

Programming Abstractions in C++

CS106B

Instructors:
Cynthia Bailey Lee
Julie Zelenski

Today's Topics

1. Introductions
2. Course structure and procedures
3. What is this class? What do we mean by “abstractions”?
4. Introduce the C++ language
 - › Variables
 - › Functions

Upcoming lectures:

- Strings
- Testing
- Our first abstraction!

Meet your instructors: Cynthia Bailey Lee

RESEARCH INTERESTS

- UCSD PhD in large-scale computing
- Recently: computer science education, DEI in tech, justice and social impacts of tech

TEACHING

- At Stanford since 2013
- CS106B, CS103, CS107, CS109, CS9, SSEA, CS80Q (introsem)

SOFTWARE ENGINEER

- iPhone educational games
- Document clustering and classification

AWAY FROM KEYBOARD

- Family, biking, hiking, pet chickens



Meet your instructors: Julie Zelenski

PROUD STANFORD ALUM (UNDERGRAD AND GRAD)

- FLI from CA Central Valley
- Coming to Stanford changed the arc of my life in every possible way
- Hope your experience is similarly transformative!
Software engineering

NEXT COMPUTER, ACQUIRED BY APPLE

LECTURER AT STANFORD

- Fantastic colleagues, awesome students
- CS department a research powerhouse AND deeply committed to education

AWAY FROM KEYBOARD

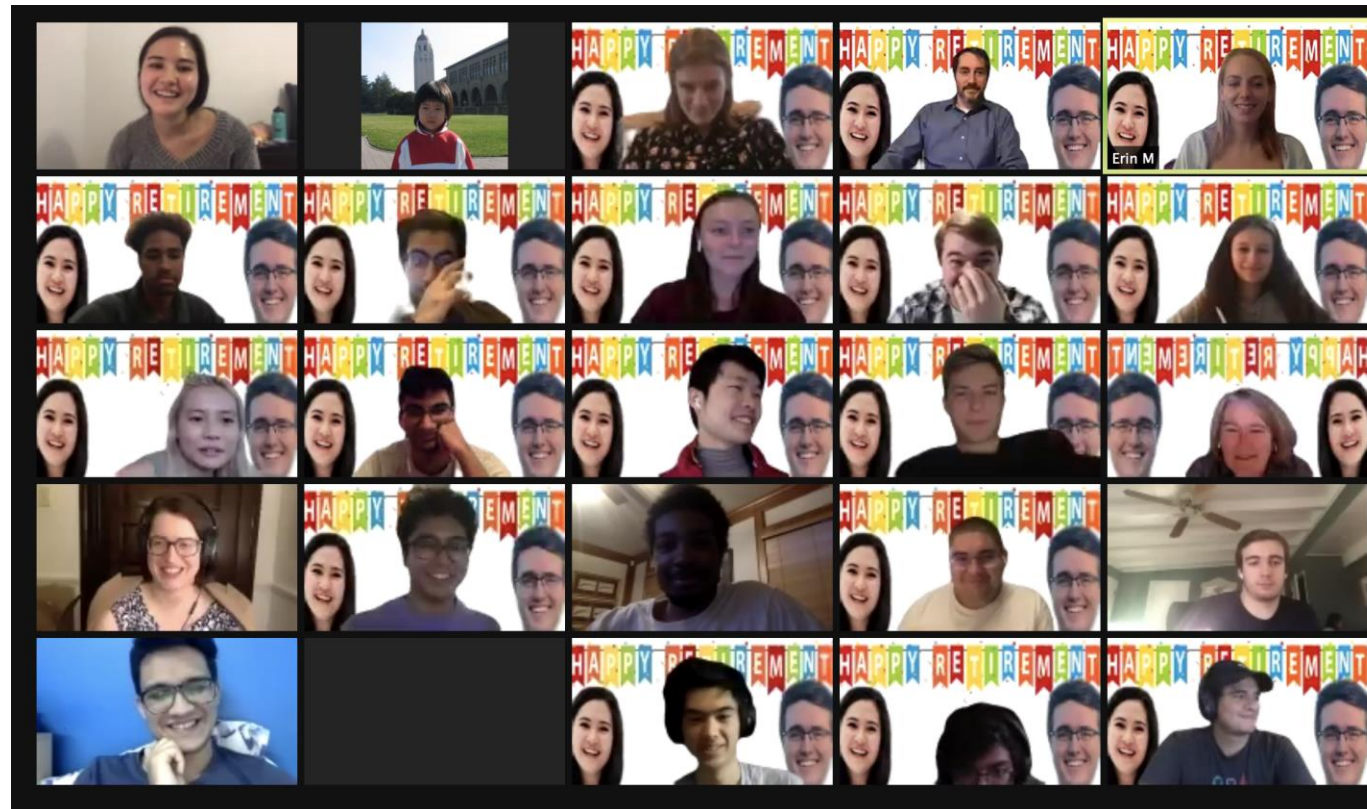
- Outdoors with family as much as possible



Discussion Section, Section Leaders (“SLs”)

Section Leaders are helpful undergraduate assistants who will:

- run your discussion section each week
- help you when you have questions
- grade your work
- ... and much more

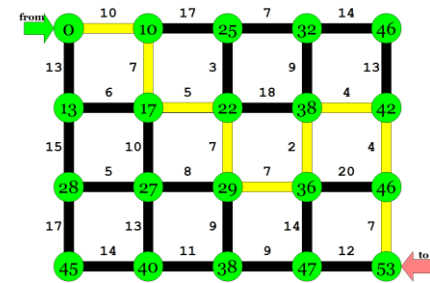
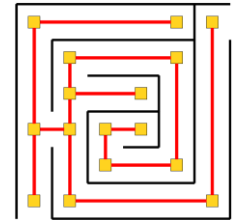
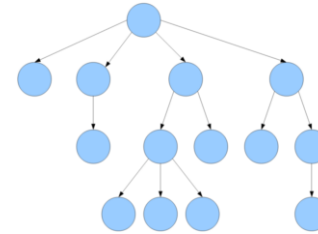
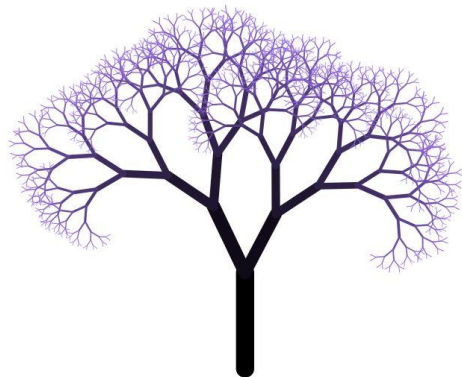


What is CS 106B?

CS 106B: Programming Abstractions

- solving big(ger) problems and processing big(ger) data
- learning to manage complex data structures
- algorithmic analysis and algorithmic techniques such as recursion
- programming style and software development practices
- familiarity with the C++ programming language

Prerequisite: CS 106A or equivalent



<http://cs106b.stanford.edu/>



Stanford University

CS 106L

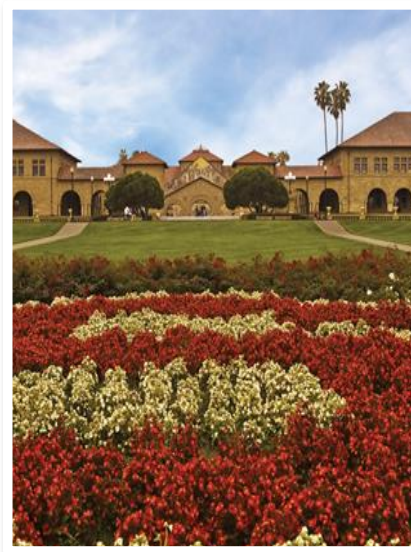
One unit course to learn the
C++ language in depth.

Lecture: T/Th 3:15-4:45 in 380-380C
Website: <http://cs106l.stanford.edu>

Questions?
Email us at:
sath@stanford.edu
fmcerk@stanford.edu

Course Logistics

QUICK OVERVIEW OF HOW TO
EARN THE GRADE YOU WANT
IN CS106B



What is this class
about?

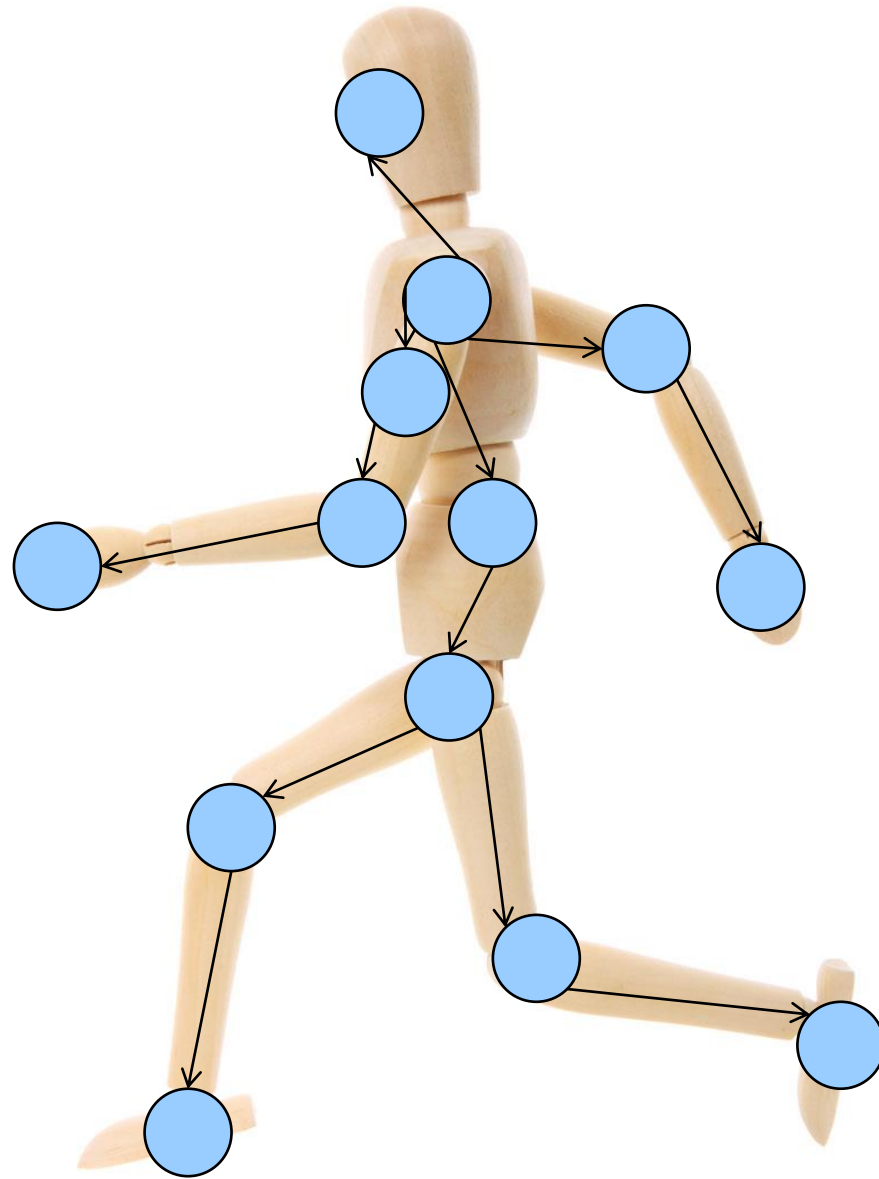
What do we mean by
“abstractions”?

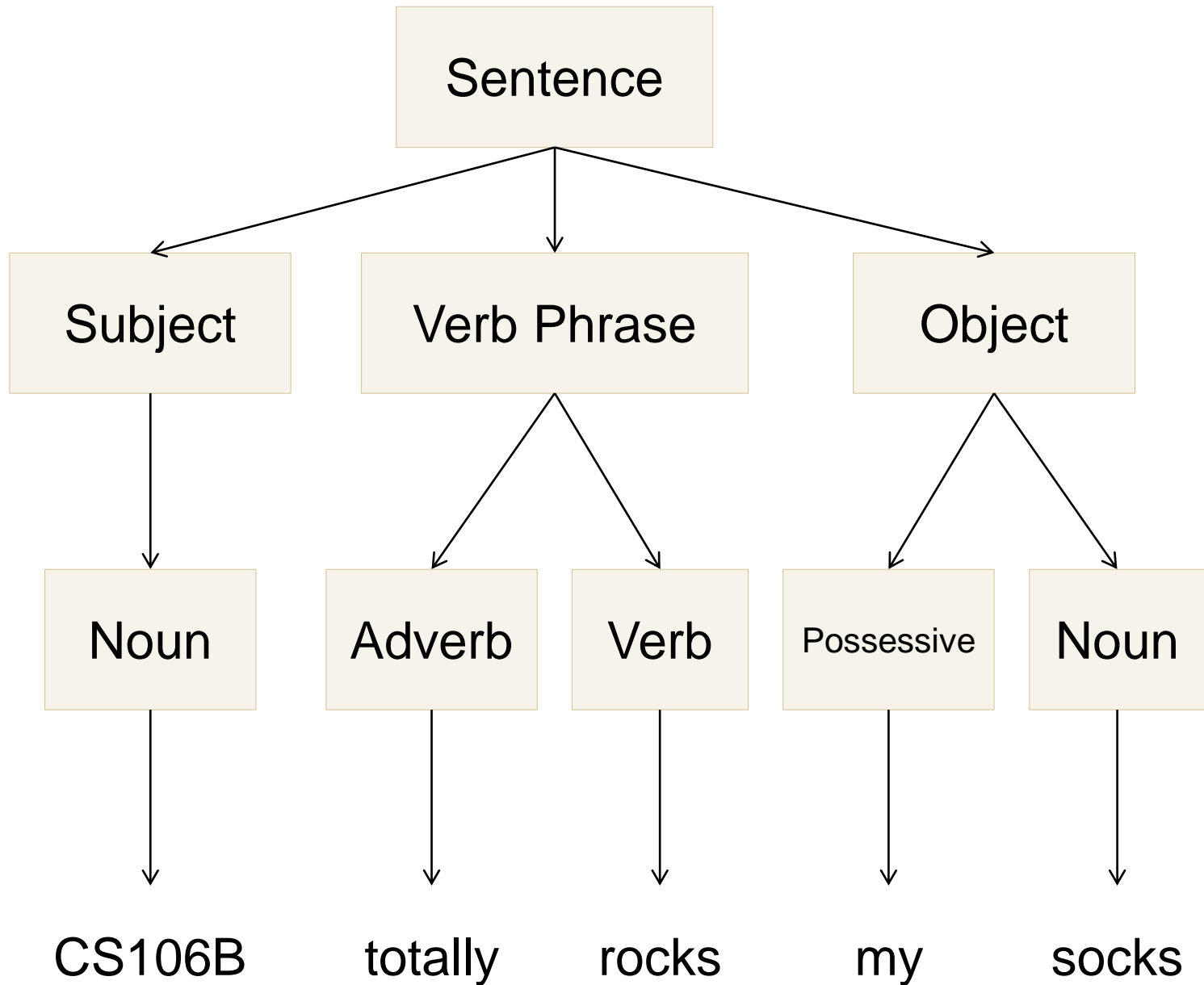


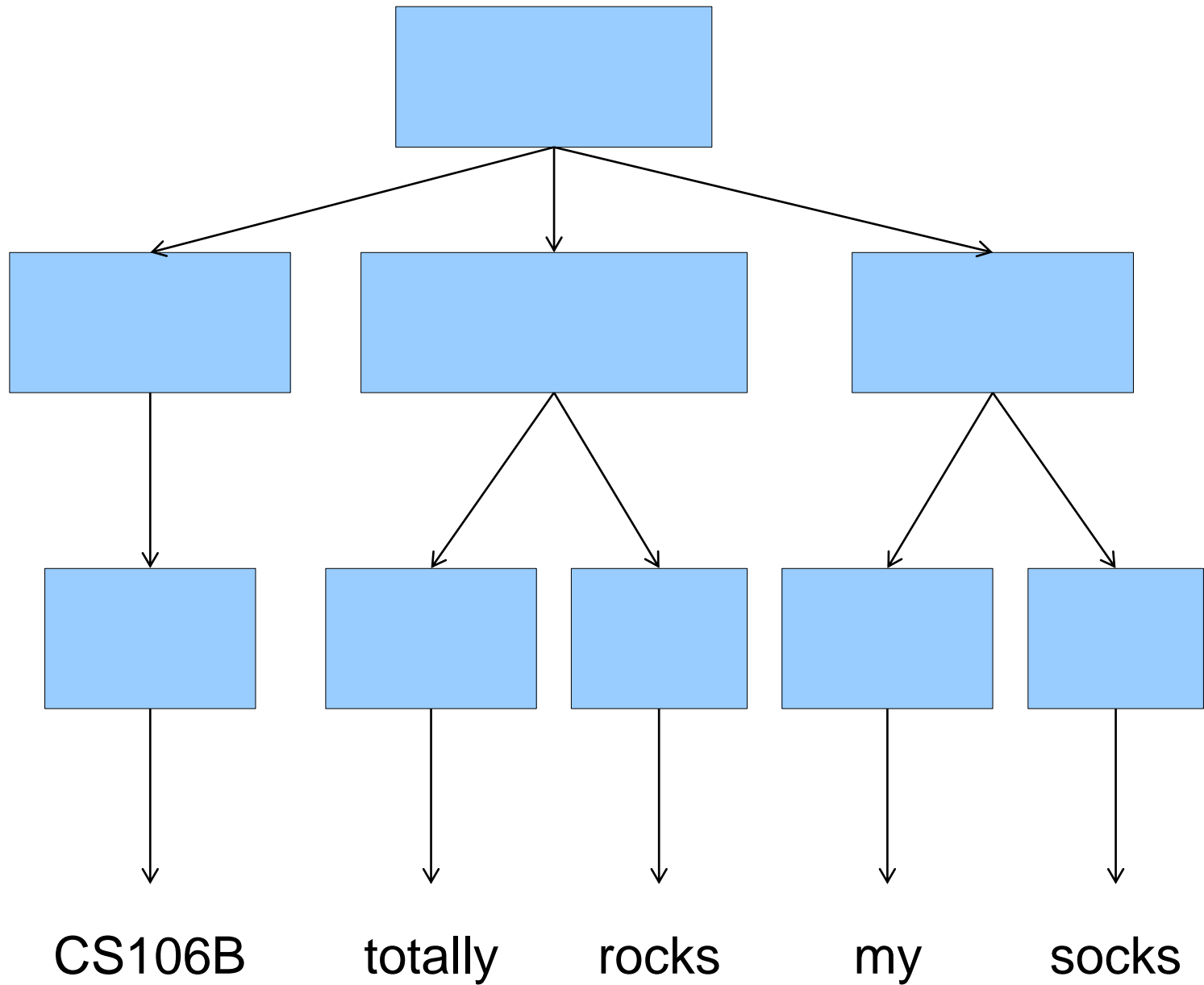
Colatina, Carlos Nemer

This file is licensed under the [Creative Commons Attribution 3.0 Unported](https://creativecommons.org/licenses/by/3.0/) license.

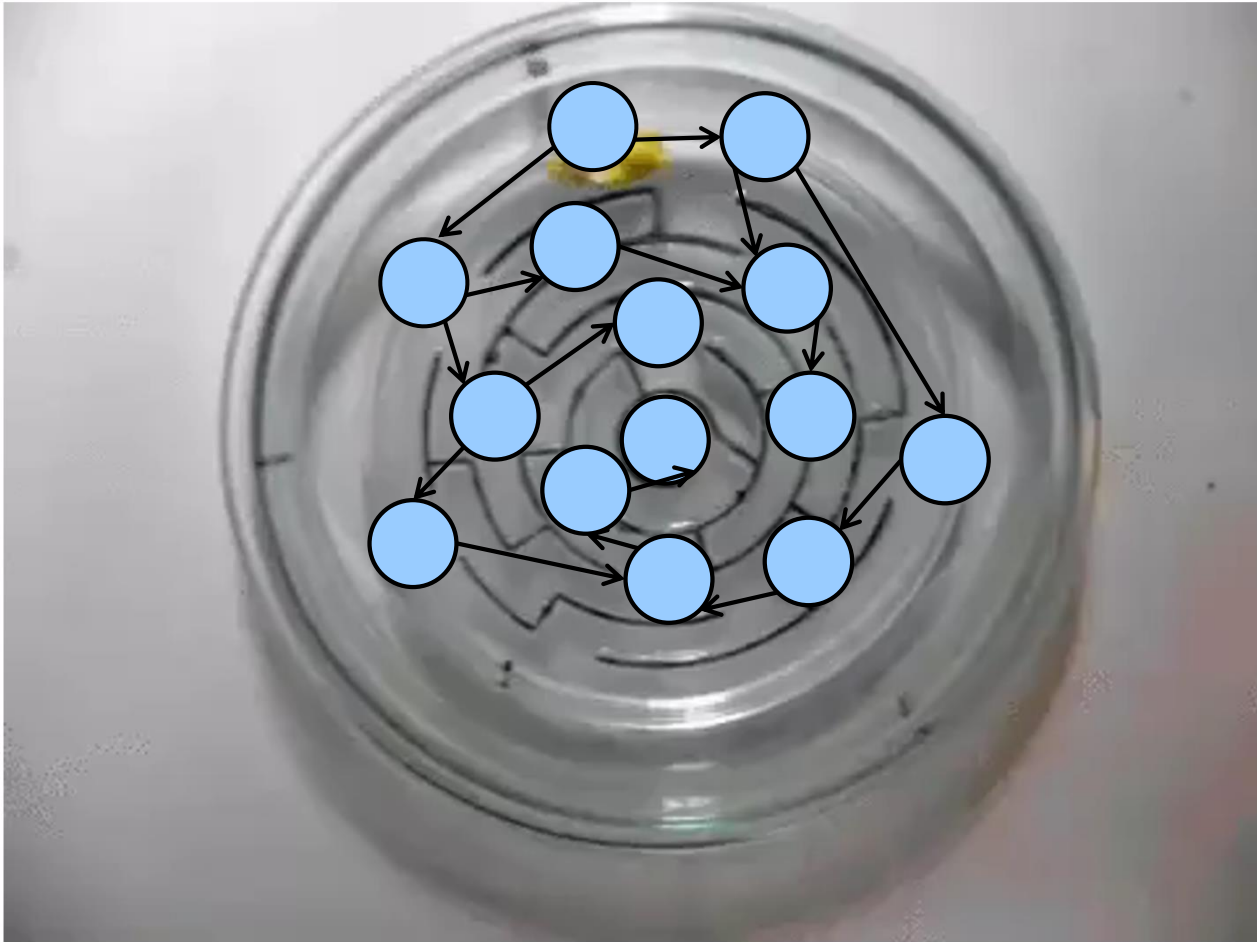






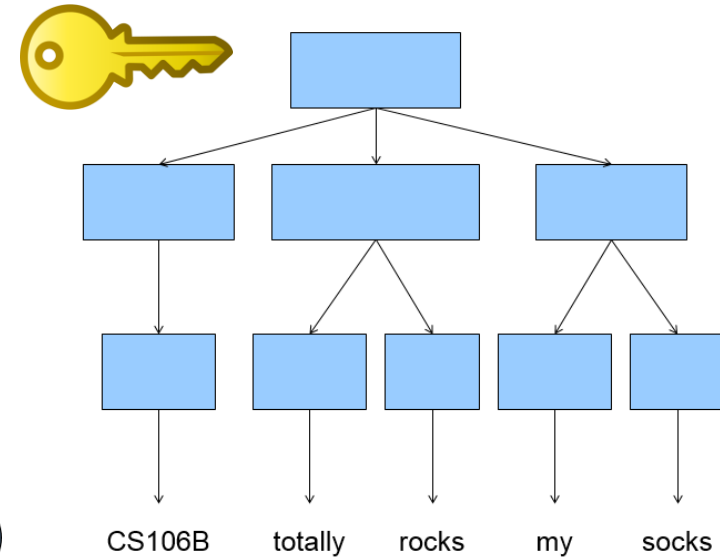
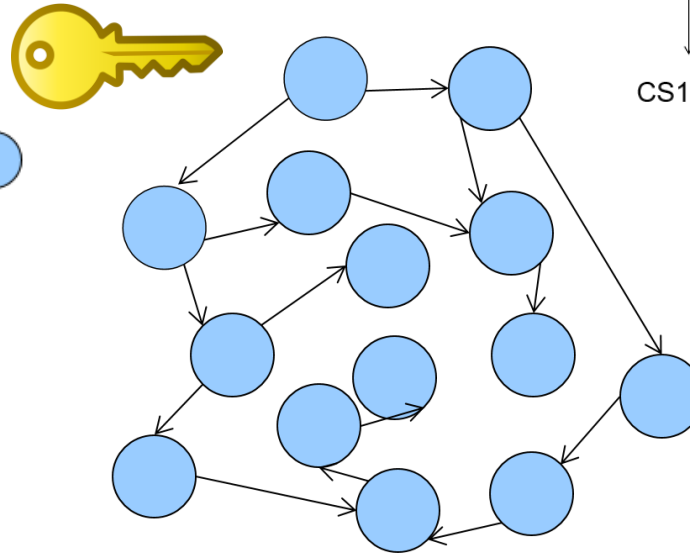
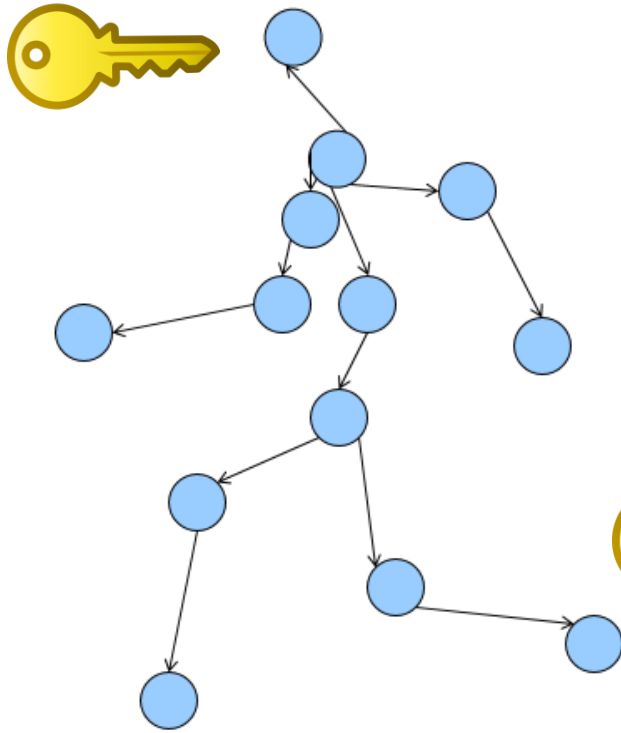






In CS106B, you'll learn to:

1. Identify common underlying structures
2. Apply known algorithmic tools that solve diverse problems that share that structure



Building a vocabulary of **abstractions**
makes it possible to represent and solve a huge
variety of problems using known tools.

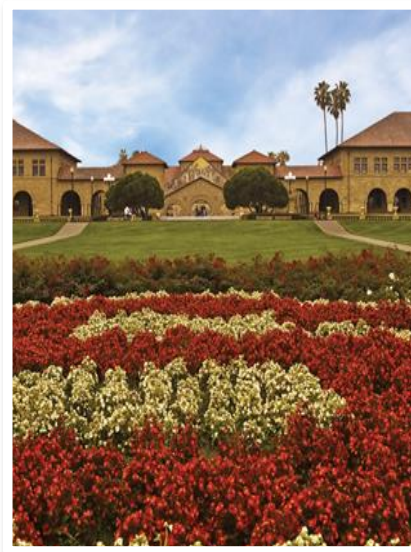
A little bit of advice for this course...

You'll have what is effectively a superpower

**Spend some time thinking about how
you'll use it.**

Welcome to C++

LET'S START CODING!!



First C++ program (1.1)

```
/*  
 * hello.cpp  
 * This program prints a welcome message  
 * to the user.  
 */  
#include <iostream>  
#include "console.h"  
using namespace std;  
  
int main() {  
    cout << "Hello, world!" << endl;  
    return 0;  
}
```

Include statements are like imports in Python. More on this in a moment.


Every C++ program has a **main** function. The program starts at main and executes its statements in sequence.

At program end, **main** returns 0 to indicate successful completion. A non-zero return value is an error code, but we won't use this method of error reporting in this class so we will always return zero.

C++ variables and types (1.5-1.8)

- The C++ compiler is rather picky about *types* when it comes to variables.
- Types exist in languages like Python (see the two code examples at right), but you don't need to say much about them in the code. They just happen.
- The **first time** you introduce a variable in C++, you need to announce its type to the compiler (what kind of data it will hold).
 - › After that, just use the variable name (don't repeat the type).
 - › You won't be able to change the type of data later! C++ variables can only do one thing.

C++



```
int x = 42 + 7 * -5;
double pi = 3.14159;
char letter = 'Q';
bool done = true;

x = x - 3;
```

Python

```
x = 42 + 7 * -5
pi = 3.14159
letter = 'Q'
done = True

x = x - 3
```

More C++ syntax examples (1.5-1.8)

```
for (int i = 0; i < 10; i++) {           // for loops
    if (i % 2 == 0) {                   // if statements
        x += i;
    }                                    /* two comment styles */
}
```

```
while (letter != 'Q' && !done) {        // while loops, logic
    x = x / 2;
    if (x == 42) { return 0; }
}
```

```
binky(pi, 17);                          // function call
winky("this is a string");               // string usage
```

Some C++ logistical details (2.2)

```
#include <libraryname>    // standard C++ library  
#include "libraryname.h" // local project library
```

- Attaches a library for use in your program
- Note the differences (common bugs):
 - <> vs " "
 - .h vs no .h

```
using namespace name;
```

- *Mostly, just don't worry about what this actually does/means! Copy & paste the std line below into the top of your programs.*
- Brings a group of features into global scope so your program can directly refer to them
- Many C++ standard library features are in namespace std so we write:
 - › using namespace std;
 - › “std” is short for “standard”