Programming Abstractions in C++

CS106B

Instructor: Cynthia Bailey
Head TA: Yasmine Alonso

Meet your instructor: Cynthia Bailey

- I've been teaching at Stanford for over 10 years, primarily CS106B and CS103. I also teach seminars "AI Governance" and "Race & Gender in Silicon Valley."
- For both undergrad and PhD, I studied CS at UC San Diego, focus on supercomputing and machine learning.
- I've worked as a software engineer or consultant at startups, NASA, and law firms.
- My partner and I are the RFs in J-Ro House in Wilbur.
- In 23-24, I was on leave from Stanford, living in DC and advising the U.S. Senate on AI legislation. I met with stakeholders like NATO officials, CEOs, artists, and academics, and wrote bills relating to AI/tech.
- Hobbies: Family, biking, hiking, cooking
- Reliable conversation starter: ask me what animal(s) I'm currently fostering

Meet your Head TA: Yasmine Alonso

- B.S. Computer Science (AI), 2025
- M.S. Computer Science (Computer + Network Security), 2027
- SL for CS106A + CS106B since Fall 2022
- Head TA for CS106B since Fall 2025! Yay!
- Here to help with whatever you need :) you can always email me for support!
- Interest in Software Engineering (worked at Zipline last summer) and also teaching! Considering applying for PhD programs this fall :)
- Outside of school: music (violin/piano), ballet, baking, board games, swimming, random arts/crafts

Discussion Section, Section Leaders (“SLs”)

Section Leaders are helpful undergraduate assistants

- Your personal trainer in 106B
- Meet with the same SL each week
- (consider being one in the future!)

Apply for CS 100B
(the CS 106B ACE section!)

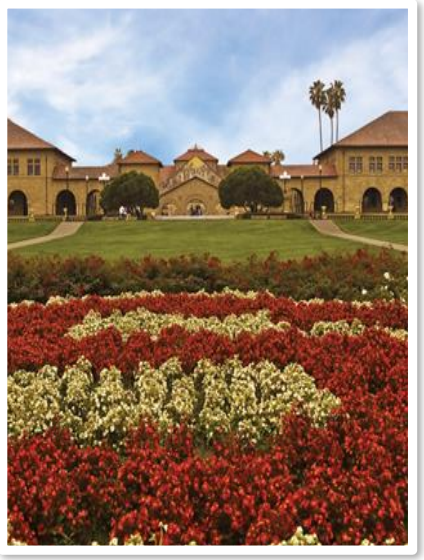
- What is ACE (Additional Courses for Engineers)?
  - A **1-unit supplemental course** that runs alongside CS 106B
  - Provides additional **small-group support** via a weekly ACE section, ACE-only office hours, and additional exam review sessions
  - Designed to **help students build confidence and succeed** in rigorous engineering courses
  - More info on the ACE website & CS 100B syllabus

Apply for CS 100B
(the CS 106B ACE section!)

- How can I apply?
  - Fill out the application on the ACE website or here
  - **Due Friday Week 1 (1/9) at 11:59 PM**
  - Decisions released no later than Sunday 1/11
- Am I eligible to apply?
  - **Yes!** As long as you can make section at **1:30 pm - 3:20 pm on Tuesdays** (starting week 2)

Course Logistics

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



Course Grade Essentials

- **35% Programming assignments**
  - A0 through A7
  - A0 is course setup, out now due Friday Week 1!! (but everyone must complete, so turn it in late if you need to; we will not charge you regular late days for it)
- **20% Midterm**
  - Monday, February 9, 2026, 7-9pm
  - On A0, you will be asked to check all your personal schedule things and attest that you will be able to attend
- **35% Final Exam**
  - Monday, March 16, 2026, 8:30am-11:30am
  - You must pass the final exam to pass the course
  - Again, on A0 you will be asked to attest that you will be able to attend
- **5% Section participation**
  - To get an A in the course, you must not miss more than one section without an approved excuse
- **5% Lecture attendance**
  - 1% off final grade per missed lecture after 1 “freebie,” but available 48-hr extensions to complete make-up quizzes if you miss
  - We won’t start tracking until Week 2

Getting Started: A0

- **A0 is course setup, out now, due Friday Week 1!!**
  - Everyone must complete it, so turn it in late if you need to
  - We will not charge you regular late days for it

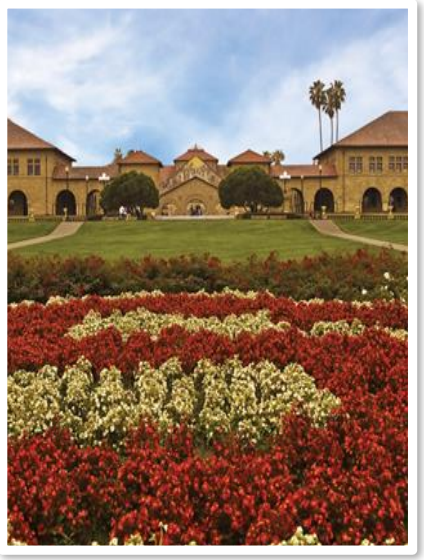
- **A0 helps you accomplish a number of essential things:**
  - Set up your QT Creator coding environment
  - Review course honor code policy
  - Check your schedule to ensure availability for the exams
  - Practice turning homework in on Paperless

- **To help you with this, there are a couple resources:**
  - Course coding files organization tips video, starring Yasmine!
  - **Qt install help session on Friday 1/9 from 2:30-4:30 in CoDa B45**

- **Instructor/TA office hours and "Lair" SL help hours start Week 2**

CS106B Course Culture

HOW TO SUCCEED IN THIS LONGSTANDING COMMUNITY AT STANFORD



Community norms and expectations

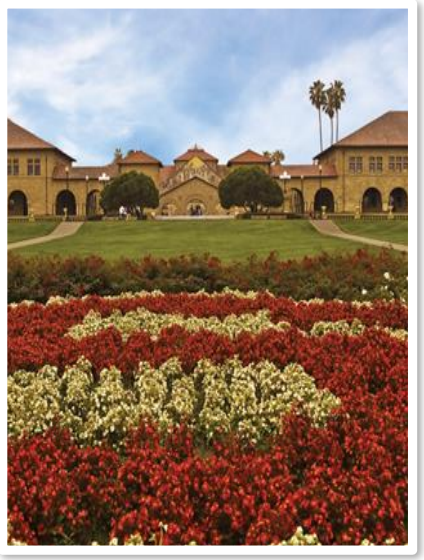
- **Celebrate discovery and growth.** No gatekeeping, shaming, or comparisons based on who knew what coming in.
  - *Example of things we're not going to do: audience "questions" in lecture that are just showing off that you know some jargon.*

- **Shed "zero-sum" and scarcity attitudes.** There are (still!) plenty of tech jobs.
  - *Others gaining strength in the power of coding doesn't take power away from you. Be helpful and encouraging, try to feel as genuinely happy when others around you succeed as when you yourself succeed. (Tip: this is a skill you will need for all levels of management/leadership in your career.)*

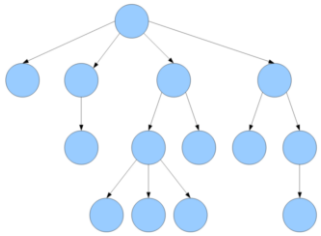
- **Do your own work.** We take this very seriously, because that's how you grow.
  - *Nobody gets good at yoga by watching videos. You have to get on the mat, and sometimes you have to sweat. No shortcuts. We do enforce, but it can't only be about enforcement—you need to decide within yourself to hold the line on integrity.*

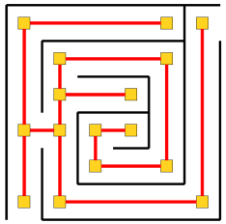
Programming Abstractions in C++

WHAT ARE "PROGRAMMING ABSTRACTIONS"?



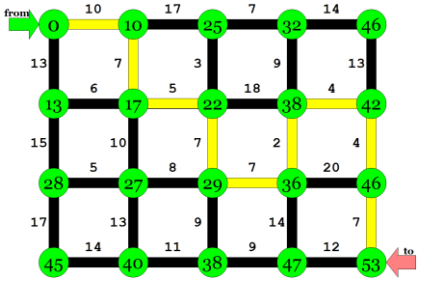
What is CS 106B?



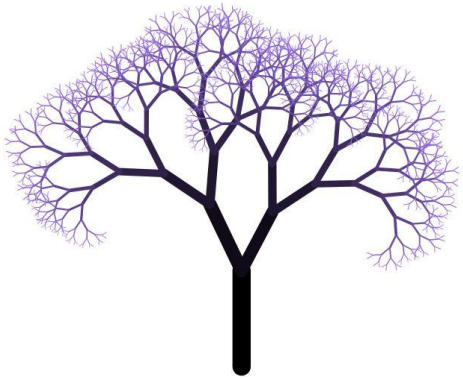


CS 106B: Programming Abstractions

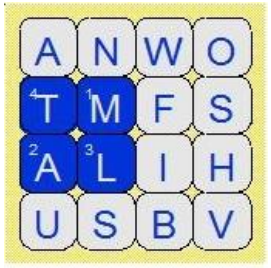
- solving big(ger) problems and processing big(ger) data
- learning to manage complex data structures
- algorithmic analysis and algorithmic techniques such as recursion
- programming style and software development practices
- familiarity with the C++ programming language



Prerequisite: CS 106A or equivalent



http://cs106b.stanford.edu/

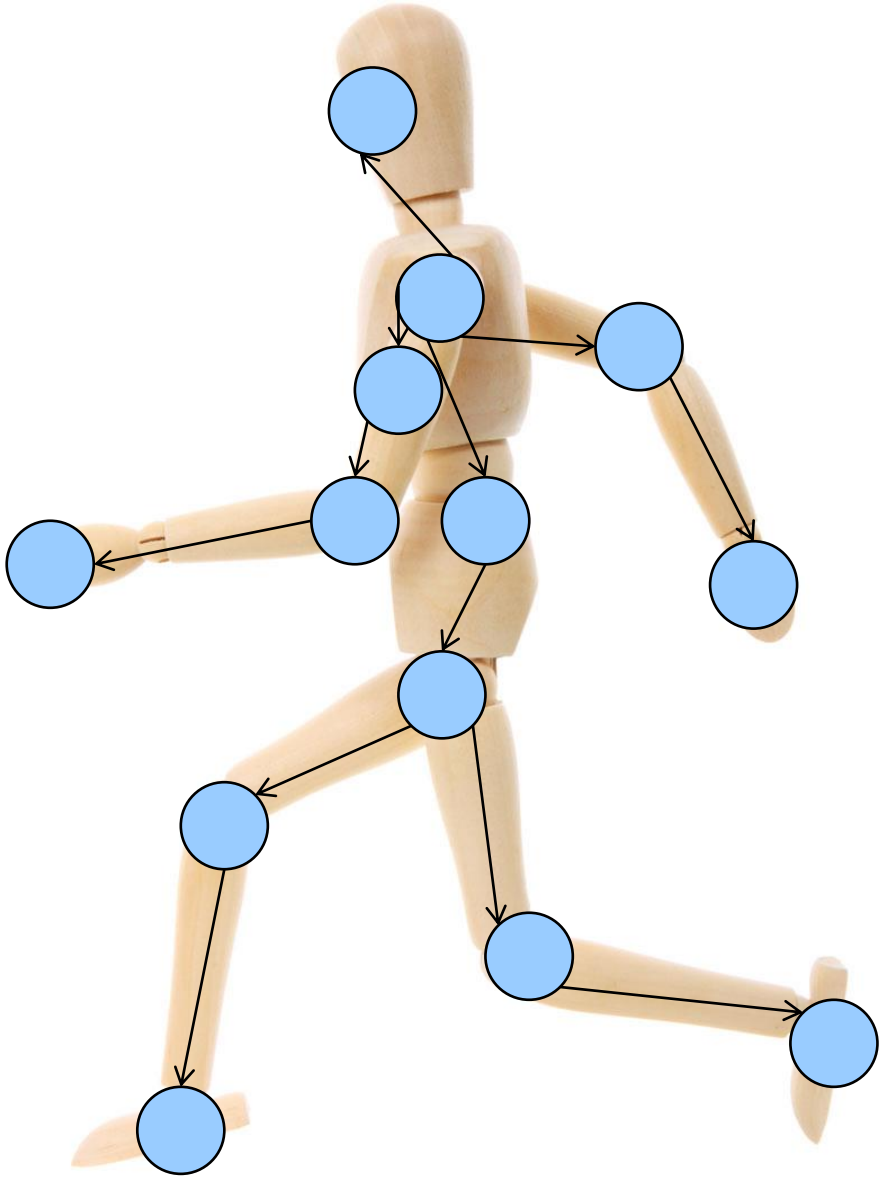


Stanford University

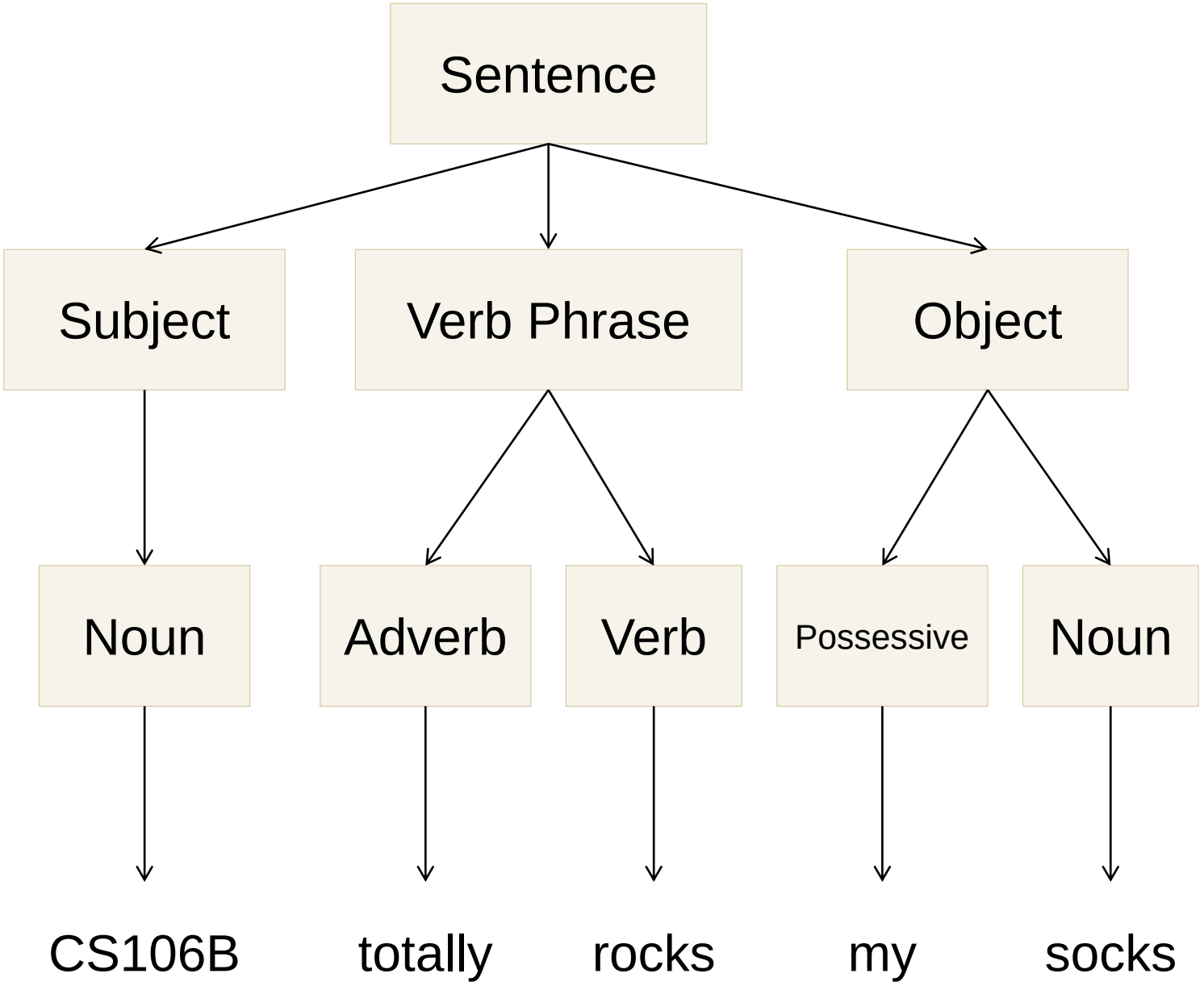
What is this class about?

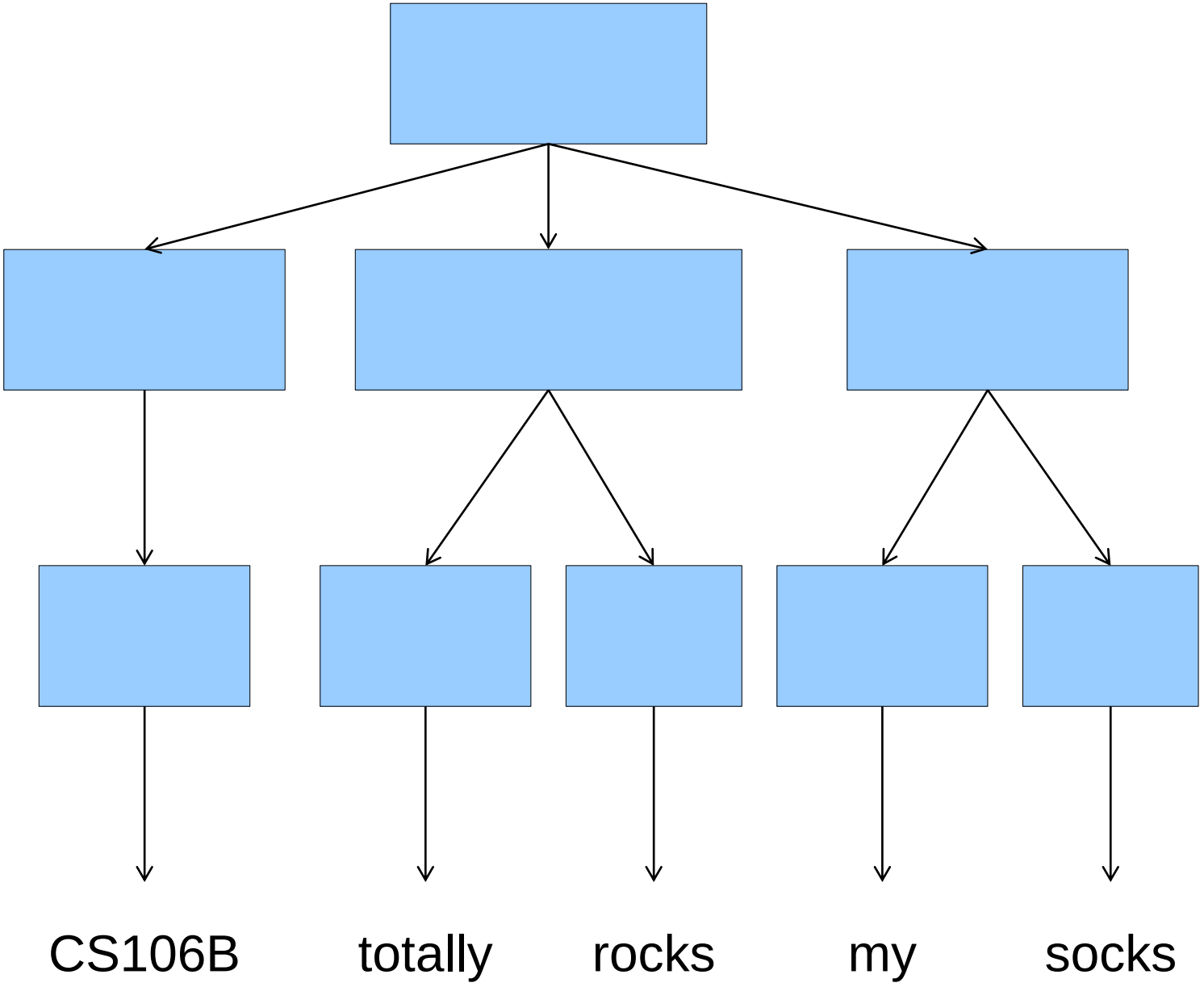
What do we mean by “abstractions”?

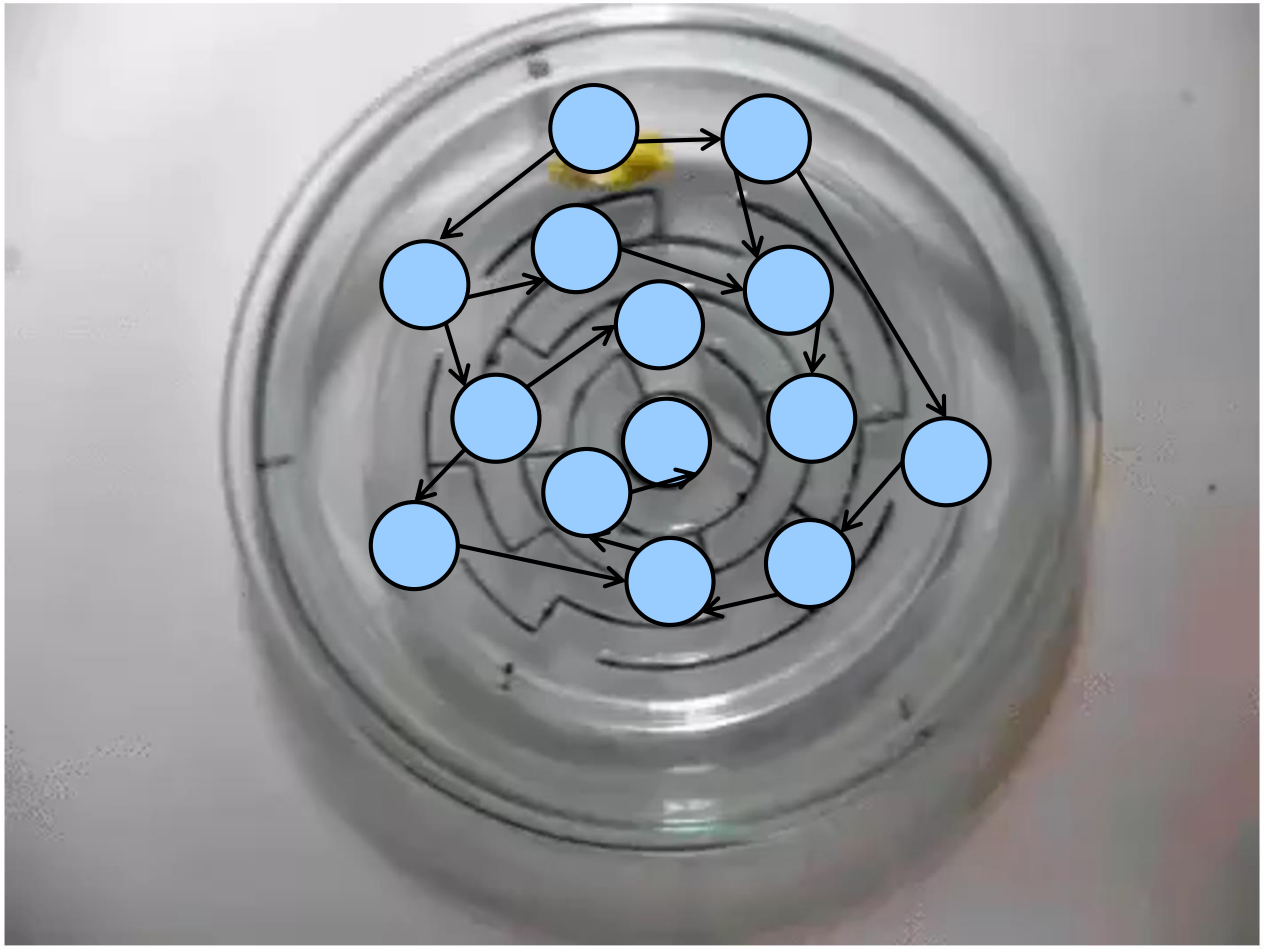
http://www.publicdomainpictures.net/pictures/10000/velka/1-1265899974oKJ9.jpg

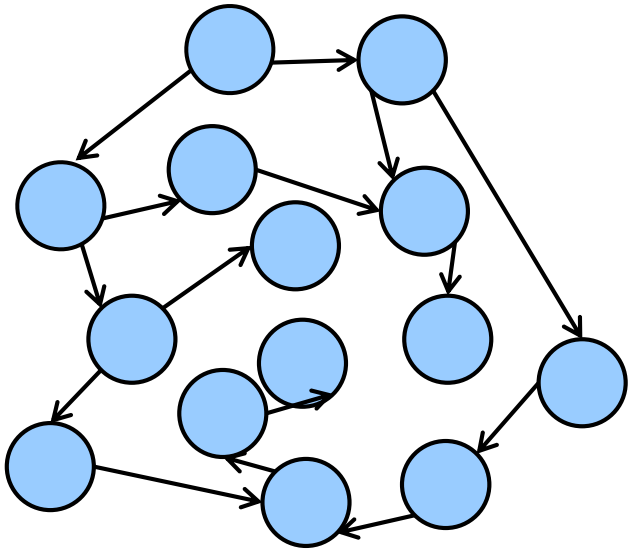


http://www.publicdomainpictures.net/pictures/10000/velka/1-1265899974oKJ9.jpg


Sentence
Subject
Verb Phrase
Object
Noun
Adverb
Verb
Possessive
Noun
CS106B
totally
rocks
my
socks

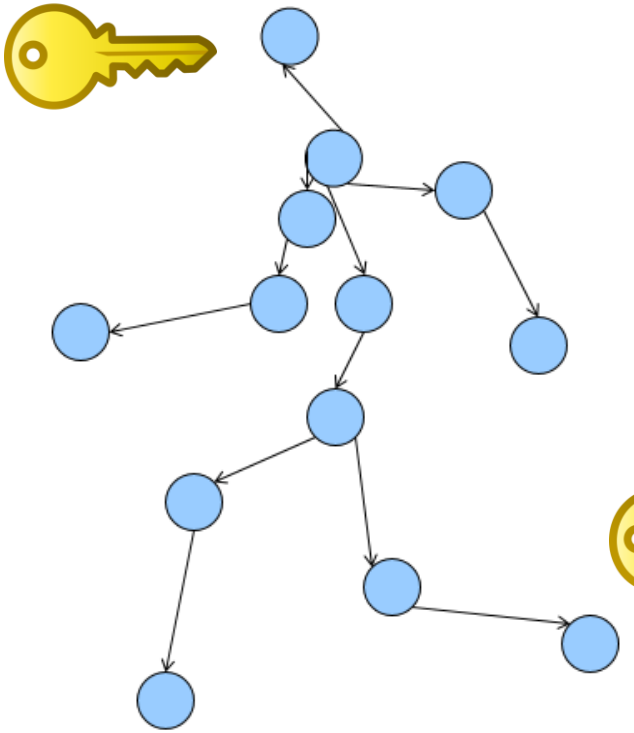

CS106B
totally
rocks
my
socks

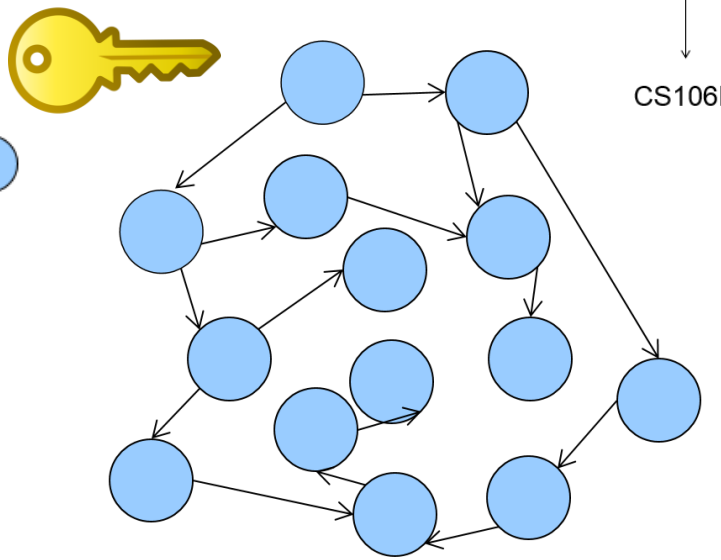


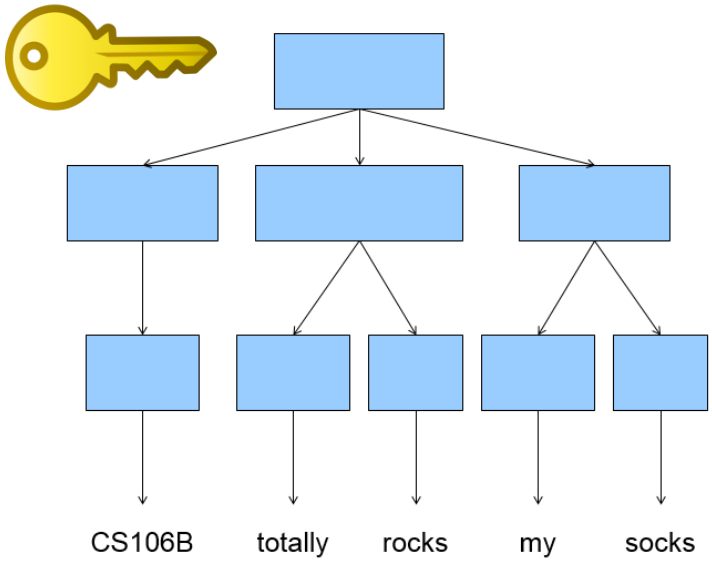


In CS106B, you'll learn to:

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 powerful algorithmic tricks & tools.

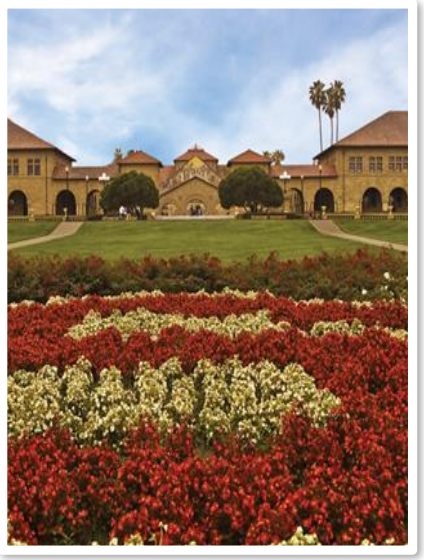
After this course…
you'll have what is
effectively a superpower.

You should spend some time thinking about how you'll use it!

(We'll help you get started with ethics reflections embedded throughout the course)

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 Java/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"