Programming Abstractions in C++

CS106B

Cynthia Lee
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
- I/O streams
Your instructor: Cynthia Lee

Research:
- PhD @ UCSD: market-based resource allocation in large-scale systems
- Recently: computer science education

Teaching:
- At Stanford since 2013, 3 years at UCSD
- CS106B, CS106X, CS103, CS107, CS109, CS9, SSEA, CS80Q (introsem)

Software engineer:
- iPhone educational games
- Document clustering and classification

When I’m not working:
- Family, biking 😊, hiking, pet chickens
SECTION LEADERS: HELPFUL UNDERGRADUATE ASSISTANTS WHO WILL:

- run your discussion section each week
- grade your homework assignments and exams
- help you when you have questions
- ... 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/
CS 106L

One unit course to learn and practice C++ programming in depth.

Lecture: T/Th 3:30-4:20 in 380-380C
Website: http://cs106l.stanford.edu

Questions?
Email us at averywang@stanford.edu aszeng@stanford.edu
Late Days

**Late day**: allows you to submit a homework **1 Lecture Day** late

- You get 3 free (no-penalty) late days for the quarter, but only one or two can be used on a given assignment
- After your late days are used up, 1 bucket grade is deducted per day
- NO SUBMISSIONS are accepted after 2 days late

Example:

<table>
<thead>
<tr>
<th>Week</th>
<th>Sun</th>
<th>Mon</th>
<th>Tue</th>
<th>Wed</th>
<th>Thu</th>
<th>Fri</th>
<th>Sat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>due</td>
<td>1 day late</td>
<td>1 day late</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>2 days late</td>
<td>2 days late</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
What is this class about?

What do we mean by “abstractions”?
CS106B totally rocks my socks.
CS106B totally rocks my socks
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.
C++ variables and types (1.5-1.8)

C++

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

JavaScript

```javascript
var x = 42 + 7 * -5;
var pi = 3.14159;
var letter = 'Q';
var done = true;
```

Java

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

Python

```python
x = 42 + 7 * -5
pi = 3.14159
letter = 'Q'
done = True
```
Every C++ program has a `main` function. The program starts at `main` and executes its statements in sequence.

At program end, `main` returns an integer value. Zero means successful run, non-zero is error code. (We will always user zero in this class, and use other error reporting techniques.)
More C++ syntax examples (1.5-1.8)

```cpp
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)

```cpp
#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`

```cpp
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`:
  - using namespace `std;`