Programming Abstractions

CS106B

Cynthia Lee

Today's Topics

Introducing C++ from the Java Programmer's Perspective

- C++ strings and streams, continued
- Reference parameters

ADTs: Abstract Data Types

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

Warm-Up (CS106A review): Parameters

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

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

What is printed?

A. |-5| = 5
B. |-5| = -5
C. Other/none/more than one of the above

"Pass by reference"

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

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

What is printed?

A. **|-5| = 5**
B. |-5| = -5
C. Other/none/more than one of the above

"Pass by value"
(default behavior of parameters)

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

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

What is printed?

A. |-5| = 5

B. |-5| = -5

C. Other/none/more than one of the above

Often used when you would want to “return” several values from a function (but there is only one return value allowed)

```
#include "random.h"

void pickLotto(int& first, int& second, int& third) {
  first = randomInteger(0,10);
  second = randomInteger(0,10);
  third = randomInteger(0,10);
}

int main(){
  int lotto1 = 0;
  int lotto2 = 0;
  int lotto3 = 0;
  pickLotto(lotto1, lotto2, lotto3);
  cout << lotto1 << " " << lotto2 << " " << lotto3 << endl;
  return 0;
}
```

Pass by reference

benefits of reference parameters:

- a useful way to be able to in effect 'return' more than one value
- often used with large structures and objects, to avoid making bulky copies when passing (more on this in a minute)

downsides of reference parameters:

- hard to tell from call whether it is ref; can't tell if it will be changed
  - `foo(a, b, c);  // will foo change a, b, or c? don't know. ☹`
- can't pass a literal value to a ref parameter
  - `grow(39);  // error`

ADTs

Vector, Grid

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.

Once we say HOPE = BIRD…

…it now makes sense to say HOPE PERCHES

…or HOPE SINGS

We could have described those aspects of HOPE without the analogy, but casting our abstract idea of HOPE as a well-understood thing gave us access to similarly well-understood and evocative verbs, helping the reader more readily grasp our meaning.

The same thing happens in code with ADTs.

Vector

- ADT abstraction similar to an array
- Many languages have a version of this
  - Java `ArrayList`
  - (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;`

Grid ADT

Code example

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

```
static void neighbors(Grid<bool>& board, int row, int col) {
    for (int drow  = -1; drow  <= 1; drow++) {
        for (int dcol = -1; dcol <= 1; dcol++) {
            // do something with board[row + drow][col + dcol]
        }
    }
}
```

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)

CS106 Honor Code

Overview

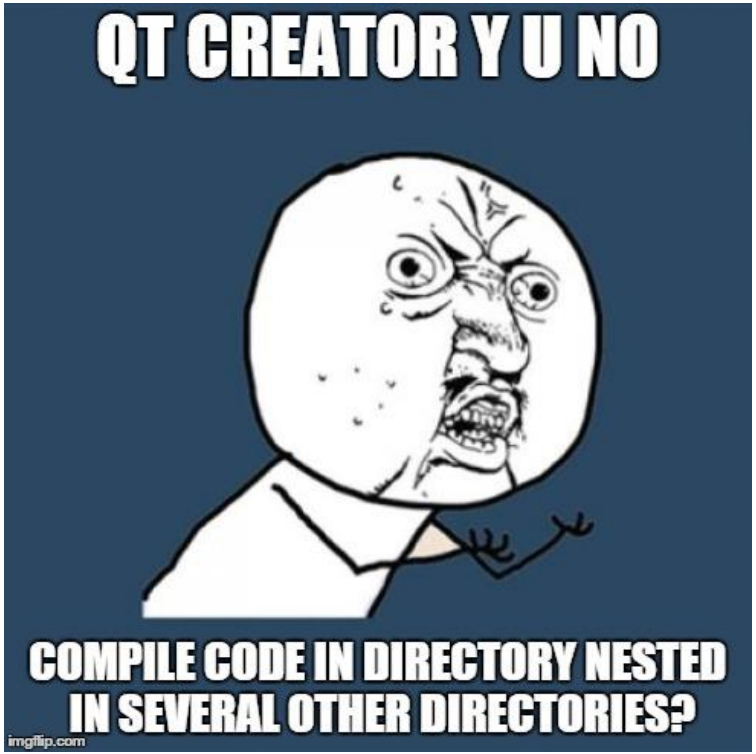
(Please read full document in detail.)

QT Creator

A few warnings & tips

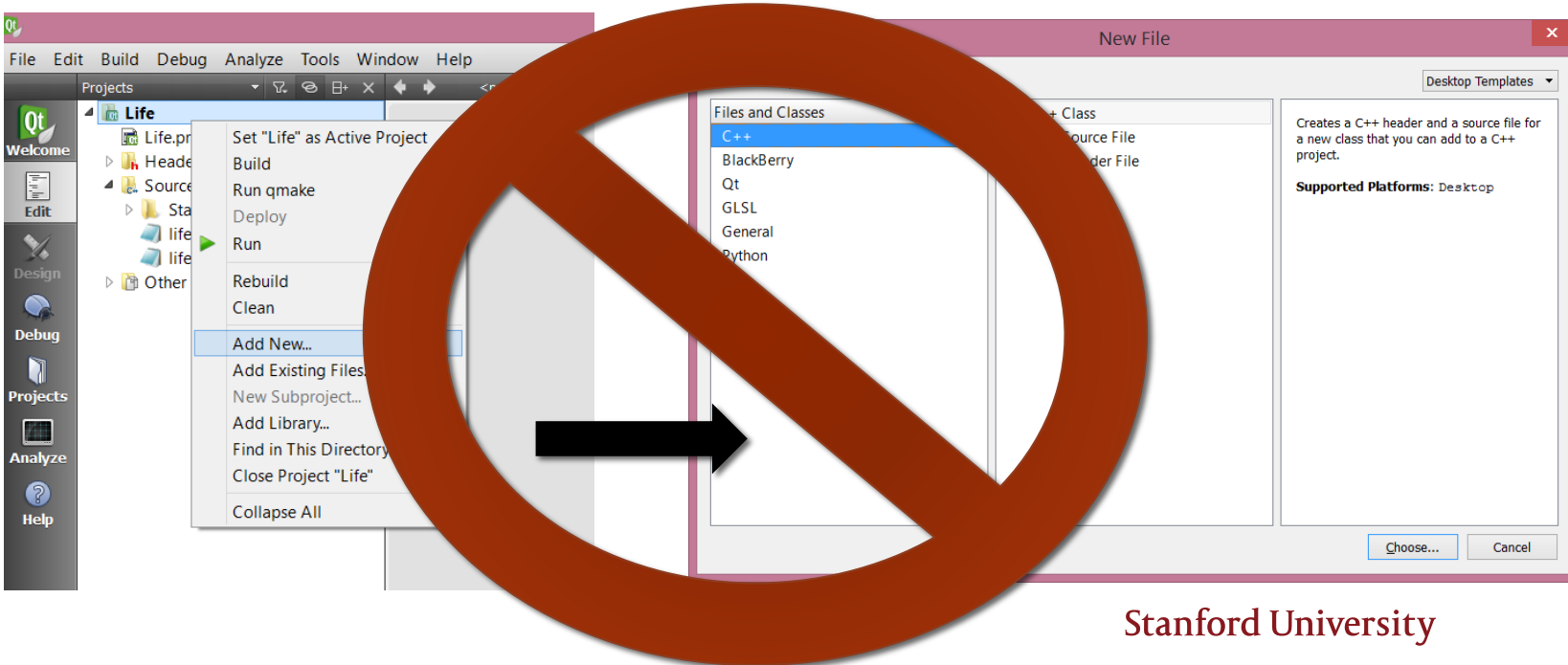
If your code doesn't compile and gives an error about unknown function Main() (note capital M):

- 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 special 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 (you don't need anything else in the new bundle).
4. ALL BETTER! Re-open QT Creator (it will make a new .pro.user)