Programming Abstractions

CS106X

Cynthia Lee

Today's Topics

Introducing C++ from the Java Programmer's Perspective

- Absolute value example, continued
  - C++ strings and streams

ADTs: Abstract Data Types

- Introduction: What are ADTs?
- Queen safety example
  - Grid data structure
  - Passing objects by reference
    - `const` reference parameters
  - Loop over "neighbors" in a grid

Strings in C++

STRING LITERAL VS STRING CLASS

CONCATENATION

STRING CLASS METHODS

Using cout and strings

```
int main(){
   int n = absoluteValue(-5);
   string s = "|-5|";
   s += " = ";
   cout << s << n << endl;
   return 0;
}
int absoluteValue(int n) {
   if (n<0){
      n = -n;
   }
   return n;
}
```

- This prints |-5| = 5
- The + operator concatenates strings, and += works in the way you'd expect.

Using cout and strings

```
int main(){
   int n = absoluteValue(-5);
   string s = "|-5|" + " = ";
   cout << s << n << endl;
   return 0;
}

int absoluteValue(int n) {
   if (n<0){
      n = -n;
   }
   return n;
}
```

But SURPRISE!…this one doesn't work.

C++ string objects and string literals

- In this class, we will interact with two types of strings:
  - String literals are just hard-coded string values:
    - `"hello!" "1234" "#nailedit"`
    - They have no methods that do things for us
    - Think of them like integer literals: you can't do `"4.add(5);" //no`
  - String objects are objects with lots of helpful methods and operators:
    - `string s;`
    - `string piece = s.substr(0,3);`
    - `s.append(t); //or, equivalently: s+= t;`

String object member functions (3.2)

Member function name	Description
`s.append(str)`	add text to the end of a string
`s.compare(str)`	return -1, 0, or 1 depending on relative ordering
`s.erase(index, length)`	delete text from a string starting at given index
`s.find(str)` `s.rfind(str)`	first or last index where the start of *str* appears in this string (returns `string::npos` if not found)
`s.insert(index, str)`	add text into a string at a given index
`s.length()` or `s.size()`	number of characters in this string
`s.replace(index, len, str)`	replaces *len* chars at given index with new text
`s.substr(start, length)` or `s.substr(start)`	the next *length* characters beginning at *start* (inclusive); if *length* omitted, grabs till end of string

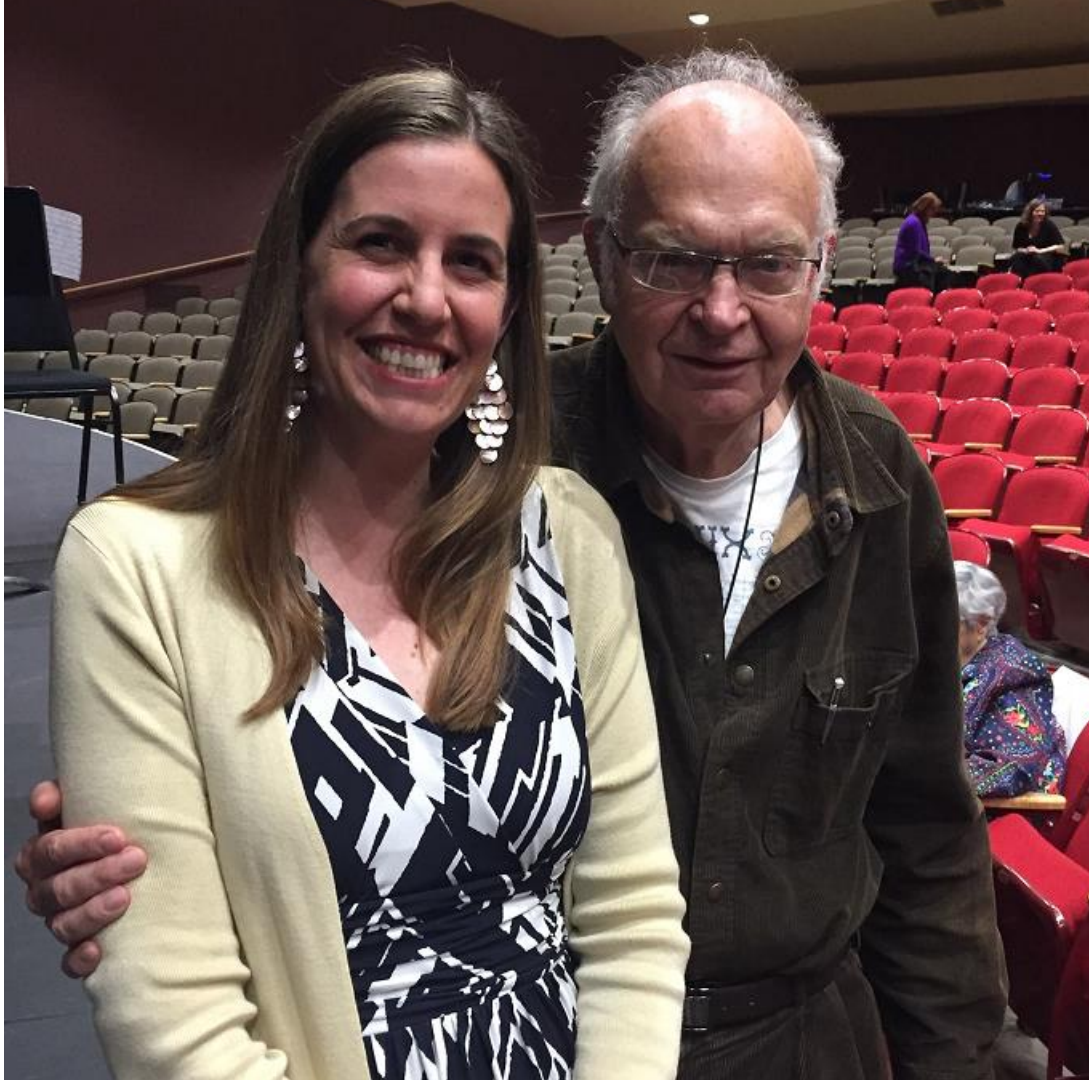
```
string name = "Donald Knuth";
if (name.find("Knu") != string::npos) {
    name.erase(7, 5);              // "Donald"
}
```

Aside: Donald Knuth

Emeritus (*i.e.*, retired) faculty in our dept.

Legend of computer science

If you're lucky, you'll still see him around campus from time to time



Recap: C++ string objects and string literals

- Even though they are different types, you can mix them as long as there is a string object around to be the "brains" of the operation:
  - Yes:

```
string s;
s = "hello!"    //string knows how to convert literal
s = s + "1234"; //string has + defined as concatenation
char ch = 'A';  //a single ASCII character with ' not "
s += ch;        //string knows how to interact with char
s += 'A';       //and char literal
```

  - No:

```
"hello!" + " " + "bye!"; //literal not 'smart' enough to
                         //do concat with +
```

```
"hello!".substr(0);      //literal has no methods
```

Practice: C++ strings

```
int main(){
    string s = "hello,";
    s += "dolly!";
    s += s + "why," + "hello,dolly!";
    s.append("hello");
    s.append("dolly!");
    cout << s + '5' << endl;
    cout << "hello" + '5' << endl;
    return 0;
}
```

How many of these lines would NOT compile?

A. 0
B. 1
C. 2
D. 3
E. More than 3

When discussing:

- Make sure you agree not only on how many but which
- Talk about the "why" for each

Stanford library (3.7)

```
#include "strlib.h"
```

- Unlike the previous ones, these take the string as a parameter.

Function name	Description
`endsWith(str, suffix)` `startsWith(str, prefix)`	returns true if the given string begins or ends with the given prefix/suffix text

`integerToString(int)`
`realToString(double)`
`stringToInteger(str)`
`stringToReal(str)`
	returns a conversion between numbers and strings

`equalsIgnoreCase(s1, s2)`	true if s1 and s2 have same chars, ignoring casing
`toLowerCase(str)` `toUpperCase(str)`	returns an upper/lowercase version of a string
`trim(str)`	returns string with surrounding whitespace removed

```
if (startsWith(name, "Mr.")) {
    name += integerToString(age) + " years old";
}
```

ADTs

VECTOR, GRID, STACK

ADTs

- Programming language independent models of common containers
- They encompass not only the nature of the data, but ways of accessing it

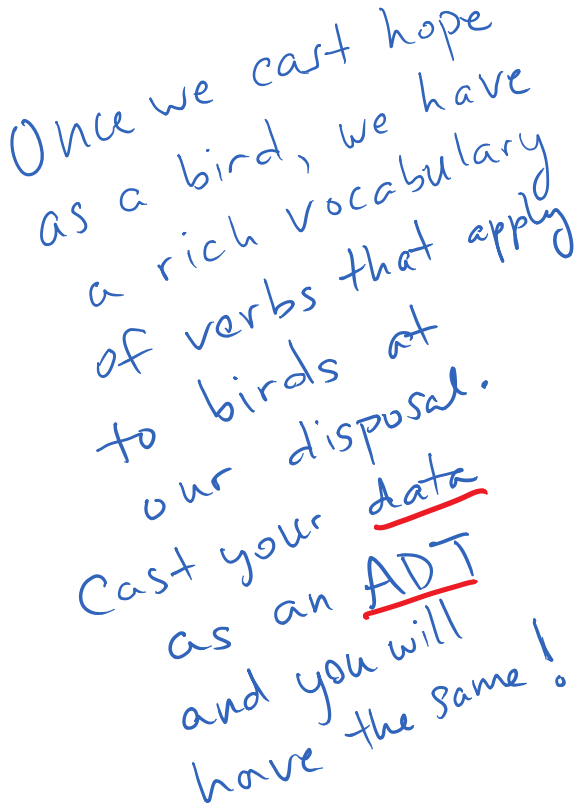
- They form a rich **vocabulary** of **nouns** and **verbs**, often drawing on analogies to make their use intuitive, and to give code written in them a certain **literary** quality

"Hope" is the thing with feathers
BY EMILY DICKENSON

“Hope” is the thing with feathers -
That perches in the soul -
And sings the tune without the words -
And never stops - at all -

And sweetest - in the Gale - is heard -
And sore must be the storm -
That could abash the little Bird
That kept so many warm -

I’ve heard it in the chillest land -
And on the strangest Sea -
Yet - never - in Extremity,
It asked a crumb - of me.



Vector

- ADT abstraction similar to an array
- Many languages have a version of this
  - (remember, ADTs are conceptual abstractions that are language-independent)

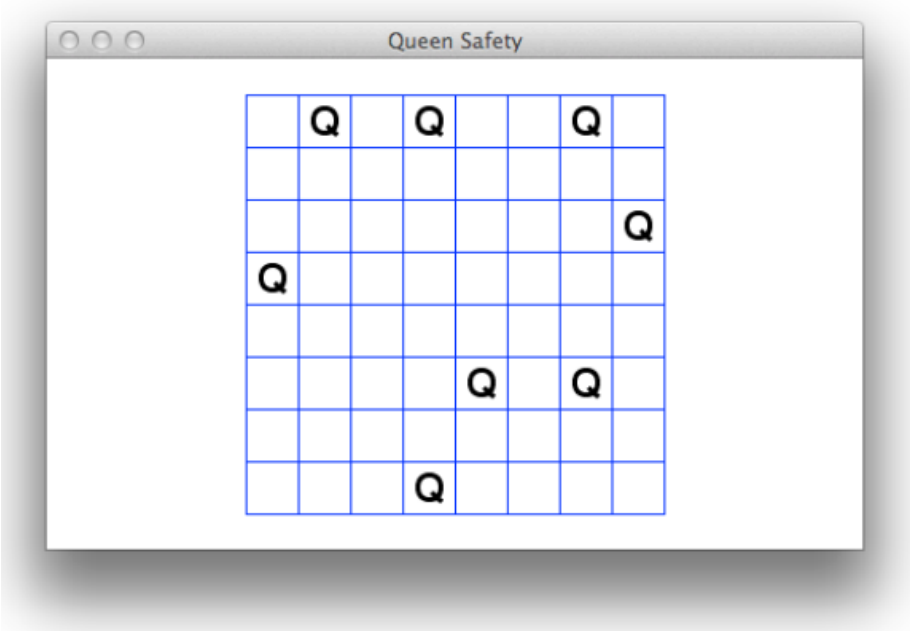
- In C++ we declare one like this:

```
Vector<string> lines;
```

- This syntax is called **template** syntax
  - Vectors can hold many things, but they all have to be the same type
  - The type goes in the < > after the class name Vector
    - `Vector<int> assignment3Scores;`
    - `Vector<double> measurementsData;`
    - `Vector<Vector<int>> allAssignmentScores;`

Code example: Queen safety

Download full code from the website to see an example of what we consider good coding style for your assignment!



Handy loop idiom: iterating over “neighbors” in a Grid

```
static bool isSafe(Grid<bool>& board, int row, int col) {
    for (int drow  = -1; drow  <= 1; drow++) {
        for (int dcol = -1; dcol <= 1; dcol++) {
            if (!isDirectionSafe(board, row, col, drow, dcol)) {
                return false;
            }
        }
    }
    return true;
}
```

R -1 C -1	R -1 C +0	R -1 C +1
R +0 C -1	R +0 C +0	R +0 C +1
R +1 C -1	R +1 C +0	R +1 C +1

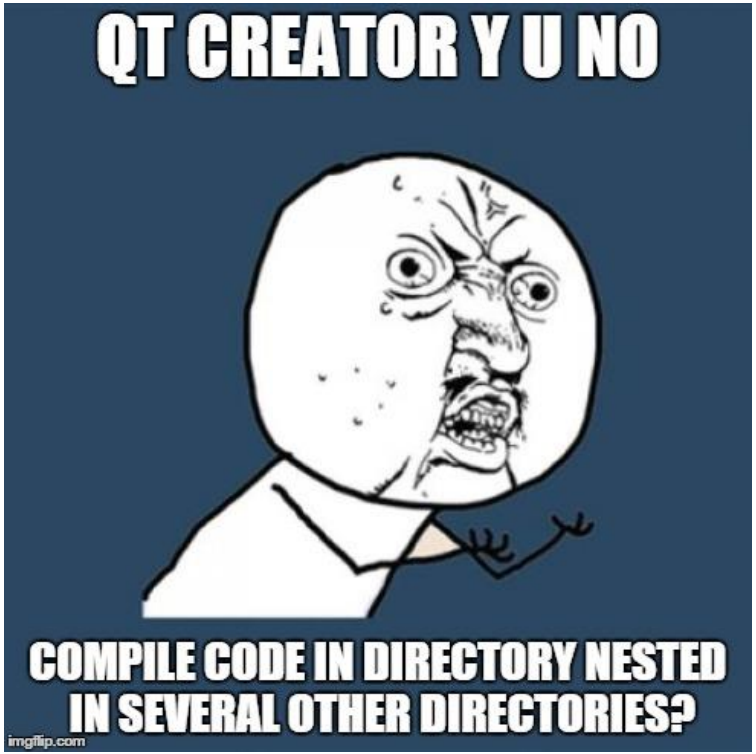
These nested for loops generate all the pairs in the cross product {-1,0,1} x {-1,0,1}, and we can add these as offsets to a (row,col) coordinate to generate all the neighbors (note: often want to test for and exclude the (0,0) offset, which is “self” not a neighbor)

QT Creator

A FEW WARNINGS & TIPS

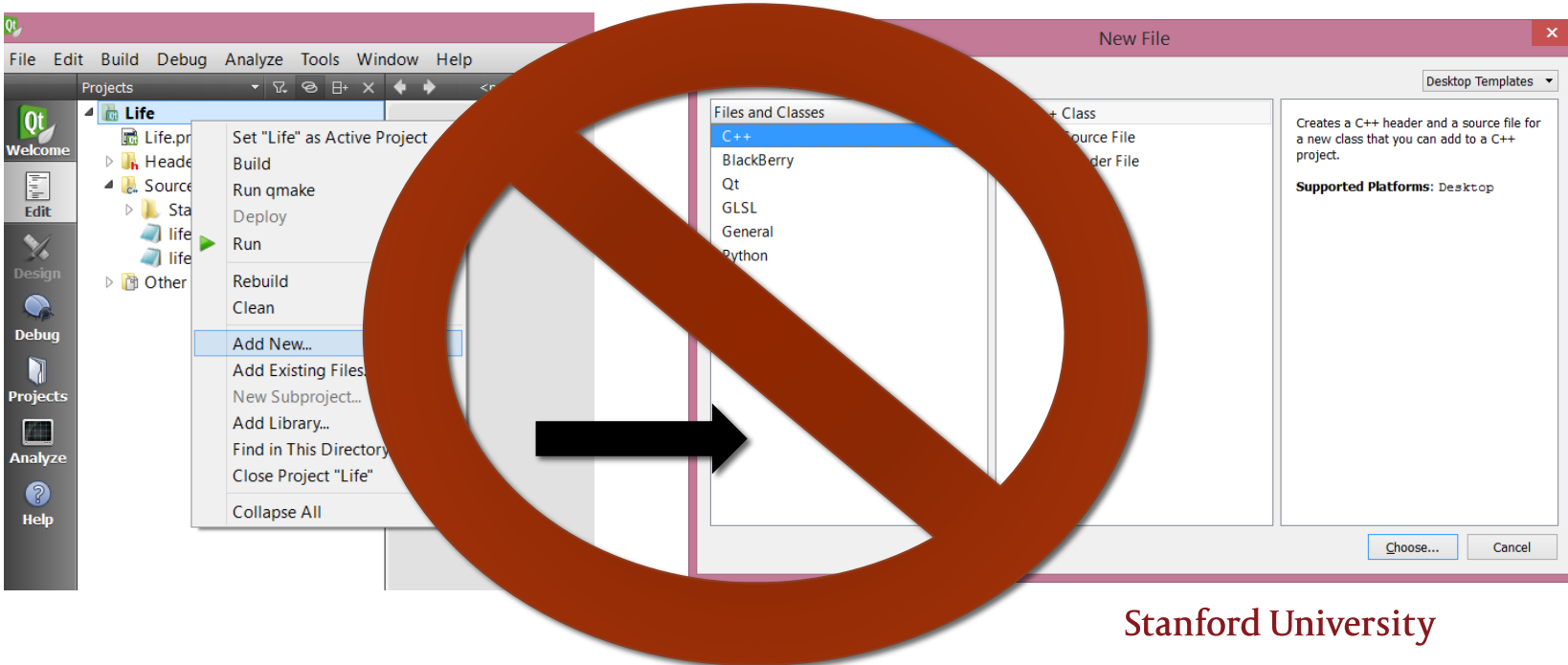
If your code doesn't compile and gives an error about unknown function Main() (note capital M):

- There is more than one reason this could arise, but the most common by far is that QT lost its ability to find the path to our Stanford library includes (which define a fake Main()), because you put your code in a directory/folder that is deeply nested on your drive.
- For example, C:\Documents and Settings\Documents\classes\Stanford\2015\Summer\CS106B\assignments\C++\Assignment1\...
- **You need to move the assignment directory closer to C:\ (or on mac, the root directory)**



If your code doesn't compile and gives you "multiple definition" errors for all the functions you have:

- This can arise if you add new files to your code inside QT creator using its handy "Add New…" feature.
- NOT HANDY! Do not use this feature. It breaks the .pro file.



If it's too late and you already used the "Add New" feature

1. QUIT: Close QT Creator
2. CLEAN UP BROKEN FILES: Delete the .pro and .pro.user files in your code directory; delete the entire build directory (this is an automatically created directory that appears alongside the directory where your code is, it's name is something like "build-simple-project-Desktop_Qt_5_2_0_MinGW_32bit-Debug")
3. GRAB REPLACEMENT: Download/unzip the assignment bundle again, and get a fresh copy of the project file (the one that ends in .pro) and put it in place of the one you deleted.
4. ALL BETTER! Re-open QT Creator (it will make a new .pro.user)