COMP 250

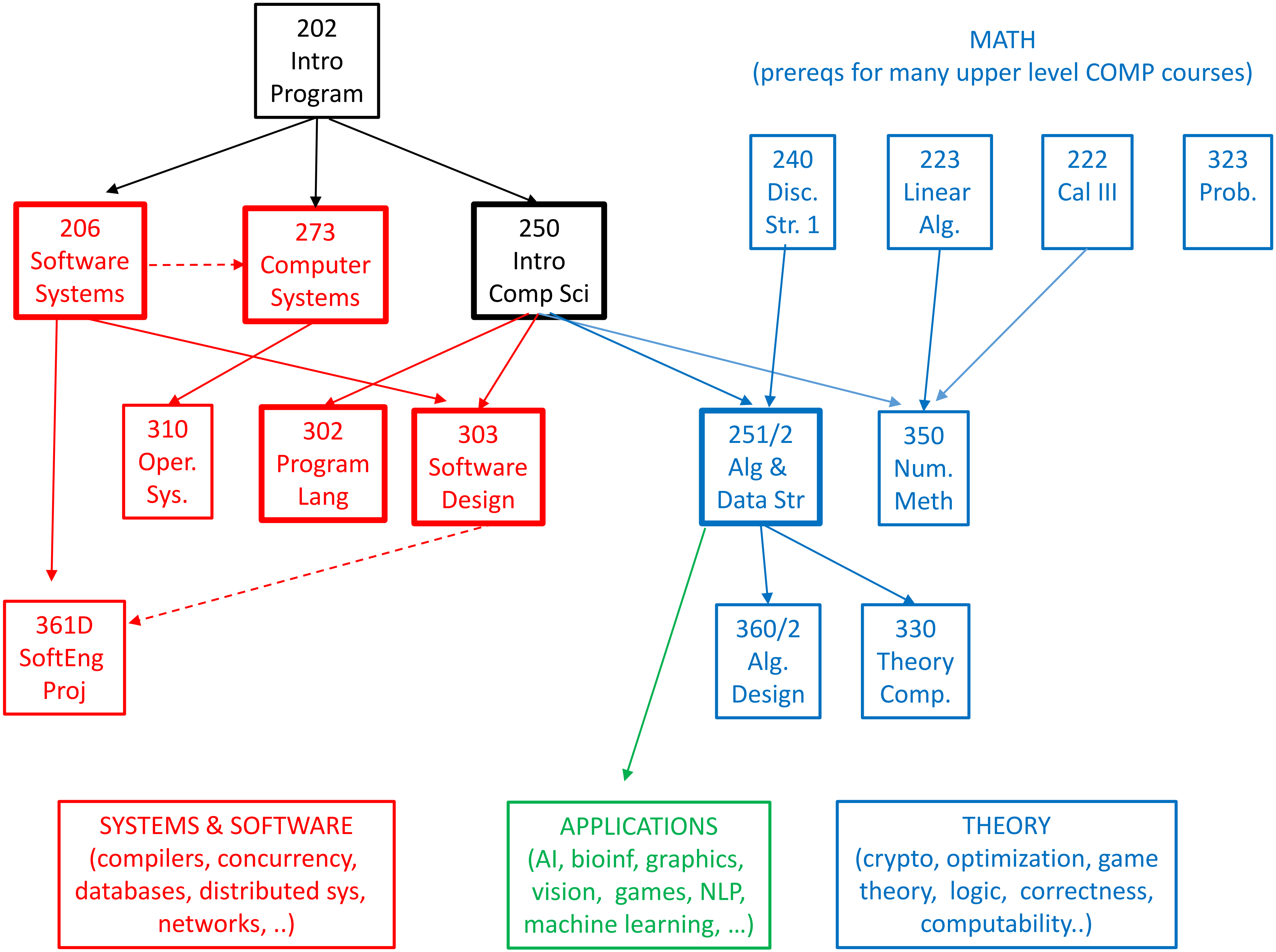
Lecture 38

MISC

- beyond COMP 250

- final exam comments

Dec 6/7, 2017



202 Intro Program

206 Software Systems

273 Computer Systems

250 Intro Comp Sci

MATH
(prereqs for many upper level COMP courses)

240 Disc. Str. 1

223 Linear Alg.

222 Cal III

323 Prob.

310 Oper. Sys.

302 Program Lang

303 Software Design

251/2 Alg & Data Str

350 Num. Meth

361D SoftEng Proj

360/2 Alg. Design

330 Theory Comp.

SYSTEMS & SOFTWARE
(compilers, concurrency, databases, distributed sys, networks, ..)

APPLICATIONS
(AI, bioinf, graphics, vision, games, NLP, machine learning, …)

THEORY
(crypto, optimization, game theory, logic, correctness, computability..)

“Systems”

- Multiple interacting parts that create a complex whole

(e.g. 273: CPU, GPU, cache, RAM, disk, …)

- Multiple levels

(e.g. 273: circuits to high level languages)

COMP 206 Intro to Software Systems

Dave Meger will teach it in Winter 2018.

Content will be similar to the Fall 2017 version

http://www.cs.mcgill.ca/~jvybihal/Courses/CS206/COMP206CourseOutline7.pdf

> "Comprehensive overview of programming in C, use of system calls and libraries, [UNIX, bash scripting], debugging and testing of code; use of development tools like make and version control systems."

Dave will use github. (New pilot project.)

COMP 273 Intro to Computer Systems

Joseph Vybihal will teach it in Winter 2018.

http://www.cs.mcgill.ca/~jvybihal/Courses/CS273/index.html

My lecture notes + exercises: http://www.cim.mcgill.ca/~langer/273.html

Topics:

Logic circuits (simulator only, i.e. no hardware)
CPU design
Assembly language (MIPS) (call stack, recursion)
Memory hierarchies (cache, *virtual memory,* disk)
User vs kernel modes (OS versus application)

COMP 302 Programming Lang. & Paradigms

Clark Verbrugge will teach it in Winter 2018.

http://www.sable.mcgill.ca/~clump/

The language will be Scala.

https://www.scala-lang.org/

From his course web page in Fall 2016: *(my italics added)*

Theoretical and practical aspects of language design and programming practice will be examined in a *functional programming* context. Topics additionally include basics of programming language design and construction, *binding and scoping, parameter passing, lambda abstraction, data abstraction, and type checking*.

COMP 303 Software Design

In Winter 2018,   it will be taught by Joseph Vybihal.

In COMP 250, the assignments use 1-5 classes.

How to manage a project that has 50 classes?   Or 300 classes?

Prereqs are COMP 250+206.

Good to take COMP 302 before 303, if you can…   BUT….. also

good to take 303 before COMP 361D Software Engineering Project (2018-2019), if you can.

COMP 251 Algorithms & Data Structures

In Winter 2018, it will be taught by Adrian Vetta.

http://www.math.mcgill.ca/vetta/CS251.dir/

For exercises and links to MOOC videos, see my web page:

http://www.cim.mcgill.ca/~langer/251.html

As you know by now… MATH 240 or MATH 235 are co-requisites, but they should be considered prerequisites.

COMP 252 (Honours version of 251)

In Winter 2018, it will be taught by Luc Devroye.

http://luc.devroye.org/252.html

Same story with MATH 240 or MATH 235.

Luc will cover much more material than in COMP 251.

To prepare, see MOOC links on previous slide. Happy holidays!

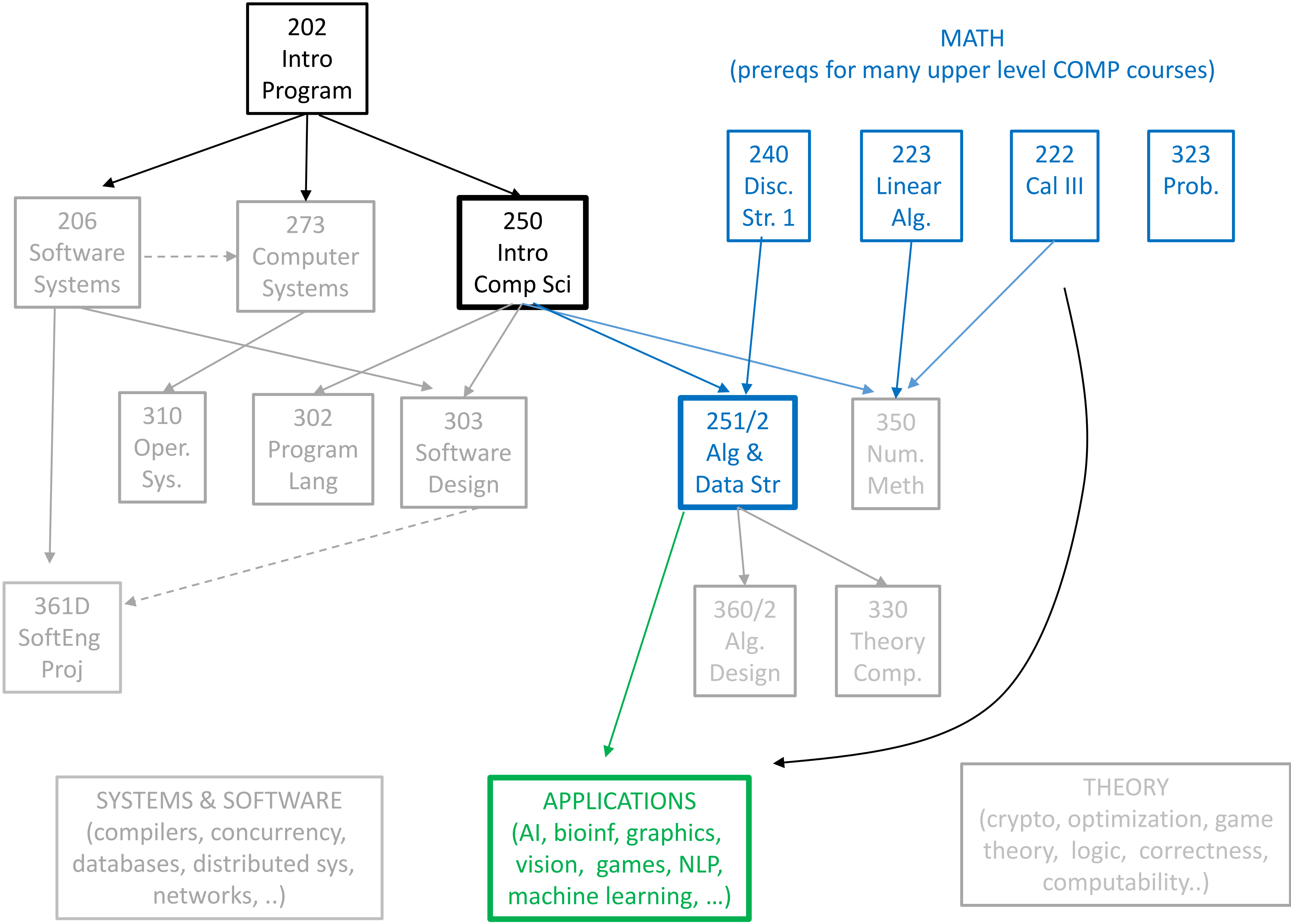
COMP 350 Numerical Computing

It will be offered in Fall 2018.

Computer representation of numbers, IEEE Standard for Floating Point Representation, computer arithmetic and rounding errors. Numerical stability. Matrix computations and software systems. Polynomial interpolation. Least-squares approximation. Iterative methods for solving a nonlinear equation. Discretization methods for integration and differential equations.

Useful for anything that involves “number crunching” or scientific computing   e.g.   Machine learning,   computer vision, …

Prereq: MATH 222 (Cal 3),   MATH 223 (Linear Algebra)



202 Intro Program

MATH
(prereqs for many upper level COMP courses)

240 Disc. Str. 1

223 Linear Alg.

222 Cal III

323 Prob.

206 Software Systems

273 Computer Systems

250 Intro Comp Sci

310 Oper. Sys.

302 Program Lang

303 Software Design

251/2 Alg & Data Str

350 Num. Meth

361D SoftEng Proj

360/2 Alg. Design

330 Theory Comp.

SYSTEMS & SOFTWARE
(compilers, concurrency, databases, distributed sys, networks, ..)

APPLICATIONS
(AI, bioinf, graphics, vision, games, NLP, machine learning, …)

THEORY
(crypto, optimization, game theory, logic, correctness, computability..)

Final Exam

Multiple Choice with 50 questions.

Four choices on each question.

I am treating a blank answer as incorrect.

So, when you are unsure of the answer, *take your best guess* ($prob_{guess\ correct}$ *>= 0.25).*

To account for correct answers from guessing, I will give small penalty for incorrect answers.

Q: Standard multiple choice exams such as SAT, LSAT, GRE, GMAT, etc, *don't* penalize for incorrect answers. So why I am doing this ?

A: The absolute scores in those exams don't matter. *Only the student ranking (percentile) matters.*

Grading policy

Your grade out of 50

$$= \max(\, 0, \;\; -10 \; + \; \frac{6}{5} * \text{raw number correct} \,)$$

This formula is mathematically equivalent to

$$\max(\, 0, \; \text{number correct} - \frac{1}{5} \; \text{number incorrect})$$

i.e. I am applying a penalty of $\frac{1}{5}$ for each incorrect (or blank) answer.

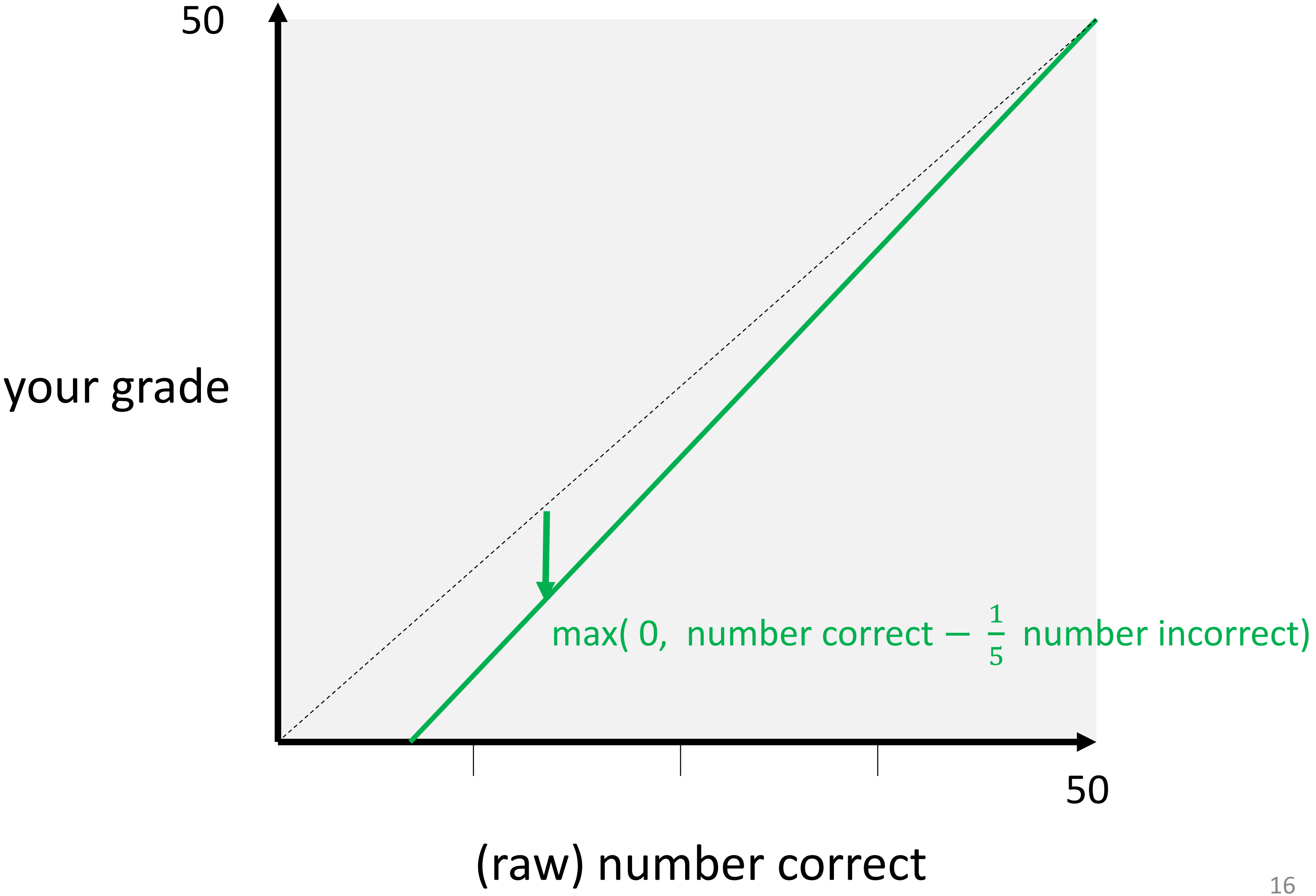
If you don’t know how to answer a question, then on average you will still benefit from guessing.

Here is why: $\frac{1}{4}$ of the time you will be correct and get 1 point.
$\frac{3}{4}$ of the time you will be incorrect and lose $\frac{1}{5}$ of a point.

On average, uniform random guessing will give you $\frac{1}{10}$ point per question, since

$$\frac{1}{4} * 1 + \frac{3}{4} * (-\frac{1}{5}) = \frac{1}{10}.$$

If you were to guess randomly on every question, then *on average* you would get 12.5/50 correct. This would give you a grade of 5/50. Thus, the penalty only partially accounts for guessing.



Final Exam review

Lectures 1-9 (11 / 50 questions)

Preliminaries

- binary numbers (and other bases)
- basic math: logs and summations
- grade school algorithms (A1)

Lists, Stacks, Queues

- data structures
- algorithms: operations e.g. add & remove, sorting
- big O (informal)

Possible Questions

- Convert a given number from one base to another
- Simplify a log or summation expreession
- What is O( ) of a some operation on some data structure?
- Given a stack or queue, what is the result of applying certain operations?
- Given a list, how does it change after some step of some sorting algorithm?

Lectures 10-18 (10 / 50 questions)

Proofs by Induction

Recursion (many algorithm examples)

Recurrences using back substitution

Asymptotic complexity: formal definitions

Possible Questions

- Induction proofs - identify steps

- What is base case of some recursive algorithm ? What is the call stack when some condition is met ?

- Solve a recurrence

- Apply the limit rules and formal definitions of big O/etc

Lectures 19-29 (20 / 50 questions)

Trees, BSTs, Heaps, Maps, Graphs

- data structures
- algorithms: add & remove, traverse, …

Possible Questions

- What is order of vertices visited in some given tree or graph, using some given traversal method ?

- Manipulate a binary search tree or heap by…

- Hash tables: how do they work? What are they good for?

Lectures 30-34, 36 (9 / 50 questions)

Object Oriented Design

- Inheritance and polymorphism
- Object methods (equals, hashCode, toString(), …)
- Modifiers

My Office Hours

- I am out of town Dec. 9-17.

- I will be at McGill some of Dec. 18-21.

  I will announce office hours.

- Final Exam is Dec 21.

- I hope to receive exam grades before holidays but I can’t be sure.