Mug Lab 2

If you haven't done muglab 1 please do it now. It shouldn't take too long. This lab is also really easy and quick and at the end you will probably be able to complete the homework.

In this lab we will add 3 things to our mug class (they're in order of importance for the homework, it's all pretty easy but take some time look at it), and one thing of note. The first is we will overload cin and cout so you know how to do it for your homework (you will mimic this). The second is overloading operator "==" and operator "=", again so you have an idea of how this will work in the upcoming homework. We will then comment out the default constructor in both the .h and the .cpp file and instead add a default value to our other constructor. Finally, we discuss the static const called CAPACITY and update one function to reflect this static const.

Overloading cin and cout: We are going to mimic the homework and make both cin and cout friend functions. This differs from the book and the lecture ramblings which has cout be a member function and cin be a friend function. I'm hoping that this will make your life easier.

First note that we include iostream in our header file. And also notice that we include iostream and use namespace std in our .cpp file. **Remember to include these in your homework (or else you won't be able to overload cin and cout)!**

Now, Look at the cin overload in mug.h (friend std::istrea…), this actually allows you to do more than cin, you can also write out to files etc. Notice the syntax because you will have to use that in your assignments .h file (you can also see the pseudocode up top for this). Now go to mug.cpp and complete the implementation of this function (look at the comment for a hint).

Similarly look at the cout overload in mug.h (friend std::ostrea…) and get familiar with it because you're going to have something similar for point and line in your homework. Now go to mug.cpp and complete the implementation of this function.

Overloading Operatior = and ==:
You can see the prototypes for these in mug.h; in mug.cpp the implementation is blank. You should implement both of these member functions, use the comments as hints (keep in mind you have two mugs in this case, the mug that this function is a member function of, and the mug that is being passed in, in one case you want to compare the two mugs, and in the other case you want to set them equal.) Remember from lecture that "=" is a member function and "==" is a nonmember function. Look at the table on page 85 if you need a refresher.

Now comment out the default constructor in the .h and .cpp files and add a default argument (page 65 of the book if you don't remember how) to the prototype of the constructor that takes in a double (You do not add these to the implementation, just the prototype). Make the default value be 15 (no reason except, otherwise your output might not match the output at the end).

This says that if someone decides to make a mug without passing in a double (i.e. Mug myMug;) then it'll have an initial amountFilled value we give it. Otherwise if they do give a value it'll override our default value.

And you're done! Compile and run and get rid of all the bugs. You're output should look as follows:

***

```
I have two mugs. My first mug is filled to: 15. And my second mug is filled to: 32
Now let's empty our second mug
the second mug is empty right now
now let's fill up the second mug to 32.33
```

```
Now the second mug is filled to: 32.33
let's try to cout a whole mug
here's mug1's value 15and here's mug2 32.33
now let's set secondMug equal to first mug
now let's compare them with == and see if they are
First Mug Equals Second Mug
***
```

This+the book should give you all the tools necessary to do this first homework assignment (these are some of the trickier parts).

The final thing to note is that there is now a public variable in our .h file:
static const double CAPACITY=74.0.

The last thing you should do is think about how you might use this to change the implementation of the fillMug function? How about the constructor? You can do if you'd like but since it's not really in the homework, just think about it because maybe some sort of similar question will one day show up on a pop quiz or something….