Computer Systems

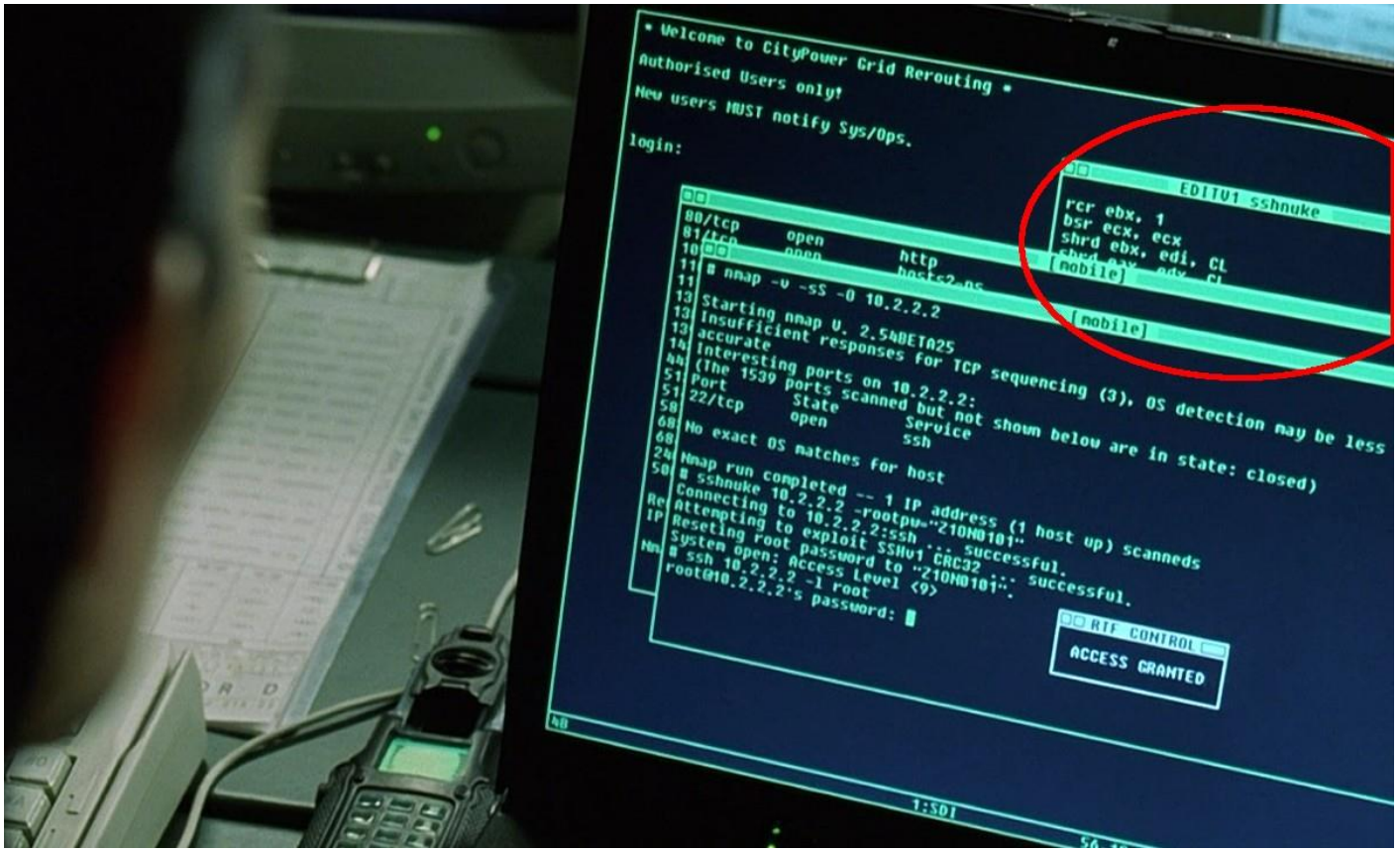
CS107

Cynthia Lee

Quarter in Review: CS107 Outcomes

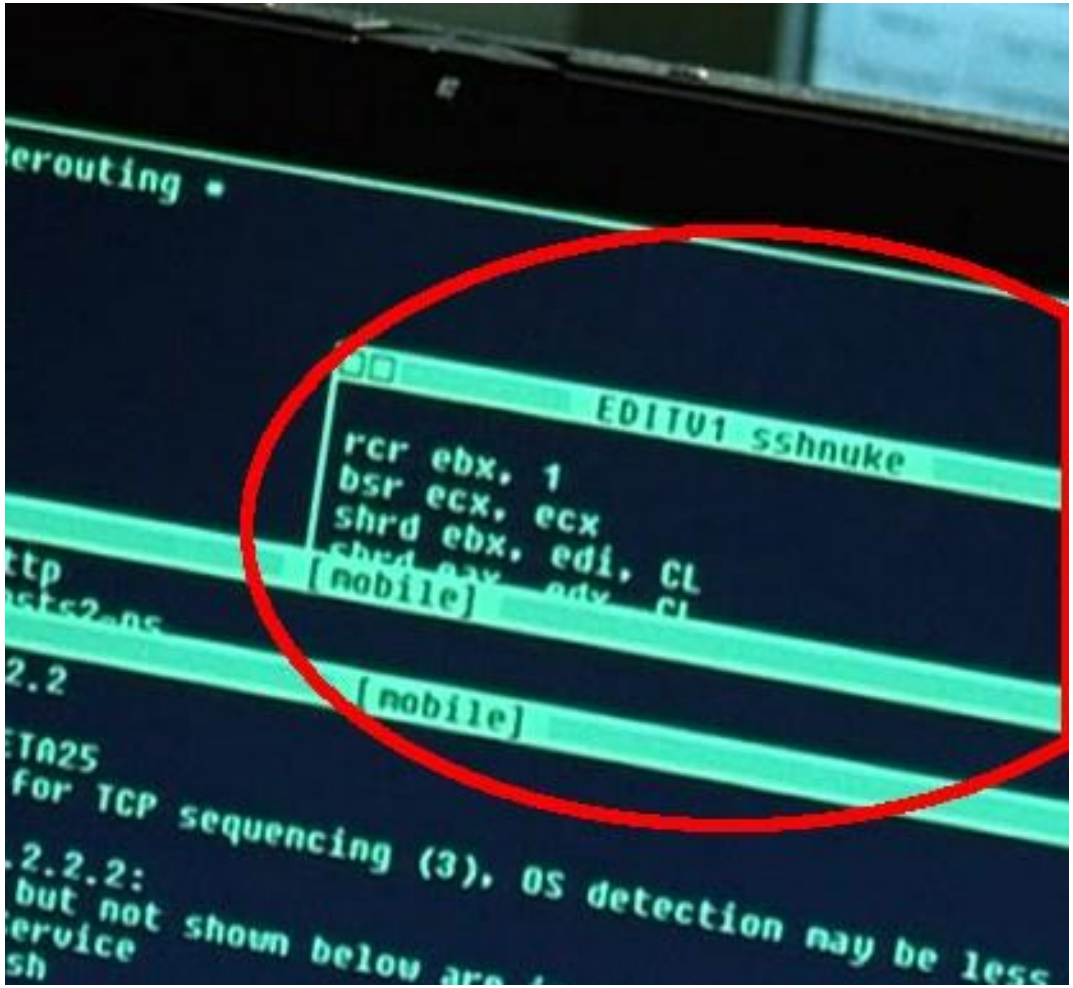
Dream in code! #goals

CS107 outcome: read x86



MATRIX RELOADED (2013)

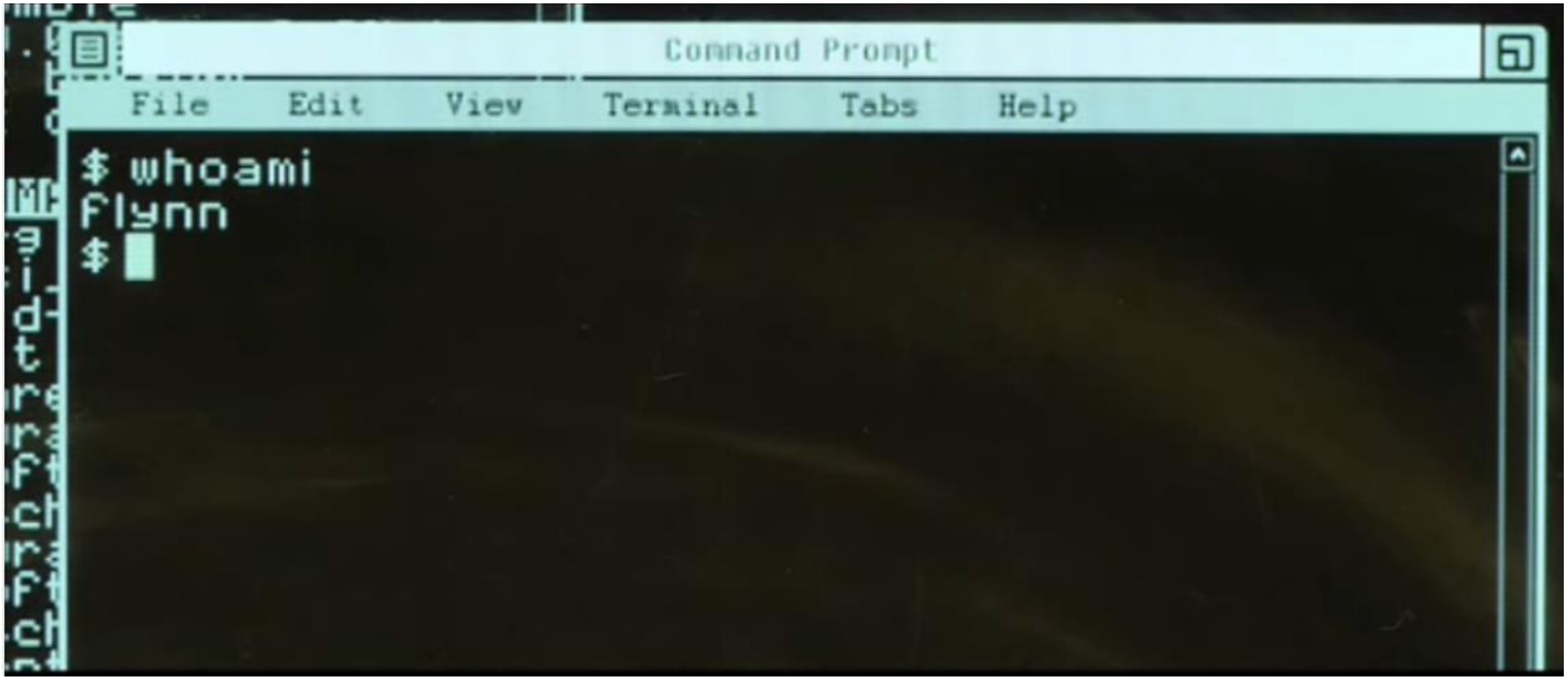
CS107 outcome: read x86



MATRIX RELOADED (2013)

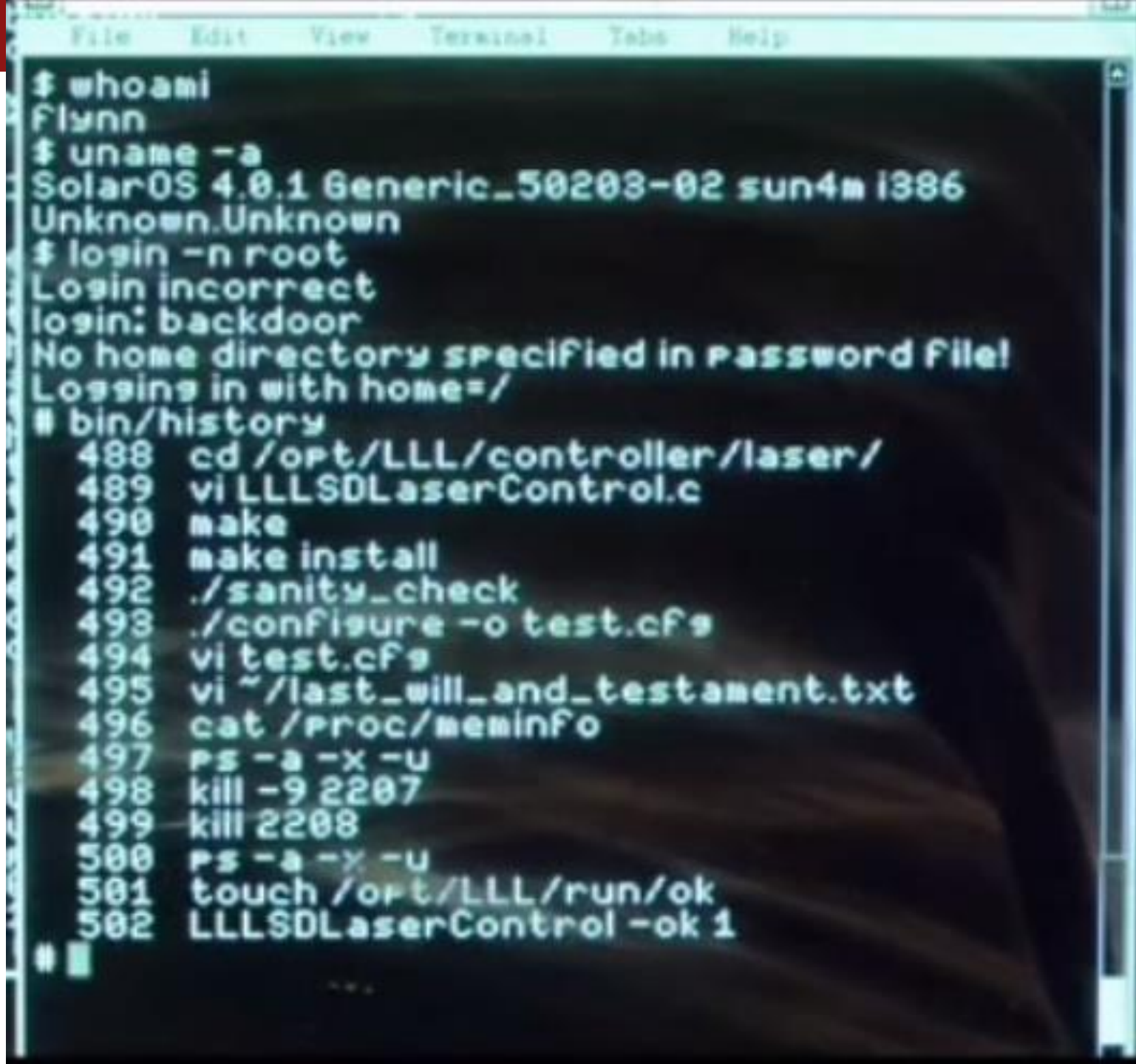
CS107 outcome: navigate on UNIX

https://www.youtube.com/watch?v=yQsaPVfze4s



TRON LEGACY (2011)

Tron Legacy (2011)



CS107 outcome: better consumer

MacBook

from $1299

- 12-inch (diagonal) LED-backlit Retina display
- 1.1GHz dual-core Intel Core m3, 1.2GHz dual-core Intel Core m5, or 1.3GHz dual-core Intel Core m7 processor
  Turbo Boost up to 3.1GHz
- Up to 10 hours battery life[1]
- Up to 512GB SSD[2]
- Force Touch trackpad
- 2.03 pounds[3]

MacBook Air 13-inch

from $999

- 13.3-inch (diagonal) LED-backlit widescreen display
- 1.6GHz dual-core Intel Core i5 or 2.2GHz dual-core Intel Core i7 processor
  Turbo Boost up to 3.2GHz
- Up to 12 hours battery life[1]
- Up to 512GB SSD[2]
- Multi-Touch trackpad
- 2.96 pounds[3]

MacBook Pro 13-inch

from $1299

- 13.3-inch (diagonal) LED-backlit Retina display
- 2.7GHz or 2.9GHz dual-core Intel Core i5 or 3.1GHz dual-core Intel Core i7 processor
  Turbo Boost up to 3.4GHz
- Up to 10 hours battery life[1]
- Up to 1TB SSD[2]
- Force Touch trackpad
- 3.48 pounds[3]

CS107 outcome: more performance-minded programmer

```
// Valgrind performance analysis

            /* Simple selection sort algorithm, O(N^2) */
            void selsort(int *arr, int n)
            {
    4,005       for (int i = 0; i < n; i++) {
    2,000           int min = i;
2,005,000           for(int j = i+1; j < n; j++)
7,003,974               if (arr[j] < arr[min]) min = j;
   13,000           swap(&arr[i], &arr[min]);
                }
            }
```

CS107 outcome: more sophisticated compiler debugging

PREPROCESSOR:

- Takes #include and #define and other preprocessor directives and replaces them with appropriate text (code.c + .h files → code.i)
  - `gcc -E code.c`

COMPILER:

- Takes processed C code and outputs appropriate AMD64 code (code.i → code.s)
  - `gcc -S code.c  # uppercase S`

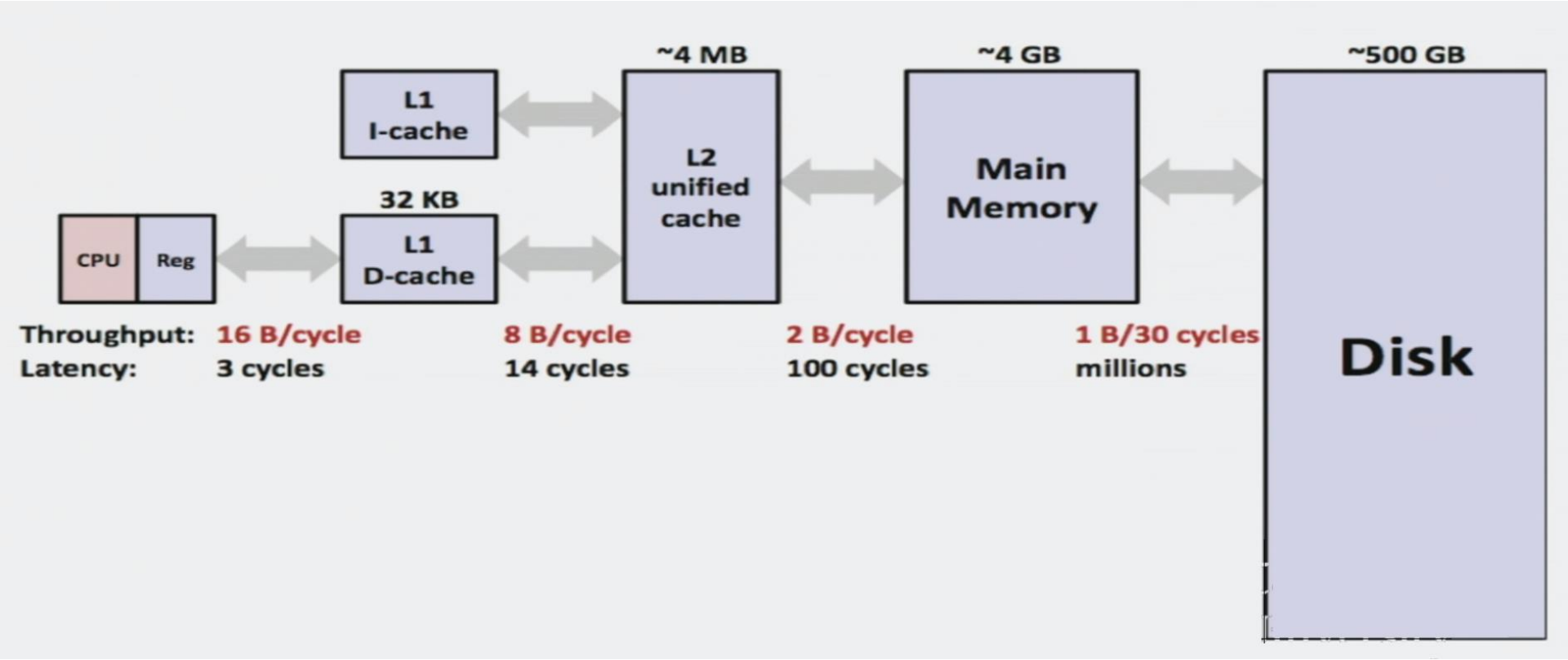
ASSEMBLER:

- Takes assembly output and makes machine output (code.s → code.o)
  - `gcc -c code.c  # lowercase c`

LINKER:

- Takes .o in question, plus other module .o files, joins them together to make the executable (code.o +.o files → a.out or code)
  - `gcc code.c -o code # lowercase o`

CS107 outcome: understanding how hardware organization impacts code performance



Quarter in Review: CS107 Experience

10 minutes of “town hall” feedback for me in the future

Discussion topic: (3 minutes)

- Do you agree with the adjectives other students are using? (first question of survey)

- What were the sources of information that gave you that impression of the course?

Discussion topic: (4 minutes)

- Go through the data, identify 2 things that stand out
- Discuss reasons why students may have voted that it was beneficial or not beneficial
- Prepare 2 points to share out to the class

Quarter in Review: Expanding on CS107

Expanding on CS107

- CS110
  - Learn about services the operating system provides you: virtual memory, file systems, networks
- CS140
  - Like heap allocator assignment, you implement operating system services you learned about in CS110
- EE108
  - Bitwise implementations of various operations in *hardware*
  - E.g., transforming “a + b” into AND, OR, XOR operations that achieve the correct sum output
- EE180
  - Looking at what the hardware does with your x86 instructions (turns out just like gcc will rearrange your C code as it sees fit, hardware will often reorder execution of instructions—all constrained to give the same end result)
  - More on cache and other hardware components we touched in in 107
- CS143
  - Learn how gcc works its syntax-checking, optimization, and assembly code generation magic