Copy Constructor
When a non-primitive variable is passed/returned by value, a copy
must be made. As with a constructor, an automatic copy constructor
is provided for you if you choose not to define one:

All copy constructors will:

The automatic copy constructor:
1. 

2. 

To define a custom copy constructor:

Recall the joinCubes function:

Bringing Concepts Together:
How many times do our different joinCubes files call each constructor?

By Value  By Pointer  By Reference

Cube() 

Cube(double) 

Cube(const Cube &)

Cubes Unite!
Consider a Tower made of three Cubes:

Automatic Copy Constructor Behavior:
The behavior of the automatic copy constructor is to make a copy of
every variable. We can mimic this behavior in our Tower class:

...we refer to this as a ______________________ because:
Deep Copy via Custom Copy Constructor:
Alternatively, a custom copy constructor can perform a deep copy:

```
Tower.cpp
11 Tower::Tower(const Tower & other) {
12    // Deep copy cube_
13
14    // Deep copy ptr_
15
16    // Deep copy ref_
17
18
19
20
21
22
23 }
```

Destructor
The last and final member function called in the lifecycle of a class is the destructor.

Purpose of a destructor:

The automatic destructor:

1.
2.

Custom Destructor:

```
cs225/Cube.h
5    class Cube {
6        public:
7            Cube();    // default ctor
8            Cube(double length);    // 1-param ctor
9            Cube(const Cube & other);    // custom copy ctor
10            ~Cube();     // destructor, or dtor
11        ...
```

CS 225 – Things To Be Doing:

1. Exam 1 next Friday.
2. mp_intro, lab_intro, lab_debug all due Wednesday@ 11:59pm
3. MP2 released on Tuesday (start early for extra credit!)
4. Daily POTDs every M-F for daily extra credit!