Quest 14

300 EXP

CPSC121 SI

Erin: Hurry in! This is urgent!

Erin: Rex had delivered a student class to me yesterday before heading out onto his next quest this morning. Some time since he delivered his class, some code eating worms broke into my hard drive and ate parts of his code.

Erin: I managed to get rid of them just before they reached the main. I am going to need you to help me restore Rex's code. I'll give you a quick rundown on classes before you get started.

Erin: Classes are programmer-defined data types that are used to define objects.

Erin: Classes are made up of public and private members. Public members can be accessed by functions outside of the class (like the main). Meanwhile, private variables can only be accessed by functions that are members of the class. Private members exist to protect against inadvertent or deliberate data corruption.

Erin: Functions that belong to the class are called member functions. You can define your functions within the class declaration, which are referred to as inline member functions. The regular way of defining member functions is after the class declaration though.

Erin: Here is an example of how you would define a member function after the class declaration:

```
int Rect::getWidth()
{
   return width;
}
```

Erin: As you can see, it is a bit different from how you would normally define a function. With class functions you need to say what the return type is, the class the function belongs to, two colons (::) also known as a scope resolution operator, function name, then the function body.

Erin: In addition to member functions, there are also functions called constructors. These member functions have the same name as the class and are used to initialize the data members of the class. The three types of constructors are default, overloaded, and copy.

Erin: The default constructor is the constructor that takes no parameters. The overloaded constructors are any number of constructors that have a unique set of parameters. Lastly, the copy constructor is when you send an object of the same class to initialize your class with.

Erin: Here are a few examples of what those constructors would look like:

Rect(); //Default constructor.
Rect(int,int); //Overloaded constructor that takes two parameters.
Rect(Rect); //Copy constructor which takes in a copy of the passed object.
Rect(const Rect&); //Copy constructor that takes in the actual passed object, but cannot change it.

Erin: The last thing you need to know are Destructors. These are similar to Constructors because they hold the same name as the class as well. The difference is that the Destructor is preceded by a Tilde (~).

~Rect();

Erin: Fortunately I managed to save the destructor in Rex's code, so you do not need to refill that one. The destructor is automatically called when an object is destroyed, it has no return type, and it takes no arguments. Oh and only one destructor is allowed per class.

Erin: In case you were wondering, the destructor is used to clean up when an object is no longer needed. The destructor of classes in the main will be called when you reach the end of the main. In other words, at the return 0, your destructors will be called. If you have a system("pause") before the return, then the destructor will not have been called yet.

Erin: Okay, I think that was enough of a run down to set you on your way. Please fill in the blank areas of the empty student class that should be found in the Quest Items/Quest_14 folder. You should not need to touch the main at all. If you fill in the class member functions correctly, the main should run as is. Good Luck.