Course Overview

Computer Systems Organization (Fall 2016)

Instructor:

Jinyang Li

Staff

- Lecturer:
  - Jinyang Li jinyang@cs.nyu.edu
- Recitation instructor:
  - Hassan Mujtaba Zaidi hmz224@nyu.edu
- TAs:
  - Subhankar Ghosh subhankar.ghosh@nyu.edu
  - Will Zhou will.zhou@nyu.edu

Computer Systems Organization

Not that kind of organization

This class adds to your CV…

- C programming
- UNIX
- X86 assembly

Not what the class is about either

What this class is about

- Those details that set hackers apart from novice programmers
  - How your program runs on the hardware
  - Why it fails
  - Why it is slow
- Modern computer systems are shrouded in layers of abstraction

What is an abstraction (in computer systems)?

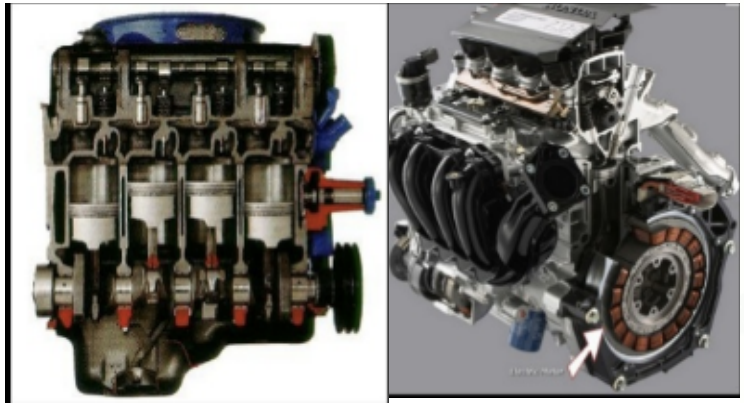
- A technique for managing the complexities in engineered systems.
  - Establish an "abstract" interface for interacting with a system/module
  - Hide details behind the interface

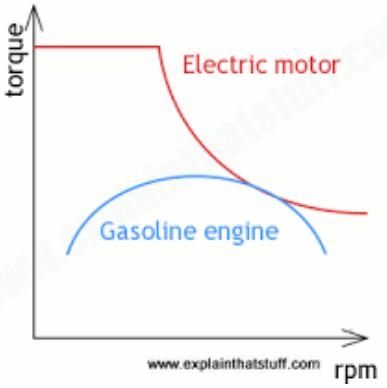
Abstractions: the car example

user-level

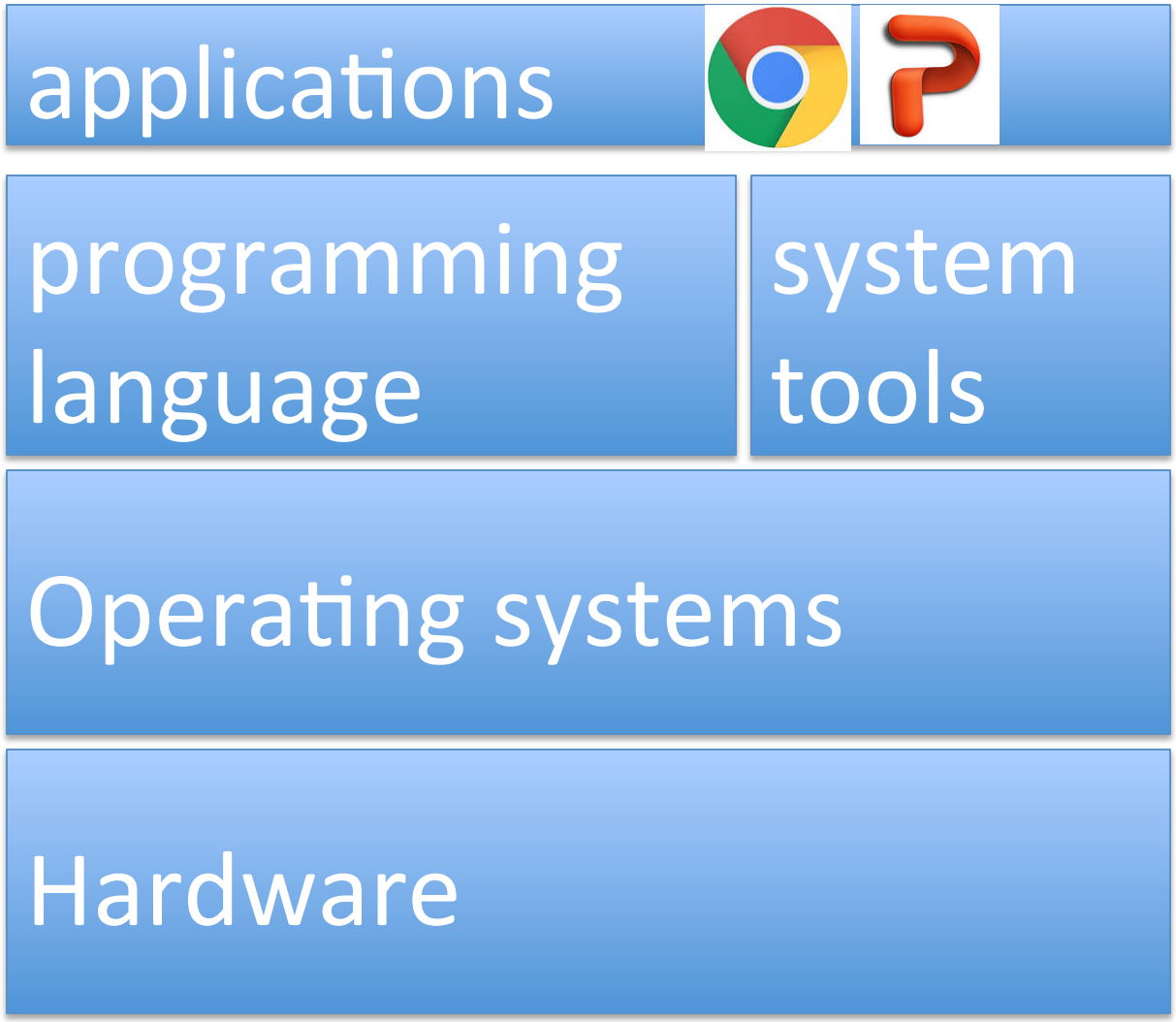
mechanic-level

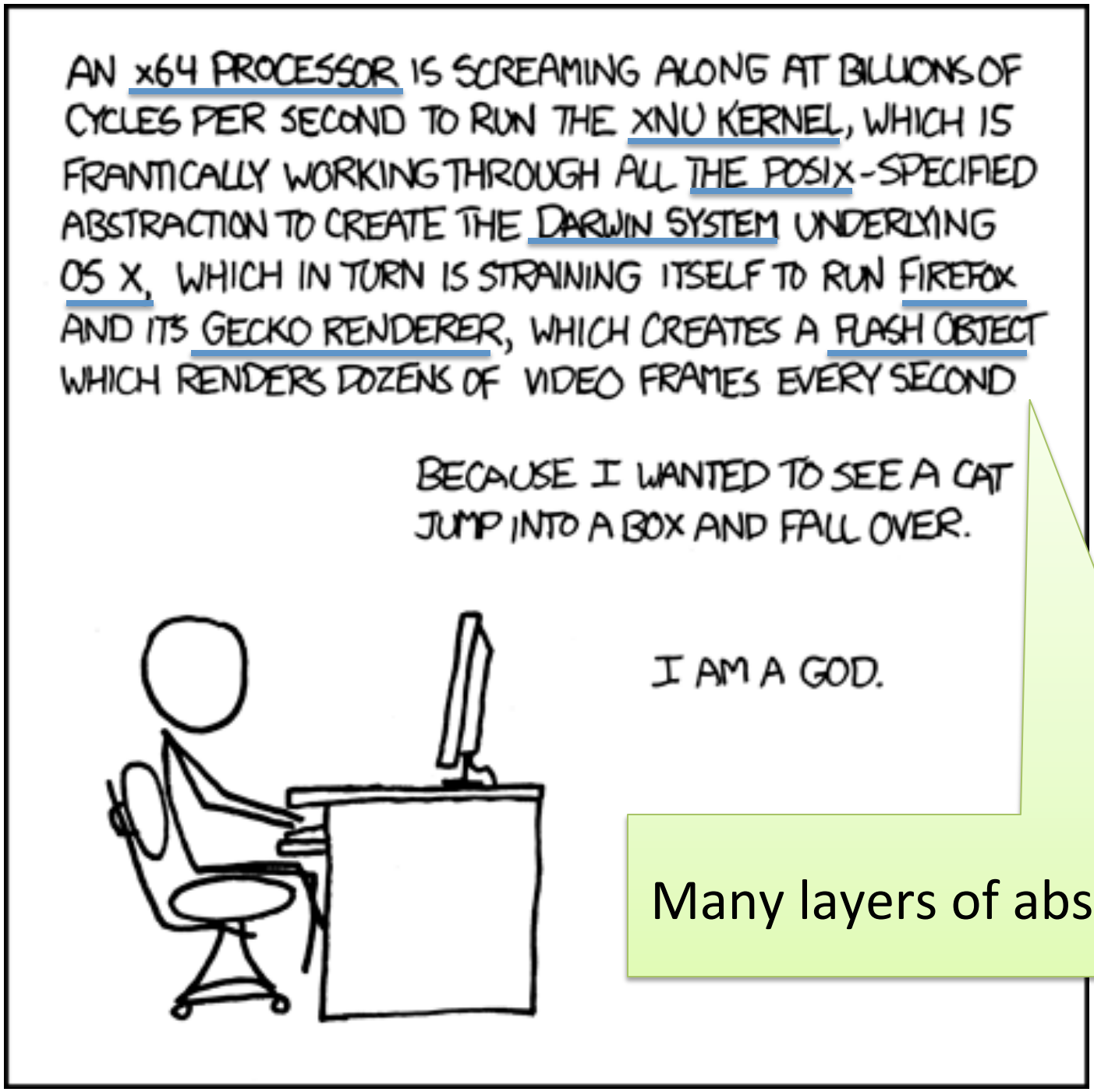
designer/builder-level





Abstractions: computer systems





Many layers of abstraction

Course Theme:
Abstraction Is Good But Don't Forget Reality

- Most CS classes stay within a single layer of abstraction
- This class:
  - Help you peek ``under-the-hood'' in many layers
- Goal:
  - Make you more effective programmers
    - Debug problems
    - Tune performance
  - Prepare you for later "systems" classes in CS
    - Compilers, Operating Systems, Networks, Computer Architecture, Distributed Systems

Reality #1:
Ints are not Integers, Floats are not Reals

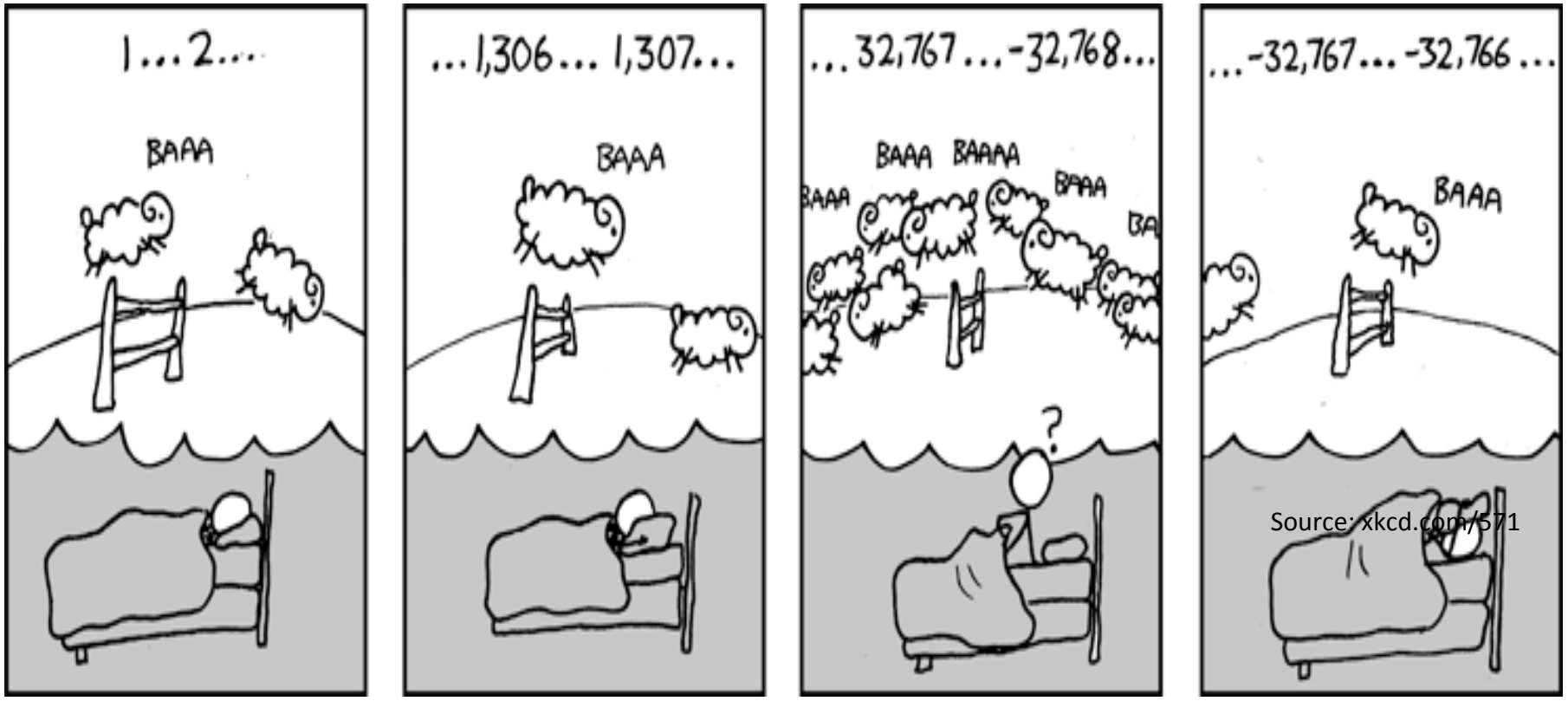
- $x^2 \geq 0$?
- $(x + y) + z = x + (y + z)$?

```
Public class Test {

      public static void main(String[] args) {
                int x = Integer.parseInt(args[0]);
                Systems.out.println(x * x);
          }
}
```

```
java Test 12 ➞    144
java Test 123456  ➞     ???
```

Reality #1:
Ints are not Integers, Floats are not Reals



Reality #2: You've Got to Know Assembly

- No need to program in assembly
- Knowledge of assembly helps one understand machine-level execution
  - Debugging
  - Performance tuning
  - Writing system software (e.g. compilers , OS)
  - Creating / fighting malware
    - x86 assembly is the language of choice!

Reality #3: Memory Matters

- Memory is not unbounded
  - It must be allocated and managed
- Memory referencing bugs are esp. wicked
- Memories of diff applications are isolated

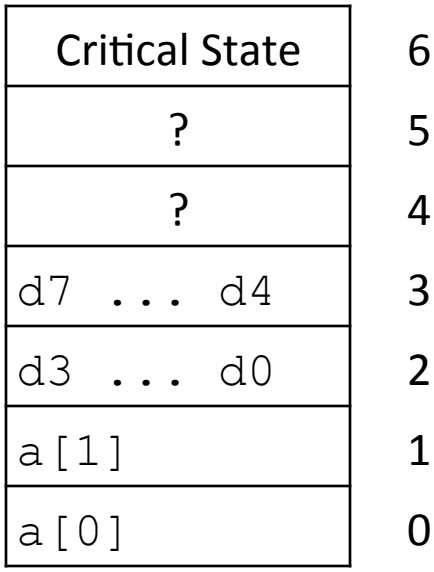
Memory Referencing Errors

- C/C++ let programmers make memory errors
  - Out of bounds array references
  - Invalid pointer values
  - Double free, use after free
- Errors can lead to nasty bugs
  - Corrupt program objects
  - Effect of bug observed long after the corruption

Memory Referencing Bug Example

```
double fun(int i)
{
  double d[1] = {3.14}; /* allocate an array of 1 double*/
  int a[2];   /* allocate an array of 2 integers */
  a[i] = 1073741824; /* Possibly out of bounds */
  return d[0];
}
```

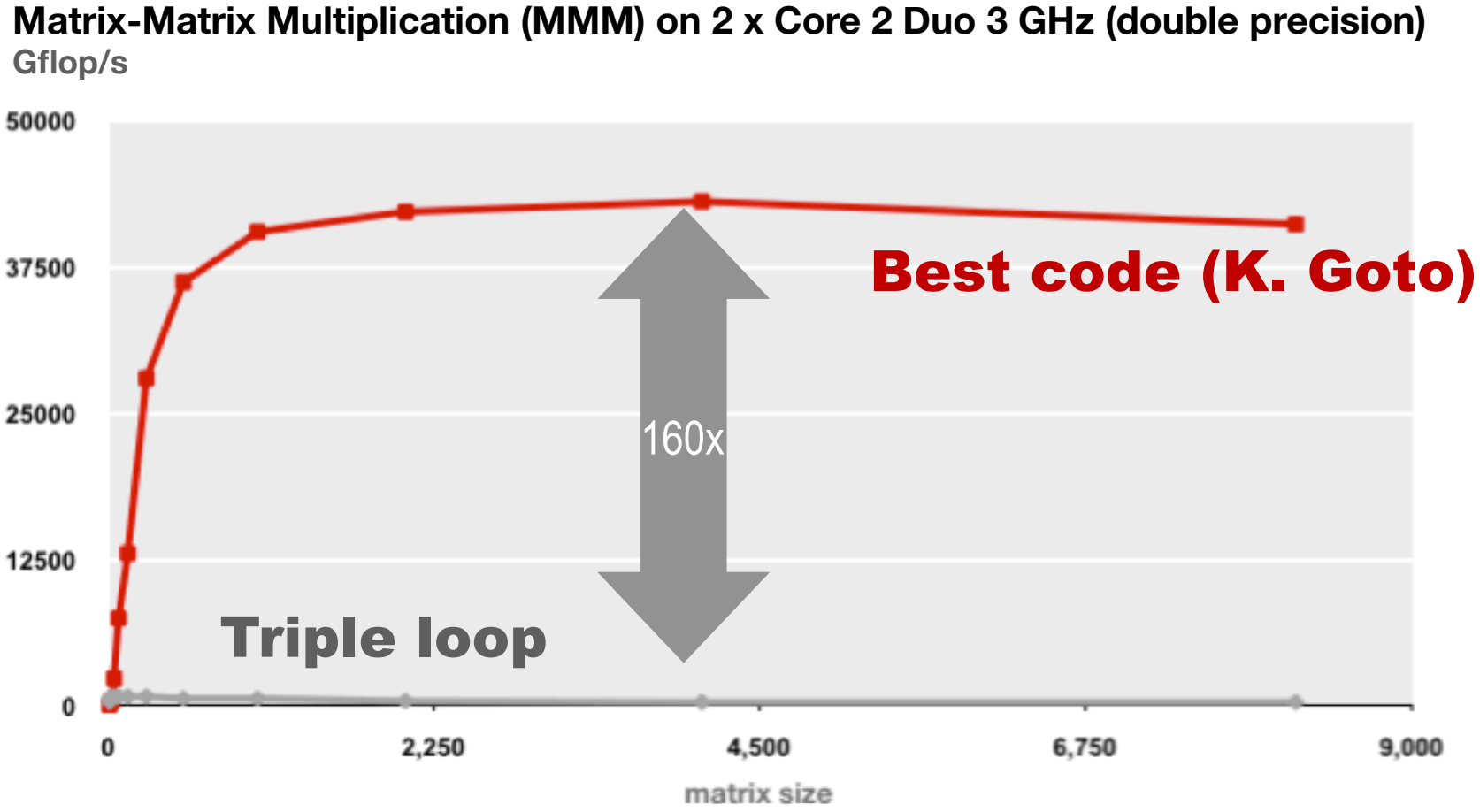
```
fun(0)  →  3.14
fun(1)  →  3.14
fun(2)  →  3.1399998664856
fun(3)  →  2.0000000610351 56
fun(4)  →  3.14
fun(6)  →  Segmentation fault
```



Reality #4: Asymptotic performance is not always sufficient

- Constant factors matter
- In order to optimize performance:
  - How programs compiled and executed
  - How to measure performance and identify bottlenecks
  - How to speedup using multiple CPU cores

Example Matrix Multiplication



- Both implementations have exactly the same operations count ($2n^3$)
- Reason for 20x: Multi-threading, blocking, loop unrolling, array scalarization

Course Perspective

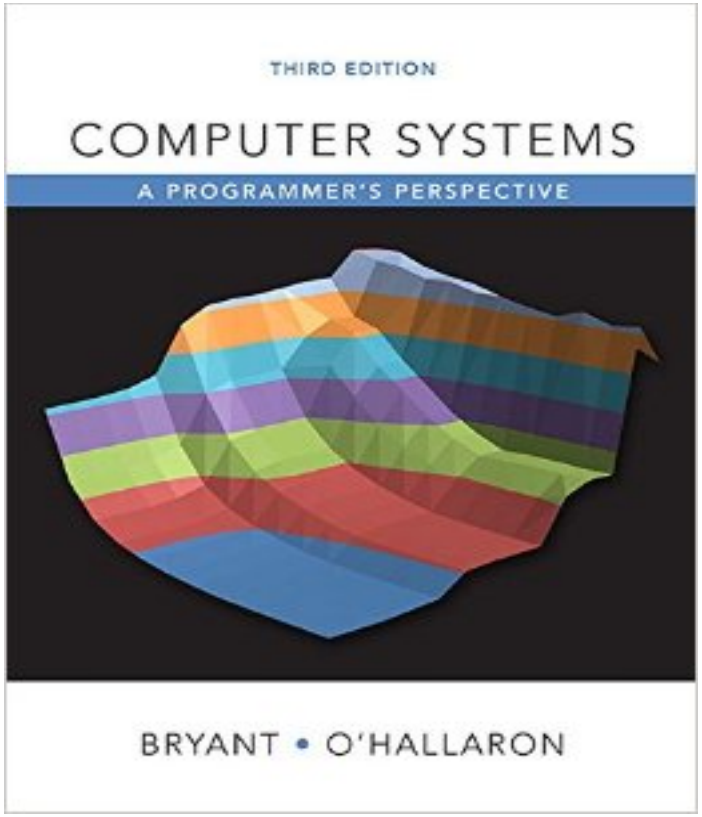
- Most Systems Courses are Builder-Centric
  - Computer Architecture
    - Design pipelined processor in Verilog
  - Operating Systems
    - Implement large portions of operating system
  - Compilers
    - Write compiler for simple language
  - Networking
    - Implement and simulate network protocols

Course Perspective (Cont.)

- This course is programmer-centric
  - Understanding of underlying system makes a more effective programmer
  - Bring out the hidden hacker in everyone

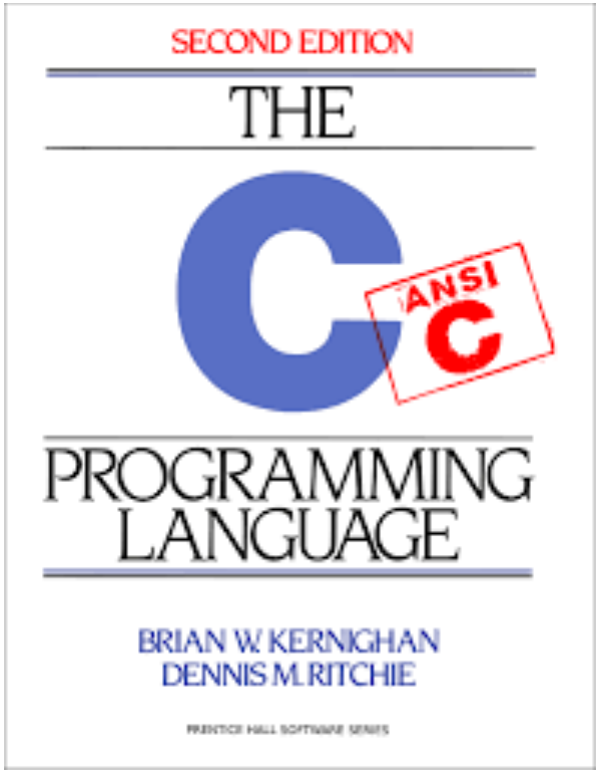
Textbooks

- Randal E. Bryant and David R. O'Hallaron,
  - "Computer Systems: A Programmer's Perspective, 3nd Edition", http://csapp.cs.cmu.edu



Textbooks

- Brian Kernighan and Dennis Ritchie,
  - "The C Programming Language, 2nd Edition", Prentice Hall, 1988, On reserve at NYU library



Course Components

- Lectures (M/W 3:30-4:45pm)
- Recitation (W 8-9:15am)
  - In-class exercises provide hands-on instruction
- 5 Programming labs
  - 2-3 weeks each
- Mid-term
- Final exam

Grade breakdown

- Participation (5%)
  - Q&A during lecture/recitation
  - helpfulness on class discussion board
- Labs (35%)
- Midterm (25%)
- Final (35%)

Course Syllabus

- Basic C
  - L1 (CLab), L2 (Rabin-Karp Lab)
- Assembly: Representation of program and data
  - L3 (Binarylab)
- Virtual Memory: address translation, allocation
  - L4 (Malloclab)
- Concurrent programming
  - L5 (Threadlab)

Getting Help

- Class Web Page: **http://www.news.cs.nyu.edu/~jinyang/fa16-cso**
  - Class schedule (subject to change...)
  - Lectures notes, assignments

- Piazza is our message board

Lab Policies

- You must work alone on all assignments
  - You may post questions on Piazza,
  - You are encouraged to answer others' questions, but refrain from explicitly giving away solutions.
- Hand-ins
  - Assignments due at 11:59pm on the due date
  - Everybody has 5 grace days
  - Zero score if a lab is handed in >2 days late

Integrity and Collaboration Policy

We will enforce the policy strictly.

1. The work that you turn in must be yours
2. You must acknowledge your influences
3. You must not look at, or use, solutions from prior years or the Web, or seek assistance from the Internet
4. You must take reasonable steps to protect your work
   - You must not publish your solutions
5. If there are inexplicable discrepancies between exam and lab performance, we will over-weight the exam and interview you.

Integrity and Collaboration Policy

- Academic integrity is very important.
  - Fairness
  - If you don't do the work, you won't learn anything

Integrity and Collaboration Policy

- We will enforce this policy strictly and report violators to the department and Dean.

- If you cannot complete an assignment, don't turn it in: one or two uncompleted assignments won't result in F.