Intro to C++

And Some Tools
9-2-2003

Opening Discussion

- Have any questions come up since last class? Have you had a chance to look over the project description? If so do you have any other thoughts on it?
- I simplified some of the submission details on Saturday. Basically I need your submission to have a makefile and two directories, one with Java code and one with C++ code. What is in those directories is up to you.

I/O in C++

- We saw last time that standard input and output in C++ use the std::cin and std::cout objects. These are streams and can be found in the iostream library.
- They have overloaded the bit shifting operators (<< and >>) to move data to and from streams.
- There are also ifstreams and ofstreams that can do I/O with files.
- The C libraries still work and we will use those for direct access binary files.

Memory in C++

- Objects can be on the stack in C++ which is ideal if possible. "Automatically managed."
- Like Java you will use the new operator to get dynamic memory in Java (don't use malloc as you did in C).
- Unlike Java, C++ has no garbage collection so you must call delete on everything you get with new.
- For arrays you use pointer=new type[#]; and delete[] pointer;. No shortcut for multidimensional arrays.

Class Basics

- Like Java, C++ is a class based OOPL.
- It also provides data hiding with public, private, and protected visibility options.
- Methods and members are grouped so you don't specify the visibility for each elements.
- Other flags have to be put on each elements: static, const, virtual, etc.

Constructors

- As with Java, when a new object is created a constructor for it is called. This happens either with new or when a stack variable of the type is declared.
- Constructors have the same name as the class and can be overloaded with different arguments. (All functions/methods can be given default values for arguments in C++.)
- For greater efficiency you can also use initializer lists.

Destructors

- Because C++ lacks garbage collection, classes also have a destructor. This is a method that has a '~' in front of the class name. It takes no arguments.
- This method is called any time delete is called on an object, or when a stack object passes out of scope. It should free any memory the object allocated and do other related cleanup.

Deep Copy vs. Shallow Copy

- A shallow copy doesn't create new copies of things that are pointed to. A deep copy does. In Java "object assignment" really copied a reference so this wasn't really an issue. In C++ anything passed or returned by value does a copy so you have to think about this.
- If a class has pointers in it you typically need to include methods to handle deep copying.

Copy Constructors

- One of the methods you need to write for this is a copy constructor. Like in Java this is a constructor that takes a reference to an object of that type.
- Anytime you pass an object by reference the copy constructor of that object will be called.
- By default C++ creates a copy constructor that does a shallow copy.

Overloading Assignment Operators

- We'll talk about operator overloading in more general terms later, but it is possible to C++ to make almost every operator do something special for a given type. This includes the assignment operator.
- Here again if a class includes pointers you will want to overload this operator. By default C++ gives you one that does a shallow copy.

Header Files

- In Java everything related to a class went into a file that bore the name of that class. In C++ the file names don't really matter. However, you typically declare a class in a header file (ends with .h) and define the methods for it in a source file (ends with .cpp).
- To include one of your header files you put double quotes around the file name at the #include.

Make and Makefiles

- We use header files so we can do partial compilation. We compile different .cpp files to object files (.o) then link the object files to create executables. Make is a tool that helps with this.
- Make is very powerful, but we only need the basics. You define various rules (often a filename created with that rule), say what files that rule depends on, and then say what happens when the rule is executed. A rule is only executed if the dependencies are newer than the specified file.

Installing Cygwin

- If you install Cygwin on your machine make sure to include the development packages. Those include make, gcc, and g++ compilers.
- You can run the Cygwin install multiple times and select different packages each time (or unselect ones you don't need to get back disk space).

Using Eclipse

- If you run Eclipse normally under Windows it won't find make or g++. That's fine for Java development but will cause problems for C/C++.
- To fix this, run Eclipse from inside Cygwin. You can get to your normal Windows directory structure by going to /cygdrive. If you put Eclipse in Program Files then go to /cygdrive/c/Program\ Files/eclipse/ and you can run eclipse.exe.
- Need to install CDE for Eclipse and have a makefile in the project directory with "all" rule.

Minute Essay

- Write C++ code to allocate an array of num ints input them and print them in reverse order.
- I will be leaving town right after class and won't get back until late Friday. My e-mail access during that time will be sketchy.
- Start working a bit on assignment #1 even without a partner. At least do design and think about approach.