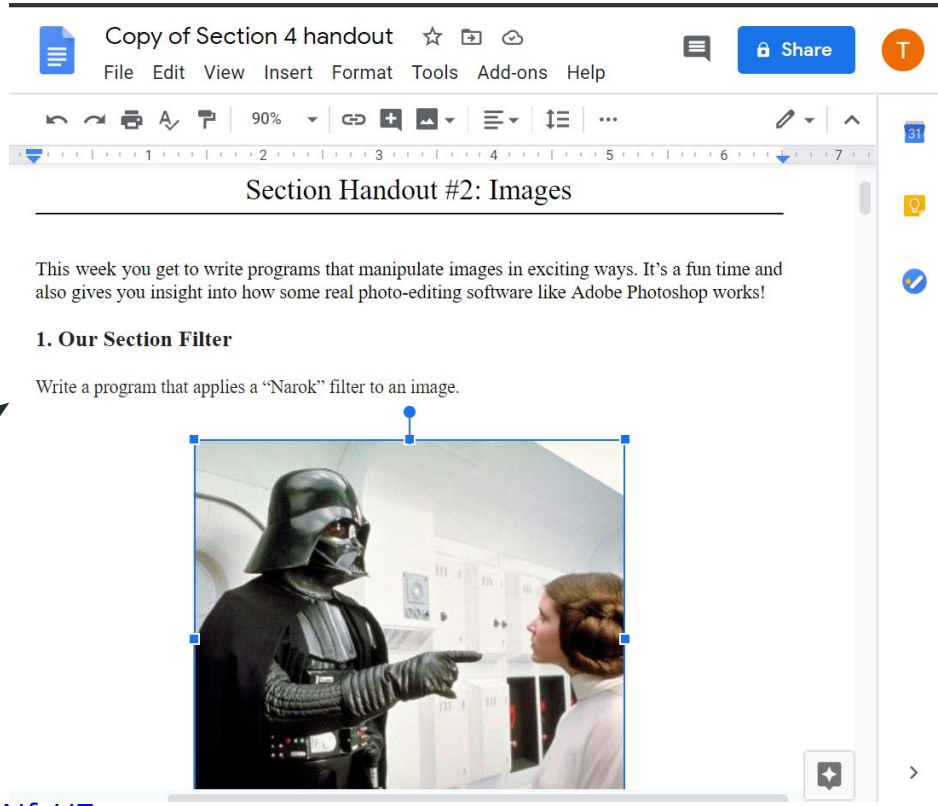
What is a Reference?

- Can think of a reference like an URL
- How do you share a google doc?



What is a Reference?

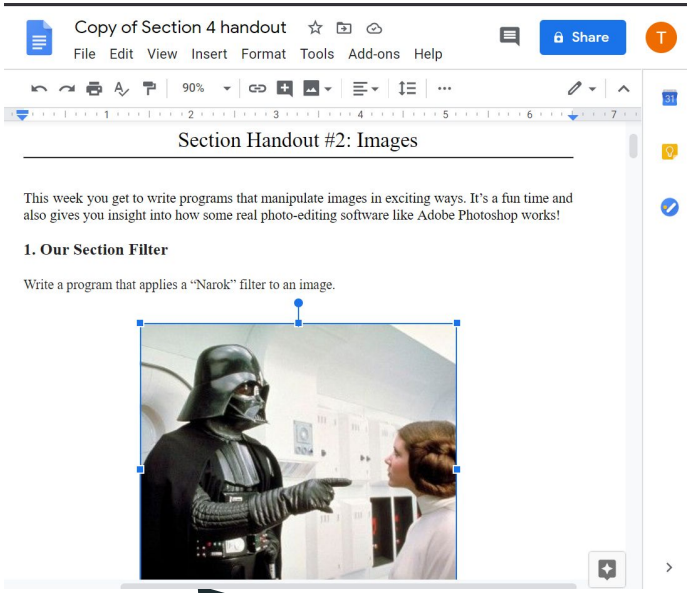
- Can think of a reference like an URL
- How do you share a google doc?
  - Click the blue share button
  - It creates an URL for you to share
  - The URL points to the actual document
  - Both parties can share the doc



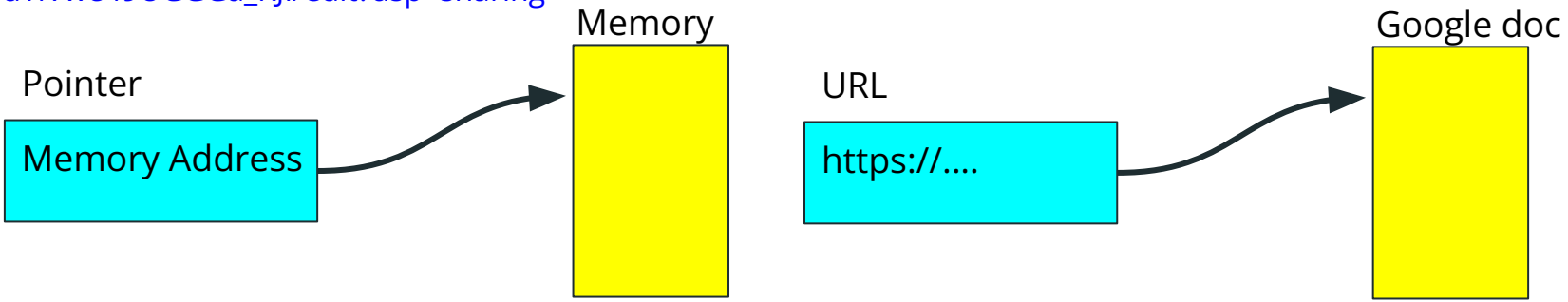
https://docs.google.com/document/d/1r-JRgHCOTpwJ8Gl4TDNfyUZreBuYlVwe49UGGGa_KJI/edit?usp=sharing

What is a Reference?

- Can think of a reference like an URL
- How do you share a google doc?
  - Click the blue share button
  - It creates an URL for you to share
  - The URL points to the actual document
  - Multiple parties can share the doc



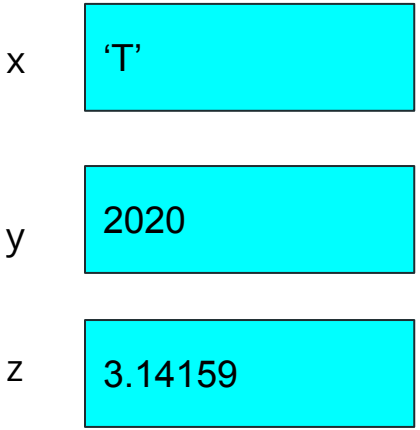
https://docs.google.com/document/d/1r-JRgHCOTpwJ8Gl4TDNfyUZreBuYlVwe49UGGGa_KJI/edit?usp=sharing

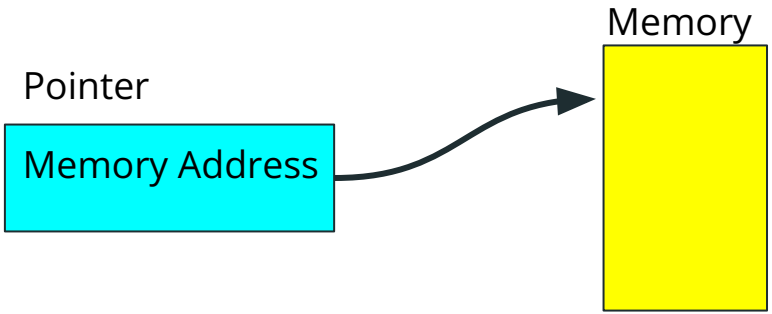


What is a reference (pointer)?

- A reference is a variable that stores a memory address

x = ‘T’
y = 2020
z = 3.14159

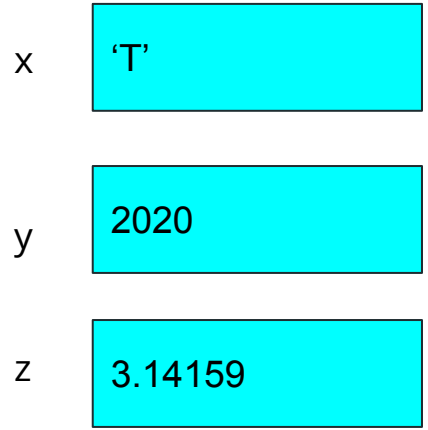


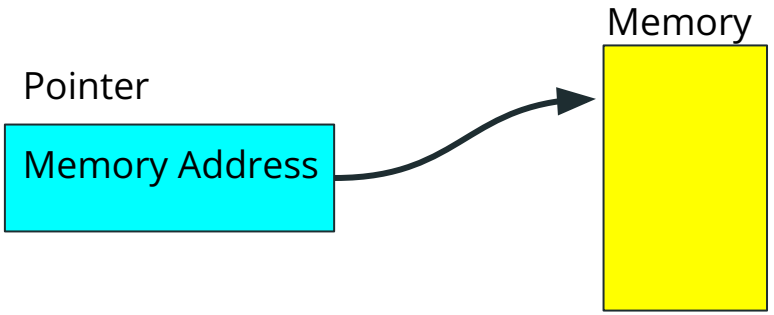


What is a Reference (pointer)?

x = ‘T’
y = 2020
z = 3.14159

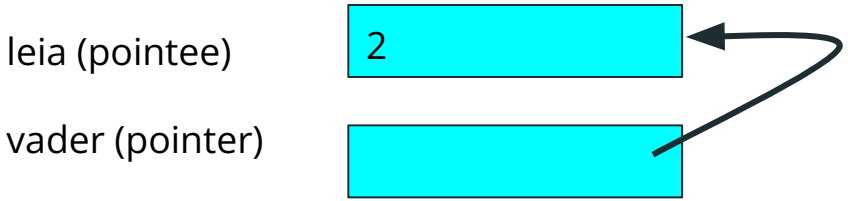
- A reference is a variable that stores a memory address
- A reference points to a location in memory





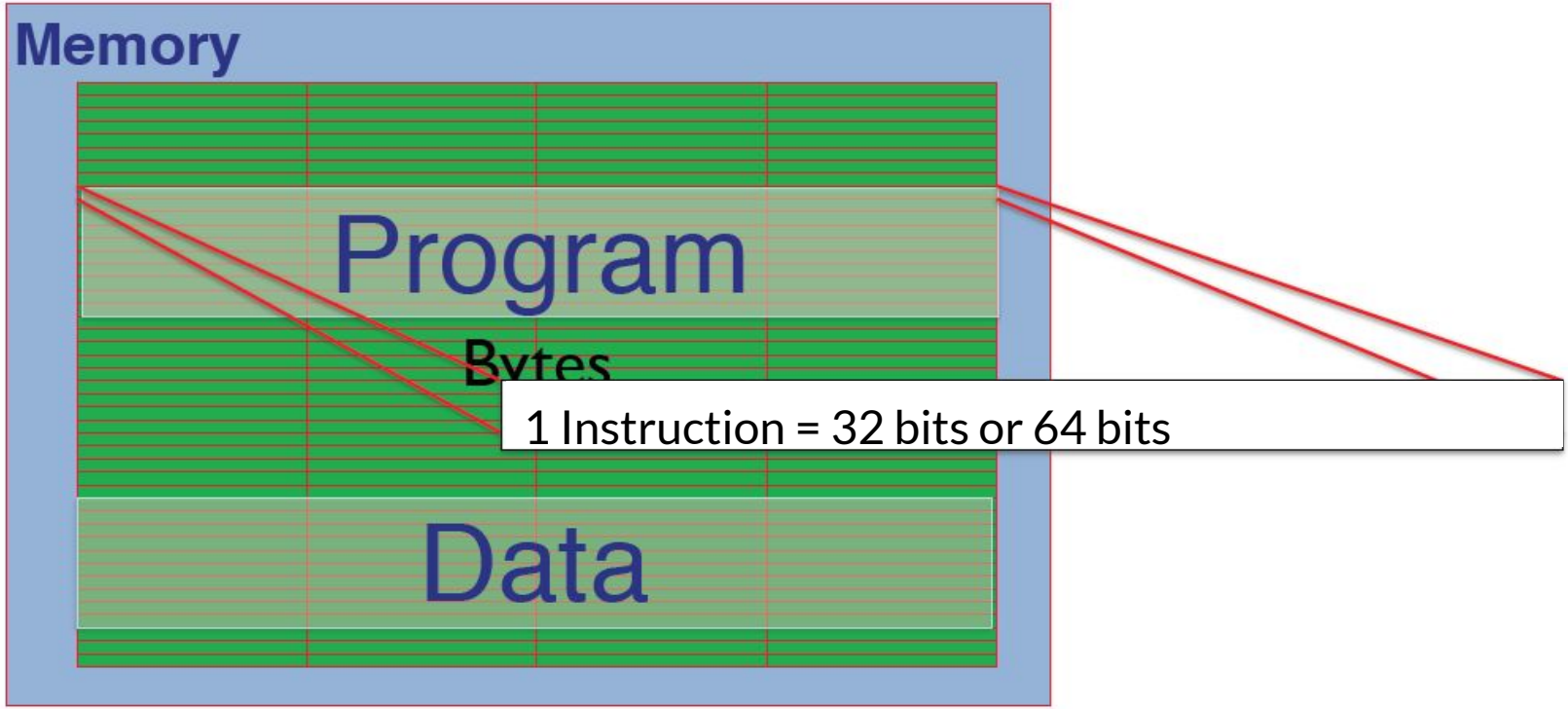
Pointer and Pointee

- Pointers do not store a value directly
- Pointers store a reference to another value
  - a memory address pointing to a location in memory
- The variable a pointer is pointing to is called the **Pointee**

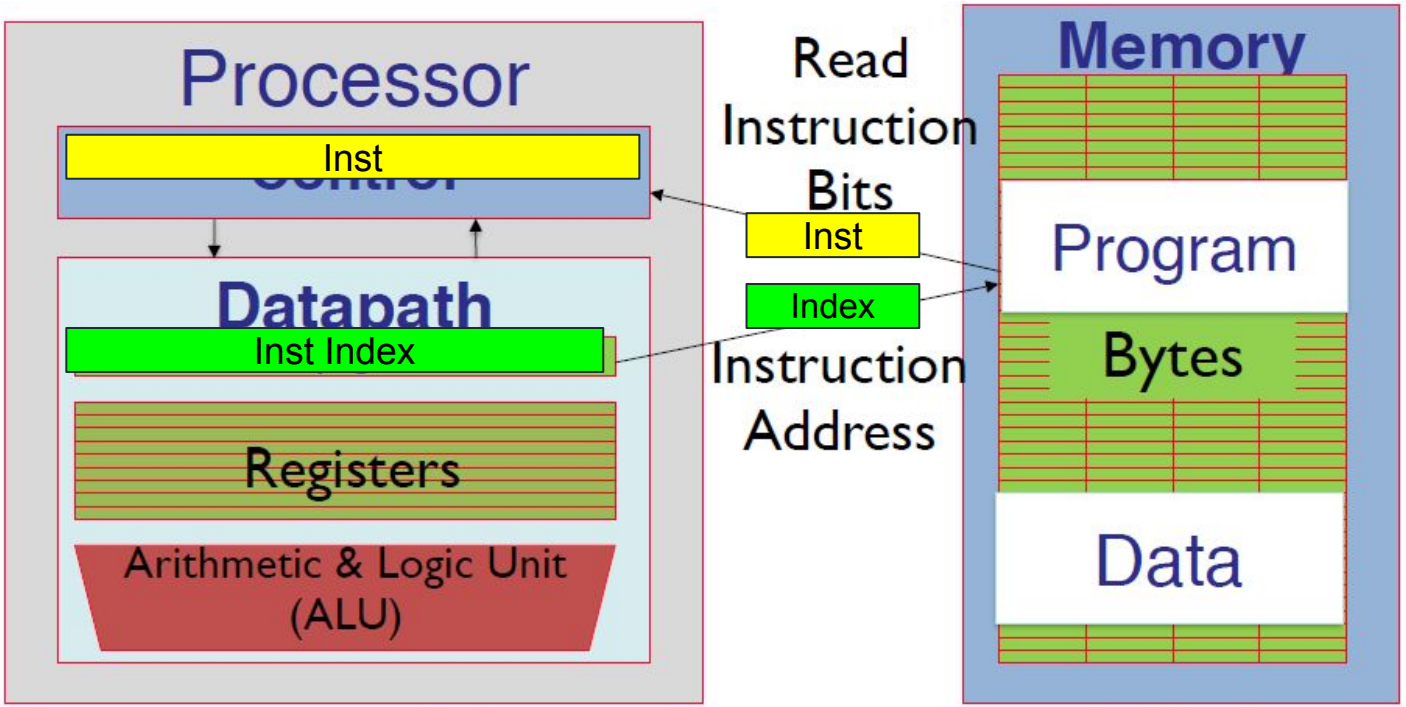


Von Neumann Architecture (same as mergesort inventor):

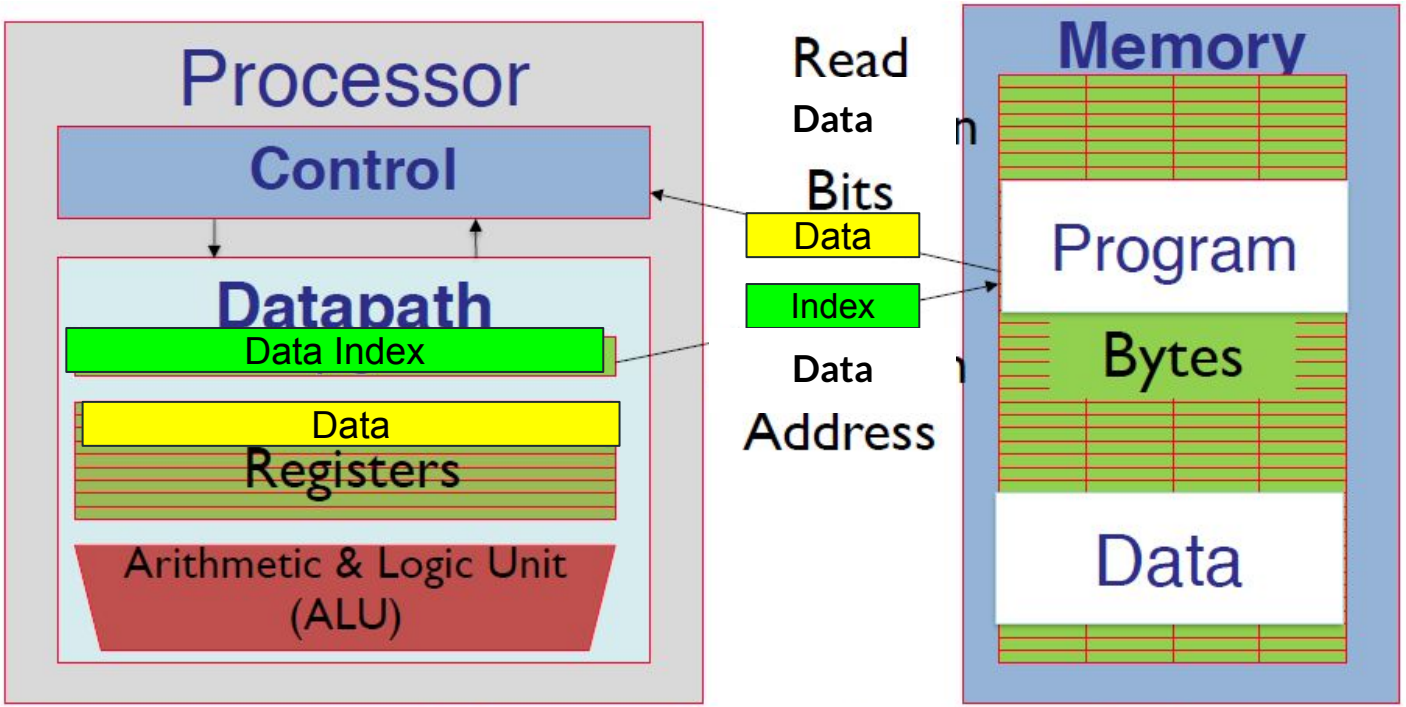
Program is stored in memory - think of memory as a large list



Program Execution:
Load instruction into processor (internal registers)



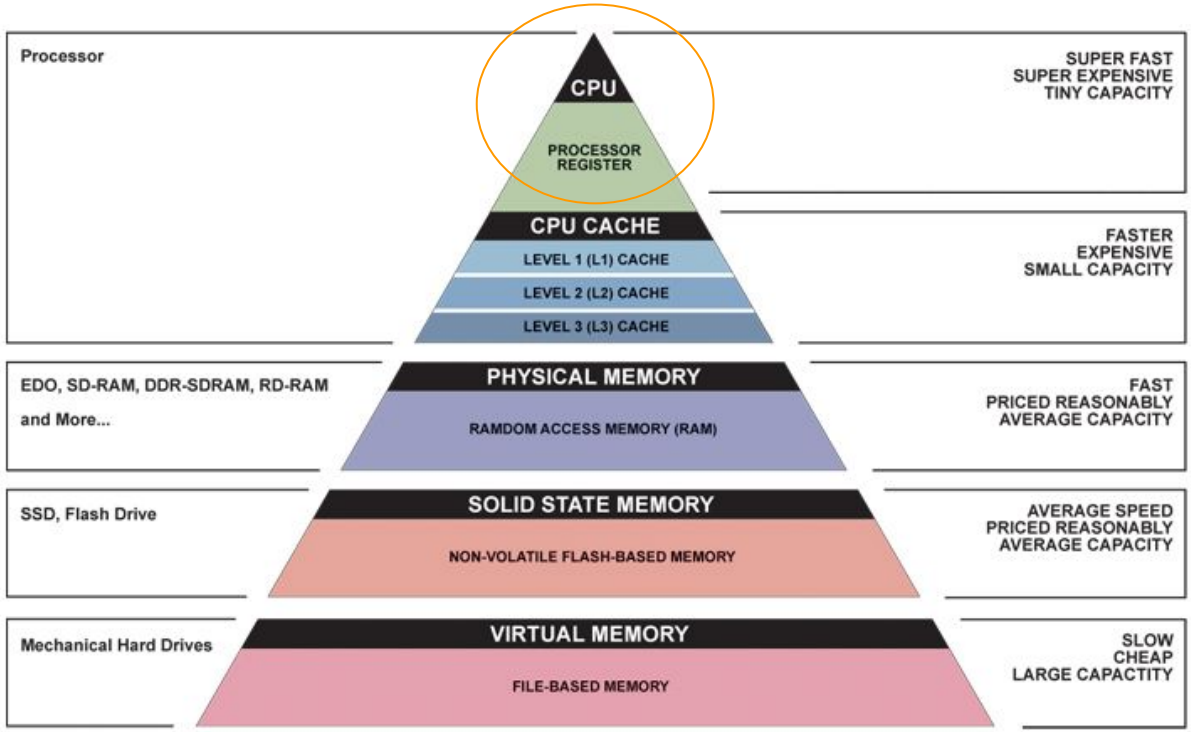
Program Execution:
Load data into registers



Typically, we only have 32 - 64 registers. You can think of these as hardware variables!

Principle of Locality – aka Memory Hierarchy

More Memory



Faster Memory

All data in layers above resides in the layer below
What should we store closer to CPU? Farther from CPU?

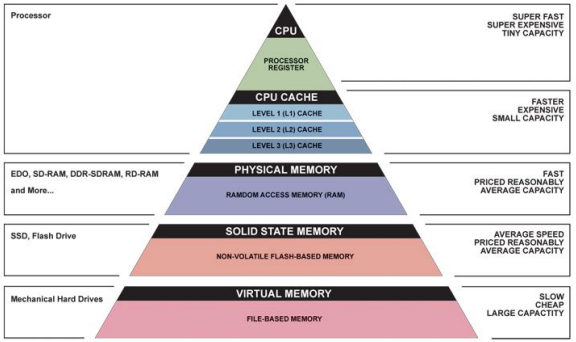
Key: Mem closest to CPU is fast, expensive, and scarce. Mem farthest is slow, cheap, plenty.

Sources of Locality

- Temporal Locality
  - If a piece of data is used, it tends to be reused

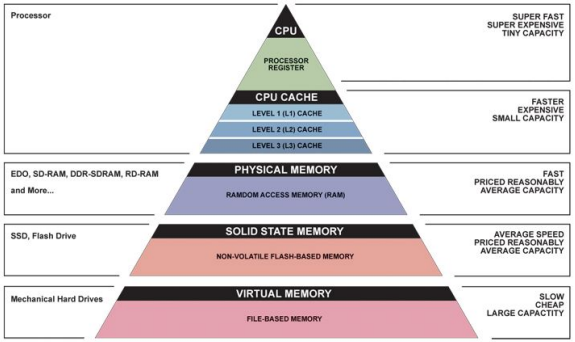
- Spatial Locality
  - If a piece of data is used, nearby data will also be used soon

Registers



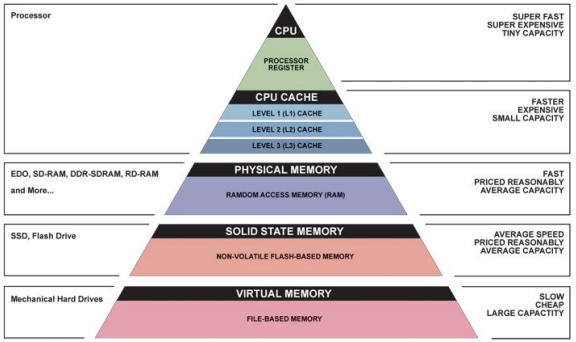
- Fastest, most expensive, tiny capacity
  - How fast is fast?
    - Registers operate at the same speed as a CPU's clock
      - A 3.33 GHz CPU has a clock period of 0.3ns
      - Access to registers are usually **single cycle** (0.3ns)
      - **C (speed of light) is $3*10^8$ m/s = 0.3 m/ns = 30 cm/ns = 10 cm/0.3 ns**
        - **Light can travel only 10cm in the span of of a clock period 0.3ns**
- 32 - 64 registers per processor core
- Each holds 32 - 64 bits of data

Cache



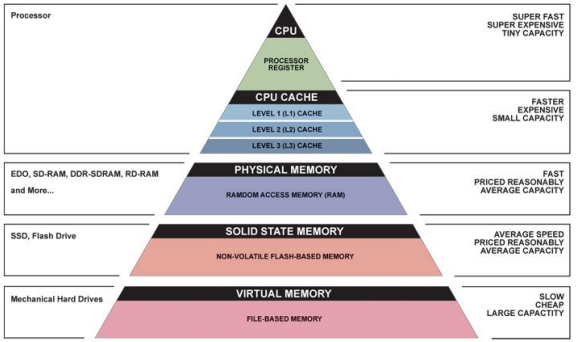
- Faster, expensive, small capacity
- Slower than registers, but faster than main memory
  - **10 - 100 CPU cycles**
- Typically, 1-3 levels (L1, L2, L3, etc.)
- 32-64 KB for L1, 128 - 512 KB for L2, 1MB+ for L3

Main memory (RAM)
Physical memory



- Fast, reasonably priced, average capacity
- Much slower than registers, but faster than disk
- 8 - 32 GB
- **100 - 500 CPU cycles**
- All programs and data must fit in memory
  - Use virtual memory when we need memory > physical memory
  - Virtual memory gives each program the illusion of having all memory space
  - Utilize disk to store data that do not fit into physical memory

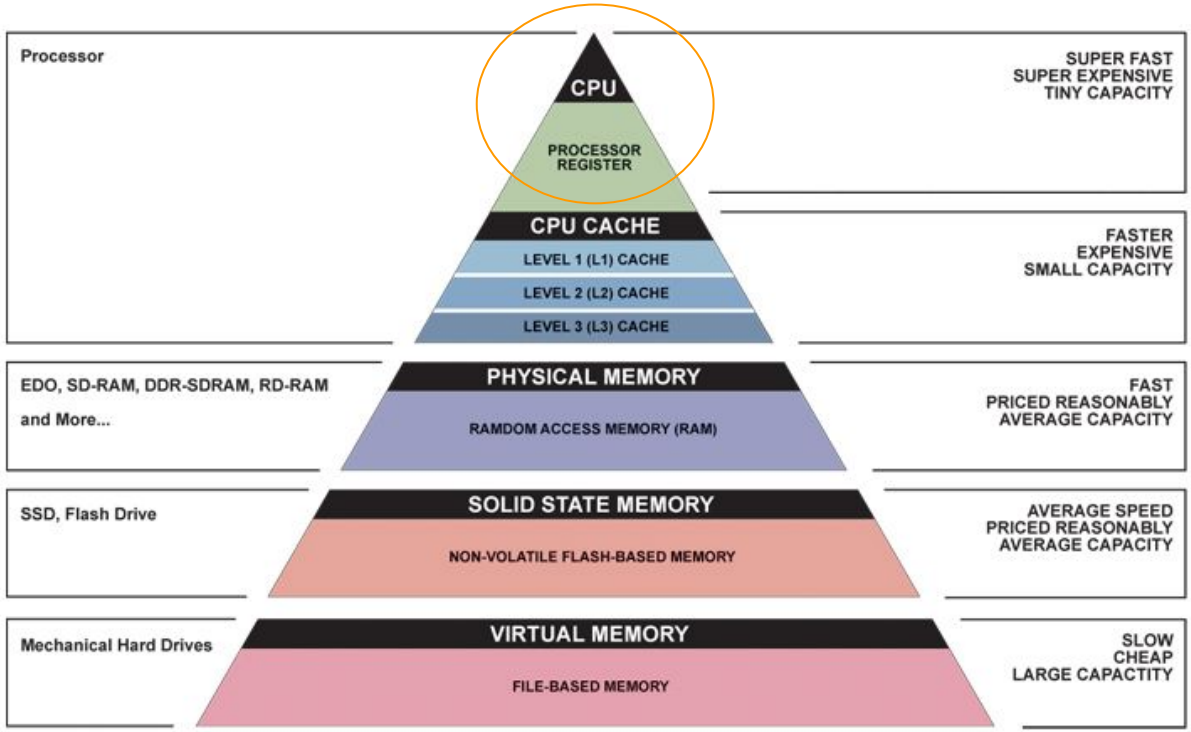
External memory (disk)



- Slow, cheap, large capacity
- Recent computers use solid state drives (SSDs)
- Hundred of GB to a few TB
- **20,000 CPU cycles latency**

#3 – Principle of Locality – aka Memory Hierarchy

More Memory



Faster Memory

All data in layers above resides in the layer below
What should we store closer to CPU? Farther from CPU?

Key: Mem closest to CPU is fast, expensive, and scarce. Mem farthest is slow, cheap, plenty.