

Welcome to CSCI 235

Tiziana Ligorio

tligorio@hunter.cuny.edu

Today's Plan

Welcome

Logistics (Rules of the game)

What is CSCI 235?

Why Software Engineering?



People

Instructor: Prof. Tiziana Ligorio

Undergraduate Teaching Assistants (Lab 1001B HN):

Owen Kunhardt

Ferdi Lesporis

Acknowledgments

This course was designed with input from many great resources other than the required textbook

Many thanks for materials and inspiration to

Simon Ayzman

Susan Epstein

Keith Schwarz

Ioannis Stamos

Stewart Weiss

Logistics

(The rules of the game)

Your first assignment.
MUST READ!!!

Course Webpage / Syllabus

Programming Rules / Programming Projects

Gradescope

Linux Accounts

Communication and Help

Course Webpage

https://tligorio.github.io/CSCI235_Summer2019.html

Visit regularly for:

- Tentative Schedule and changes

- Lecture Notes

- Programming Projects

- Study questions

Syllabus


MUST READ!!!

Textbook (optional with alternatives)

No Makeups

~ 1 Project/week - No Late Submissions

NOTE EXAM QUESTIONS WILL BE DIRECTLY BASED ON THE PROGRAMMING PROJECTS

Component	Per Item %	Total %
Lecture Activity 		5%
Programming Projects	No extra-credit projects	35%
Exams	Midterm Exam 20%	60%
	Final Exam 40%	
If you fail or miss the midterm exam, the final exam will replace it		

Programming Projects

Five programming projects

No extra-credit project

All projects submitted on **Gradescope**

If you haven't done so already, login to **Gradescope ASAP**

MUST USE YOUR CORRECT EMPL ID

MUST READ: [Programming Rules](#) document on course webpage

Projects due on Thursdays 6pm: come to class and ask questions

https://tligorio.github.io/gradescope_help.html

Gradescope

To be used for submission of ALL programming projects

Check your email and follow invitation instructions

Gradescope

If you haven't received an invitation email:

Course Entry Code: **97X5DX**

If you DON'T already have an account:

1. Go to www.gradescope.com
2. Sign Up as a Student
3. Enter Course Entry Code
4. Enter your information

If you DO already have an account:

1. Go to www.gradescope.com and log in
2. At the bottom right click on Enroll in Course
3. Enter Course Entry Code

Gadescop is not for
Debugging

Linux Accounts

MUST HAVE!!!

Go home, try to remotely login and if you can't see tutors in lab tomorrow!!!

Follow instructions in:

Programming Rules document on course web page
and

[http://www.geography.hunter.cuny.edu/tbw/
CS.Linux.Lab.FAQ/departments_of_computer_science.faq.htm](http://www.geography.hunter.cuny.edu/tbw/CS.Linux.Lab.FAQ/departments_of_computer_science.faq.htm)

Project Grading

Unless otherwise specified:

80% correctness

- a submission that does not compile will receive 0 points
- incremental development
- version control -> submit latest working version for partial credit

10% documentation

10% style and design

All occurrences of plagiarism will be reported to the office of student affairs with no exception.

Communication and Help

Let us hear from you!

If you find a typo or mistake let me know!!!

Blackboard forum

If you don't understand something ask!!!

In class or Blackboard forum

If you have concerns on something other than course content come talk to me

Office hours: **by appointment**

Questions and Forum Etiquette

Different prior exposure to the material

All questions are good questions!!!

Friendly and collegial environment - we are here to help

UTAs in Lab

Lab 1001B

Drop-in Tutoring

Mo 12:00 - 1:00 , 3:00 - 5:00

Tu 12:00 - 1:00 , 3:00 - 5:00

We 11:00 - 1:00 , 3:00 - 4:00

Th 11:00 - 1:00 , 3:00 - 4:00

Introducing Lecture Activity

5% of final grade

Attendance but must show an adequate attempt

Work out a new problem 

What is CSCI 235?

Programming => Software Analysis and Design
Expected professional and responsible conduct

Think like a Computer Scientist:

Design and maintain complex programs

Software Engineering, Abstraction, OOP

Design and represent data and its management

Abstract Data Types

Implement data representation and operations

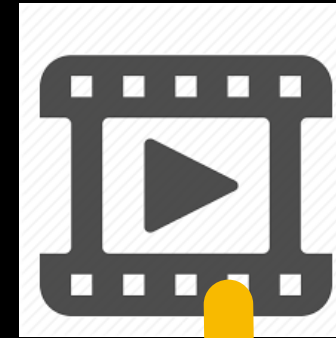
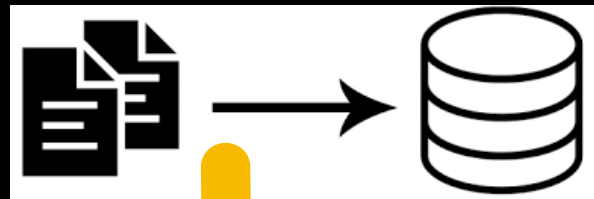
Data Structures

Algorithms

Understand Algorithm Efficiency



“It’s all just bits and bytes...”



STRUCTURE

101010101010101010
010101010101010101
10011001100110011001

Increasing software complexity

Society keeps digitizing more aspects of life

Software keeps getting bigger

Complexity of software systems ever increasing

Exciting!!!

Daunting for software engineers



What is software complexity?

Lines of code?

Not an exact measure but can be revealing

~1 - 10	Hello world
~100	Most STL queue implementations
~1,000	Typical Computer Science curriculum term project
~10,000	Intensive team project
~100,000	Most Linux command line utilities
~1,000,000	Linux g++ compiler
~10,000,000	Mozilla Firefox
~50,000,000	Microsoft's Windows
~2,000,000,000	Google (search, maps, docs, gmail, ...)

Problems of software complexity

Every bit counts!

A single incorrect bit may result in:

- negative instead of positive int
- pointer past the end of an array
- unsorted rather than sorted vector
- ...

Program performs unexpectedly

Problems of software complexity

Every bit counts!

A single incorrect bit may result in:

- negative instead of positive int
- pointer past the end of an array
- unsorted rather than sorted vector
- ...

Program performs unexpectedly



Problems of software complexity

Two lines of code interact if they manipulate same data

```
int x = 5;    // if I change the x to my_var  
cout << x;    // I must change it here too
```

Assume n lines of code

Any line may interact with any number of other lines

$n(n-1)/2 = (n^2-n)/2$ possible interactions

With 10 lines of code there may be
45 interactions

Unlikely but it gives
you an idea of how
bad it can get

Problems of software complexity

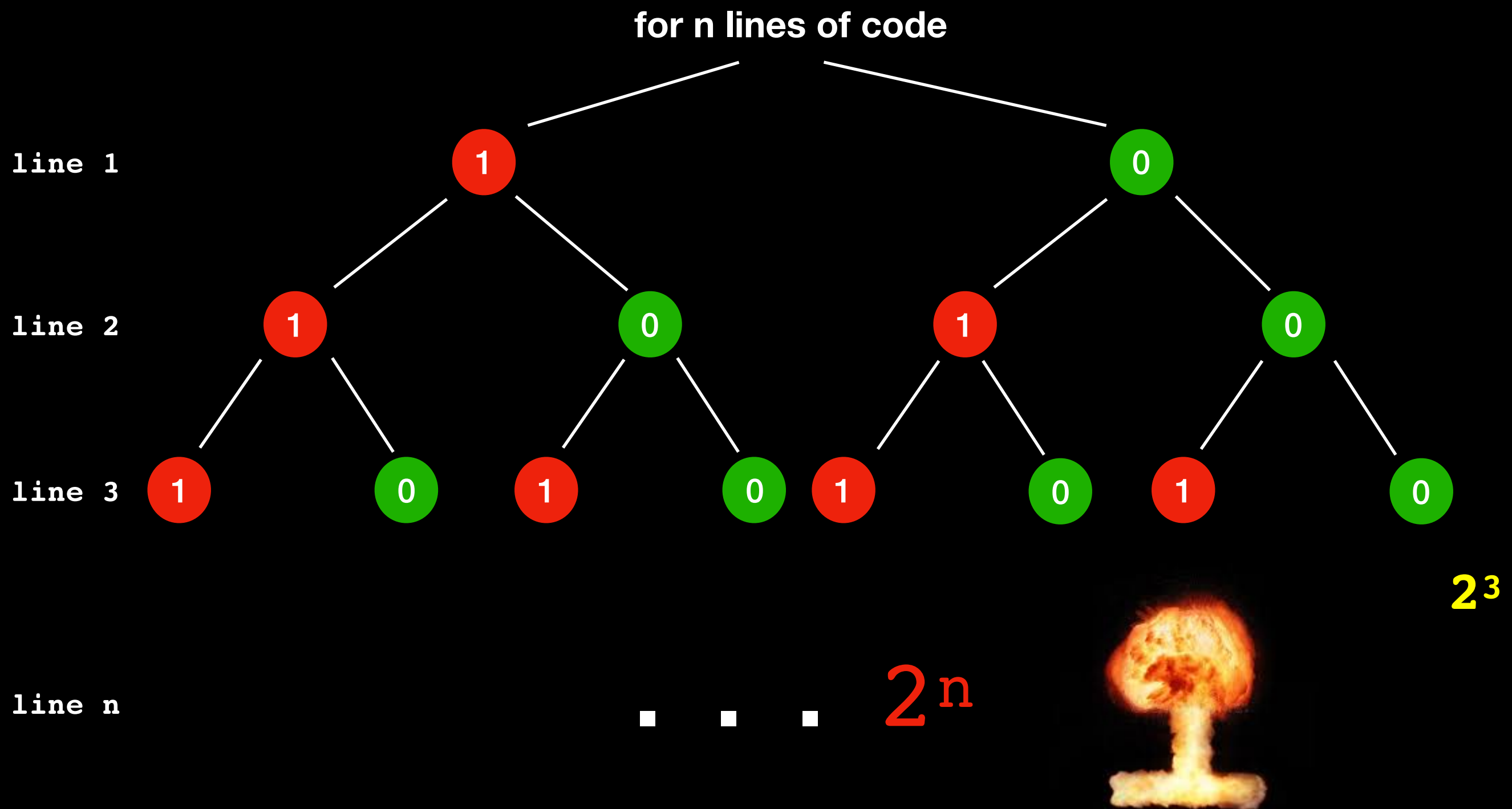
Assume line 1 shares same data with lines 6 and 23.

So in actuality the three lines all together form an interaction

If we think of subsets of lines of code interacting (sharing the same data) . . .

How many possible subsets?

Again unlikely all possible subsets will interact but it gives you an idea of why you'd want to control it



Every path down the tree is an interaction among one possible subset of lines of code

Lecture Activity

Draw a **very small** square on the leftmost bottom corner

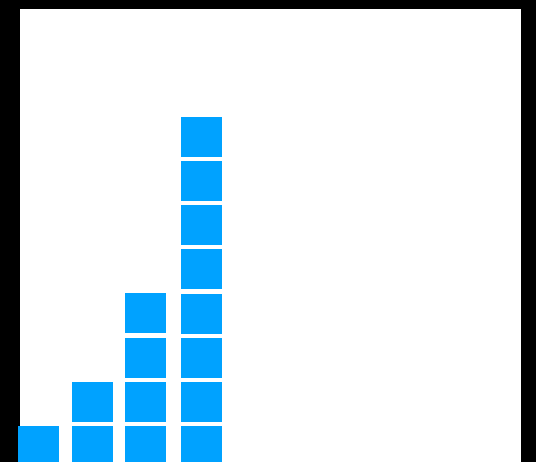
Next to it, double it (2 squares one on top of other)

Next to it, double it (4 squares one on top of other)

Next to it double it , ...

Keep going...

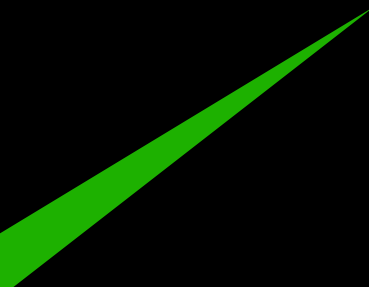
How quickly do you run off the top of the page?



Problems of software complexity

How folding paper can get you to the moon:

<https://www.youtube.com/watch?v=AmFMJJC45f1Q>



Watch this home if we
don't have time at the
end of lecture

Problems of software complexity

How do you go about modifying code with many interactions?

Larger software has greater likelihood of error

More difficult to debug and modify

Minimize complexity!!!

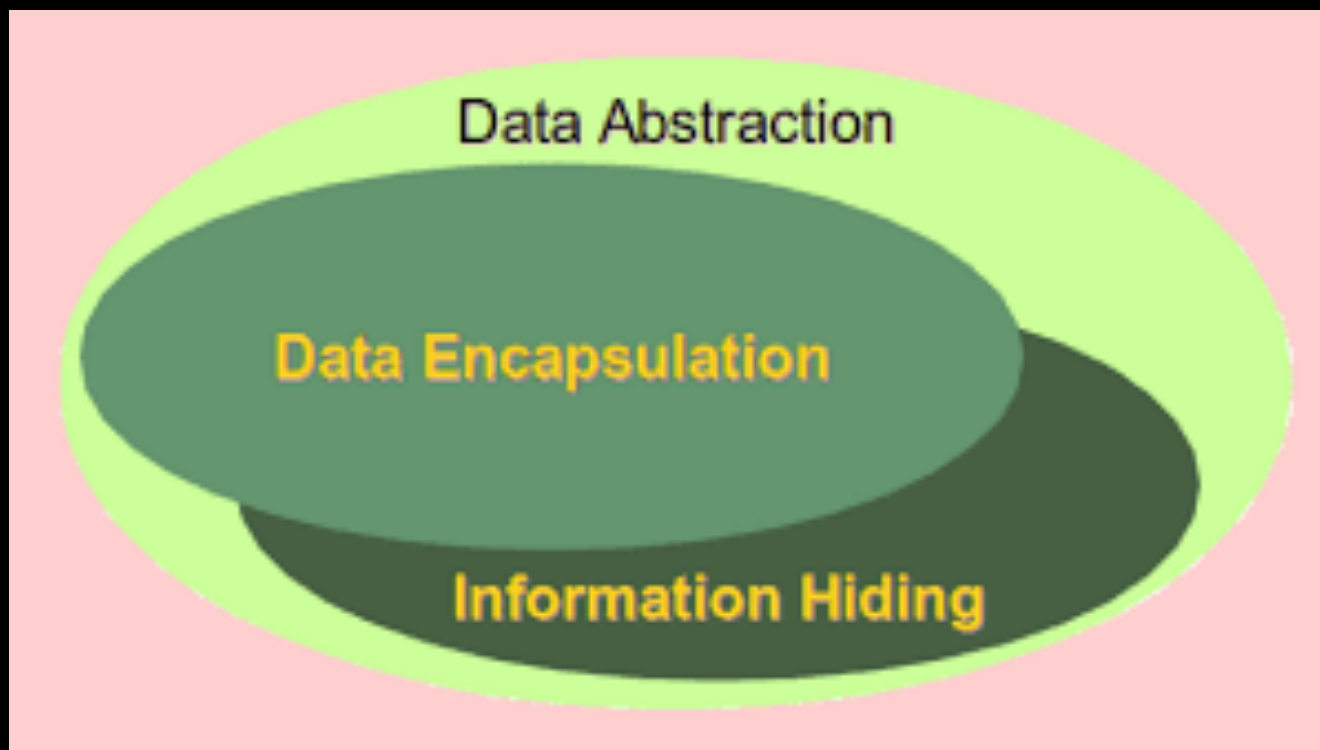
So complexity is **bad!!!!**

Write small units of code

Minimize Interactions/Coupling!!!

Enforce strict rules on how code interacts

Minimize complexity!!!



What is Software Engineering?

"The application of a **systematic**, **disciplined**, **quantifiable** approach to the development, operation, and maintenance of software"

IEEE Standard Glossary of Software Engineering Terminology

Big Ideas of Software Engineering

Modularity

Modifiability/Extensibility

Ease of Use

Fail-Safe Programming

Debugging

Testing

We will come back to these throughout the course

APPENDIX B

Next Time

Abstraction and OOP