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: (Me) Prof. Tiziana Ligorio

Undergraduate Teaching Assistants:

Parakram Basnet

Jayson Tan

Tommi Tsuruga

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

Linux Accounts

Communication and Help

Course Webpage

https://tligorio.github.io/CSCI235_Spring2019.html

Visit regularly for:

- Announcements
- Tentative Schedule and changes
- Lecture Notes
- Programming Projects

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	6 Projects (3 multi-part) (Project 6 **optional** to replace your lowest project or project-part grade)	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

Six programming projects
(Project 6 optional to replace your lowest project/project part grade)

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 Tuesdays 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: **MYR77V**

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

Linux Accounts

MUST HAVE!!!

Follow instructions in:

Programming Rules document on course web page

and

http://www.geography.hunter.cuny.edu/tbw/CS.Linux.Lab.FAQ/department_of_computer_science.faq.htm

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: **Fr 12-2pm HN1001A** (**subject to change**, check course page for announcements) or 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

Schedule:

Mo - 11:00 - 3:00

Tu - 11:00 - 5:00

We - all tutoring in lab C

Th - 1:00 - 6:00

Fr - 11:00 - 5:00

Be mindful
No disruptions!!!

Lab 1001C

Tutoring available to everyone during 136 labs

Schedule:

Mo - 1:00 - 7:15

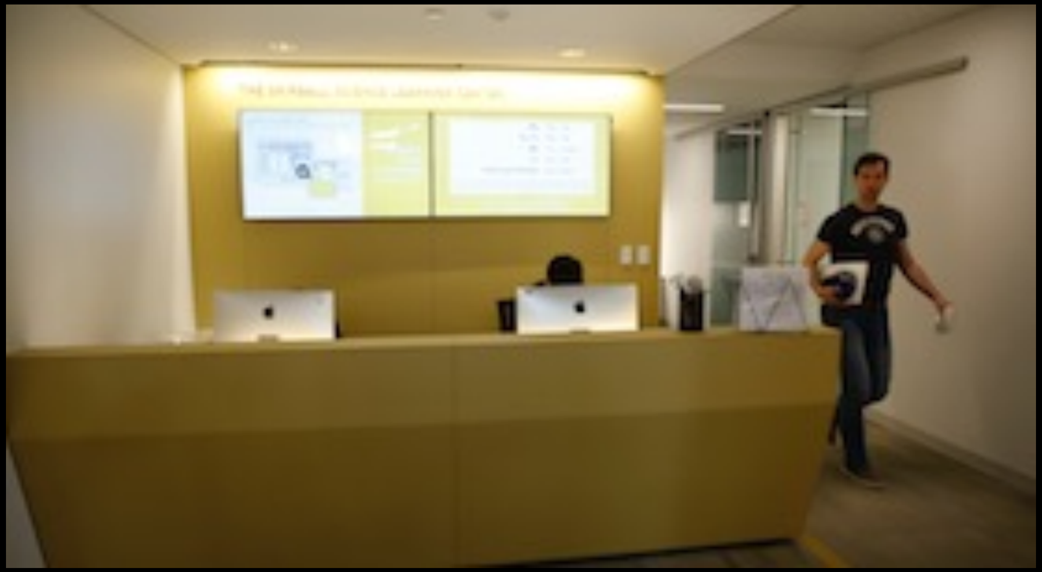
Tu - 2:00 - 4:00

We - 9:00 - 7:15

Th - 1:00 - 3:00, 7:00 - 9:00

Fr - all tutoring in lab B

Skirball Science Learning Center



Hunter East 7th floor

Drop-in tutoring for CSCI 235

- Check the schedule on their website or pop in for a a more accurate schedule for the day

https://library.hunter.cuny.edu/skirball-science-learning-center

Introducing
Lecture Activity

5% of final grade

- Attendance
- Work out a new problem

What is CSCI 235?

Programming => Software Analysis and Design

Think like a Computer Scientist:

Design and maintain complex programs

- Software Engineering, Abstraction, OOP

Design and represent data and its management

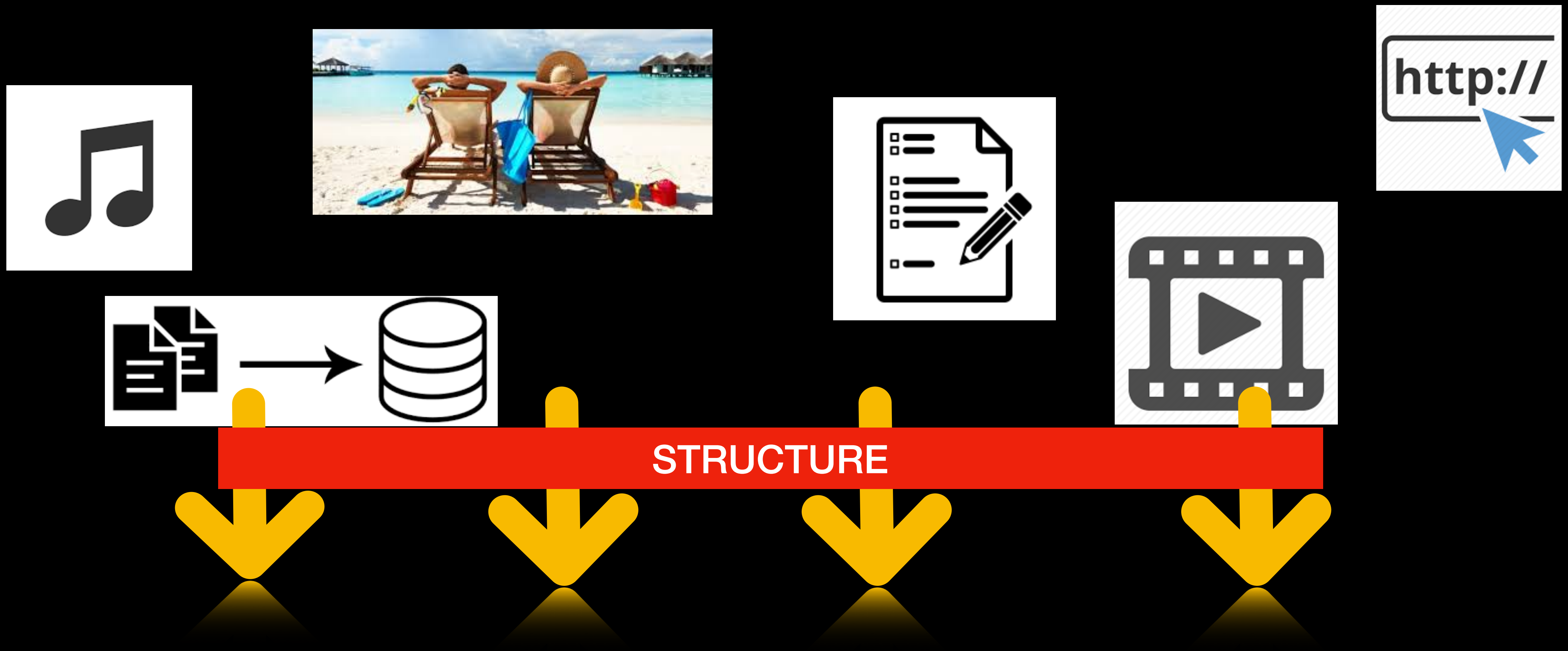
- Abstract Data Types

Implement data representation and operations

- Data Structures
- Algorithms

Understand Algorithm Complexity

“It’s all just bits and bytes…”



STRUCTURE

1010101010101010
0101010101010101
1001100110011001

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, …)

Illustrative example, may not be up to date

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

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

With 10 lines of code there may be

90 interactions

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

Problems of software complexity

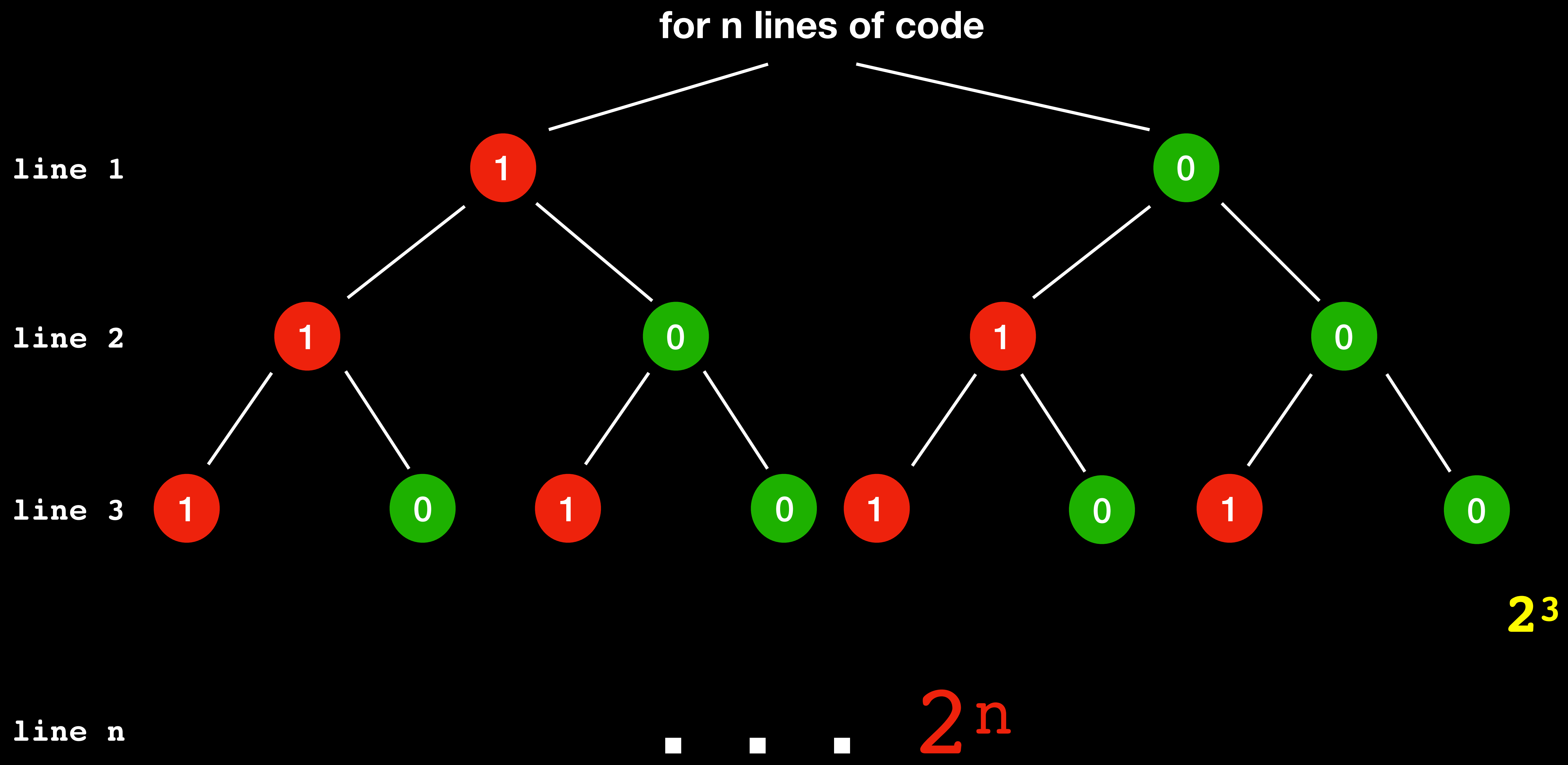
Assume line 1 shares same data with lines 6 and 23.

So in actuality the three lines all together form an interaction

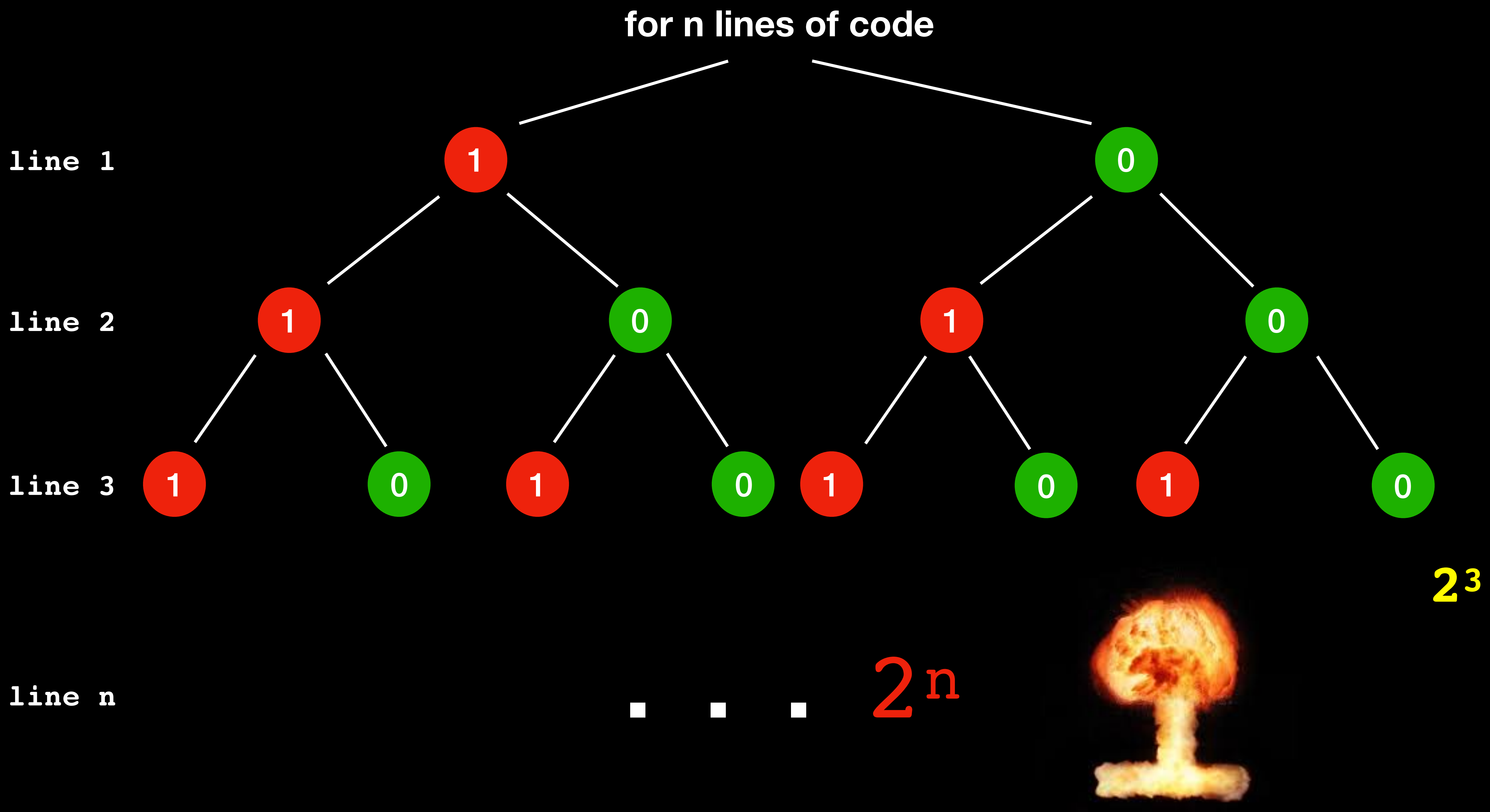
If we think of subsets of lines of code interacting (sharing the same data) . . .

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



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

Lecture Activity

ON THE BACK

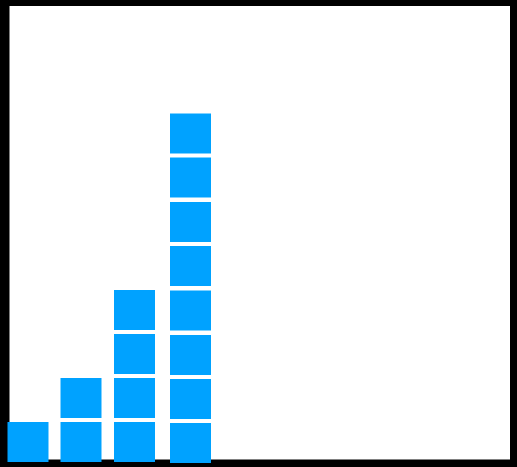
Draw a **very small** square on the leftmost bottom corner

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=AmFMJC45f1Q

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!!!

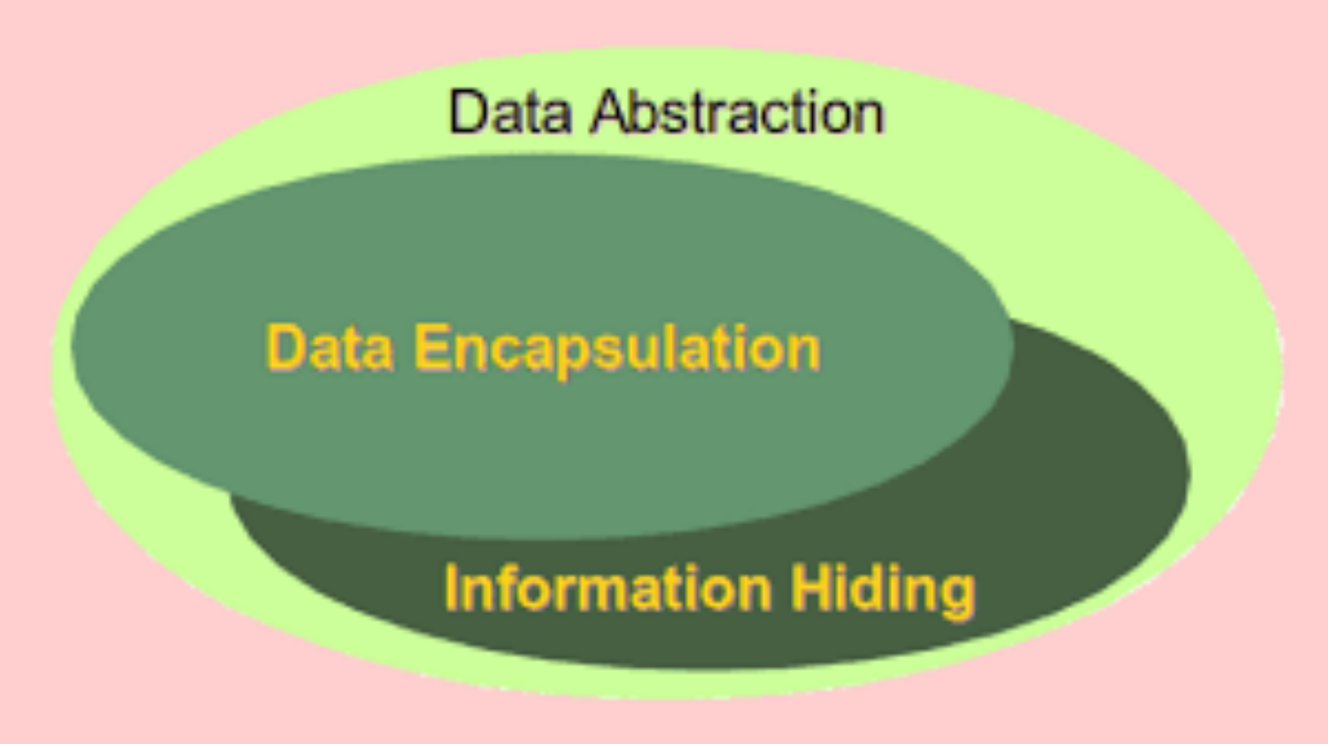
So complexity is **bad**!!!!

Write small units of code

Minimize Interactions!!!

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