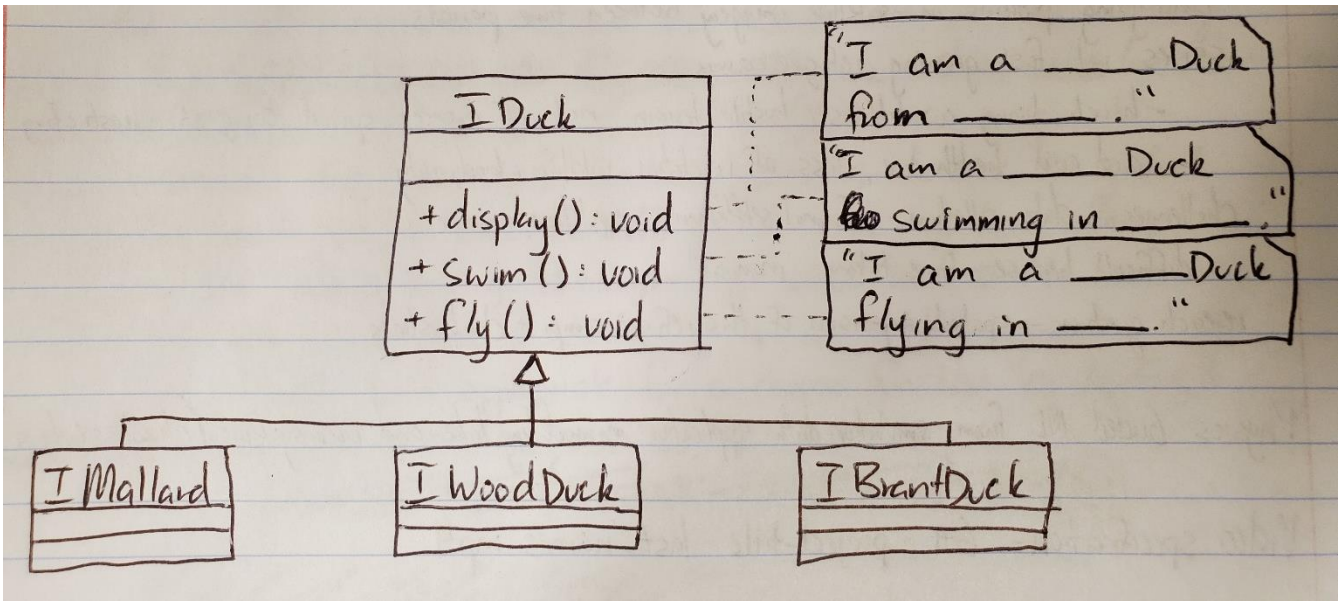
Hey everyone,

Here are the instructions for Workshop 4!

Step 1:

Create an interface called IDuck with the following functions: (see image below)

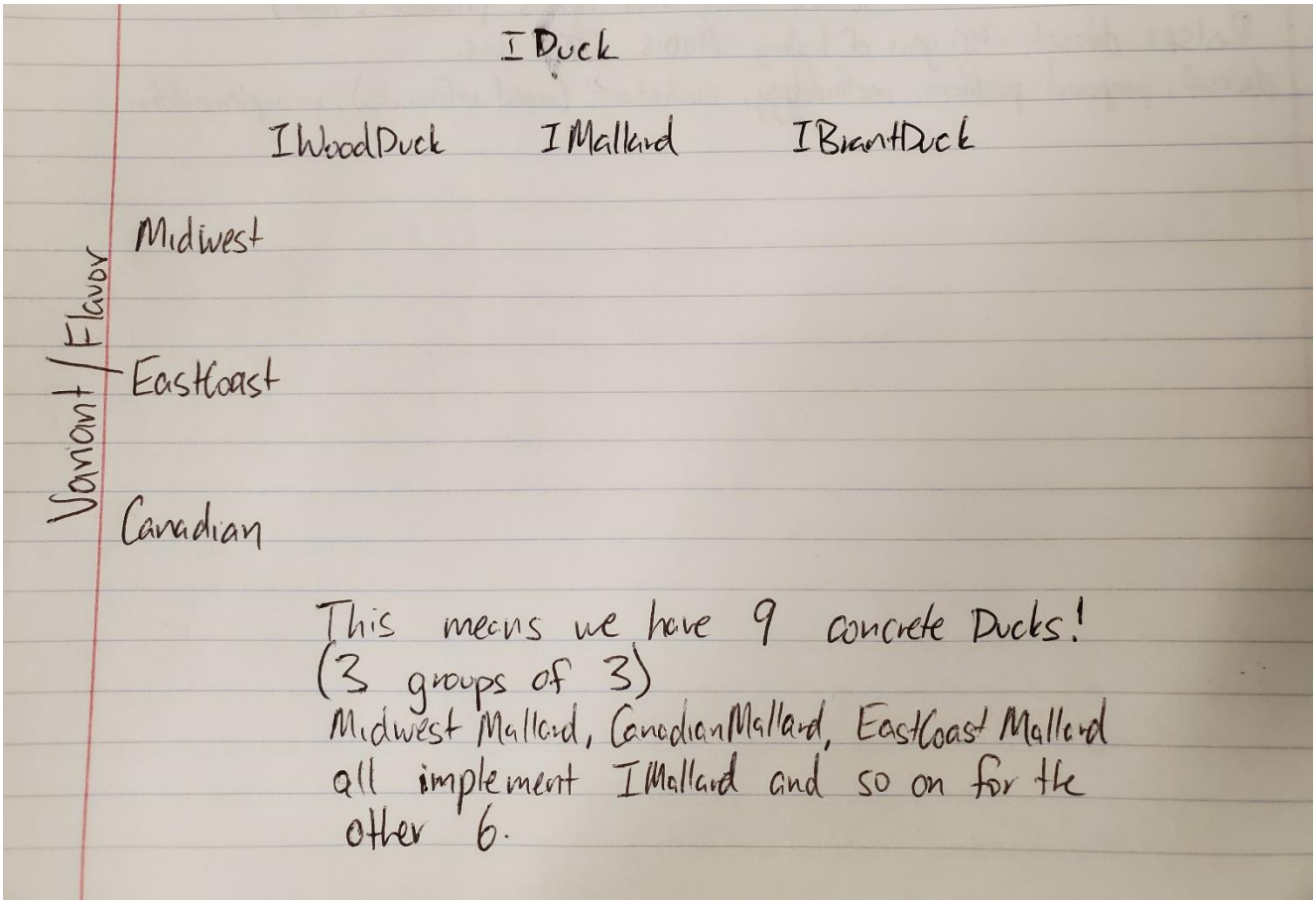
From there, make 3 subclasses/derived classes (all interfaces) called IMallard, IWoodDuck, and IBrantDuck.



I will address what those blanks are later in the instructions.

Step 2:

Let's come up with some variants/flavors of these ducks. I did my variants by geography. Some people could choose to do variants based on color. For example, RedMallard, BlueMallard, GreenMallard etc.



Now we have 3 products: IWoodDuck, IMallard, and IBrantDuck. We also have three variants/flavors: Midwest, EastCoast, and Canadian. This means that we will have 9 concrete classes (3*3). At the end of this step, you should have 4 interfaces and 9 concrete classes.

For the functions in each concrete class, here's how we will define them:

void MidwestMallard::display() { std::cout << "I am a Mallard from the Midwest." << std::endl; }

void MidwestMallard::swim() { std::cout << "I am a Mallard swimming in the Midwest." << std::endl; }

void MidwestMallard::fly() { std::cout << "I am a Mallard flying in the Midwest." << std::endl; }

etc.

Notice that anything that is underlined is subject to change depending on which of the 9 concrete classes it is in.

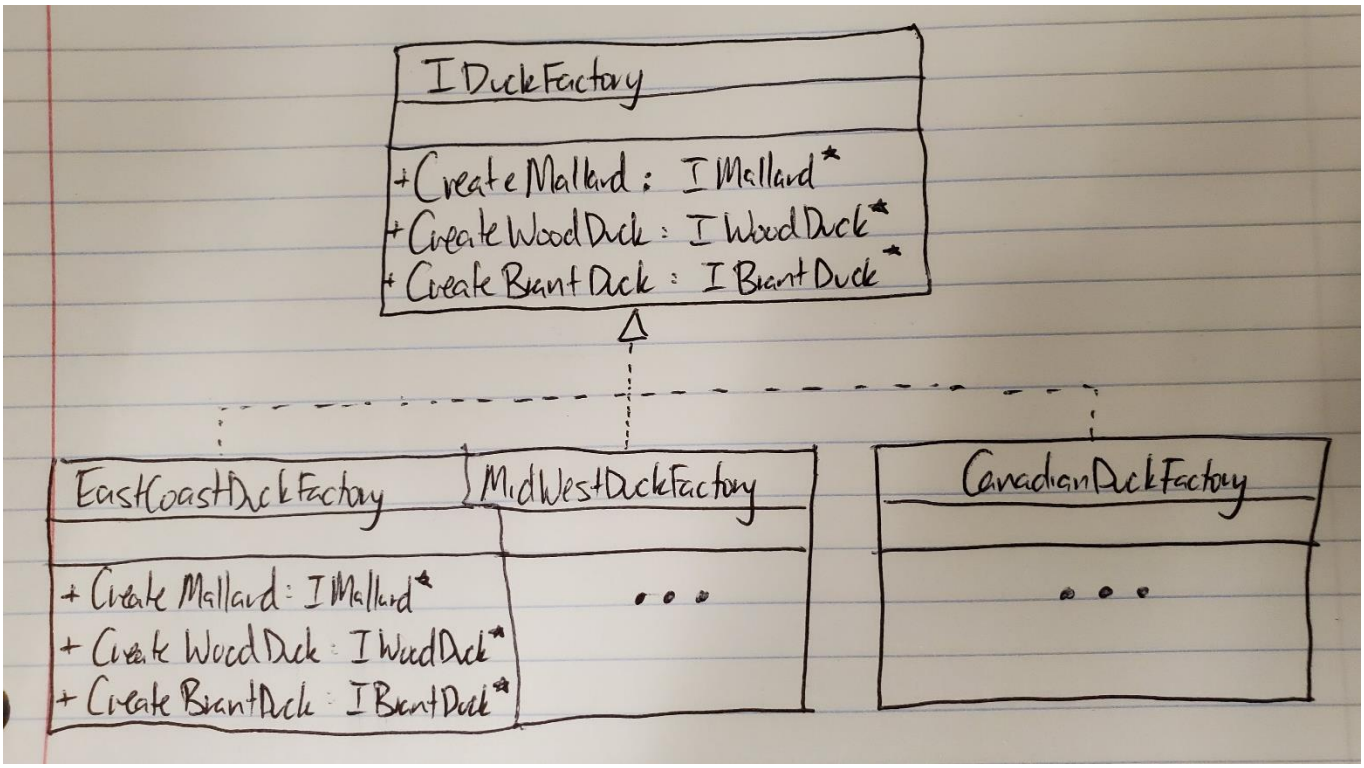
Step 3:

Let's start making the factories!

First, make an interface IDuckFactory with the following functions. From there, make 3 concrete factories, one for each flavor.

Hint:

IMallard* EastCoastDuckFactory::CreateMallard() { return new EastCoastMallard; }



Step 4:

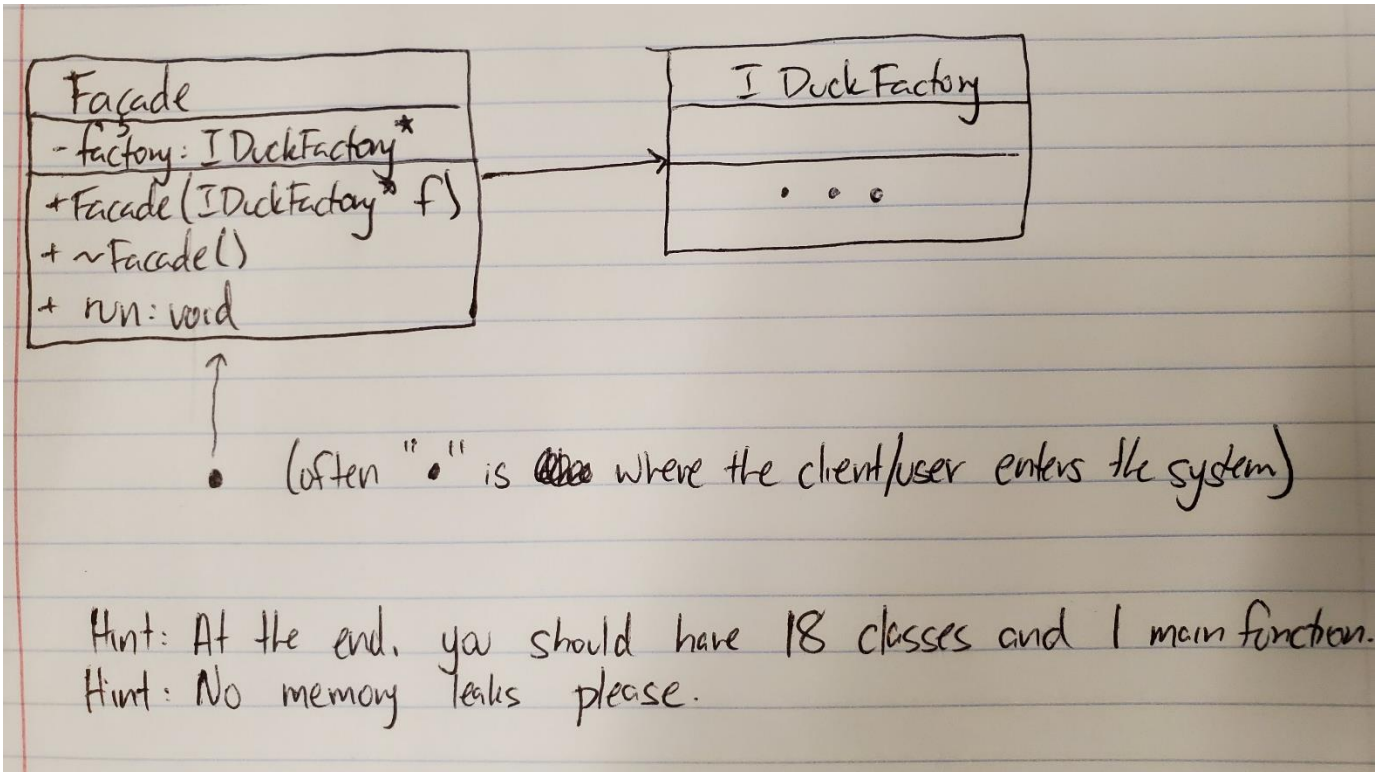
Now let's make our Façade. (Façade is a French word. Anytime that you have the cedilla (English) / cédille (French), you pronounce the c like an s. Think garçon (boy).)

In our Façade class, we will have one private member variable factory of type IDuckFactory*. And then we will have three functions (shown below).

Let's talk about the run function.

The run function will use the Façade's factory to instantiate all three types of duck then each duck will use all its own functions. (3 each, 9 total, ie duck1->display(), duck1->swim(), …, duck3->fly())

Make sure that you delete variables on the heap appropriately!



Step 5:

Now in your main function, create a concrete factory and pass it into your façade then call run from your façade. Make sure you don't have any memory leaks! (Refer to the Valgrind lab if you need help on this)

Submission:

Submission is on Canvas. If you worked with someone else, please say who in your submission. Also, it's individual submission. Submit one file with all 18 classes and main function. If you submit more than one file, you will automatically get a 0% on this workshop.

Grading:

0%: didn't follow instructions, very incomplete, forgot to submit on time, or doesn't compile

90%: memory leak(s)

100%: everything is correct, followed instructions, no memory leaks, and only one file submitted