Week Eight

Max Olivier

March 7, 2009

Readings
Please read Chapter 14 and Chapter 15 through page 370 in Overland. You should stop at the section “Deep” Copying and the Copy Constructor.
Main Ideas

- One of the most important aspects of C++ is that, unlike languages such as Python and Java, there is no garbage collection. This can be liberating because you suddenly have more control over allocating the computer’s memory, but also devastating since any memory that you allocate and forget to deallocate remains unusable even after the program stops running. Thus, since there will inevitably be times when you need to set aside dynamic memory that you want to remain for the entire life of a program (such as in the linked list you will build for this week), it is extremely important to know how to properly deallocate memory in C++. This week is devoted to giving you practice with this memory allocation and deallocation. Specifically you should know:
- How to allocate and deallocate dynamic memory in C++ using the new and delete keywords.
- Why it is EXTREMELY important to deallocate any dynamic memory that is allocated.
- Pointer dereference syntax in C++.
- Why class destructors are important and how to use them.
- How to make a linked list in C++.
- For the last point, I would suggest you briefly review some of the algorithms you used for linked list functions in data structures to make this aspect of the week go faster.

Exercise 1 (Exercises from the Reading)
All the exercises from the reading in Overland except 15.1.2.

Exercise 2 (Cryptololgy Program)
This week’s cryptology program is a little different. Instead of adding to your ciphers as you have been for several weeks, you are going to write a linked list in C++. I know this may not seem related, but you will see how we use it once the final project starts.

Because writing a Linked List in C++ can be tough (it took your course writer upwards of 8 hours), start by looking at the program LLskeleton.cpp. Read through the comments and write the functions it asks you to write. Two notes before you begin this quest though. First, at some point in making your Linked List, you will likely get the message *Bus Error* with no indication as to what happened or what line is causing he problem. If you get this it means you have a memory error or NULL pointer exception of some kind. Because the compiler often doesn't catch this, the department has a program called Valgrind installed on the lab computers. So, if you are getting errors like that go to the lab computer and type:

```
g++ -g -o prog programName.cpp
```

where programName is the name of the .cpp file yoiu have saved, and prog is the name of the executable file you are going to run. As you can see the only difference between this and usual compiling is the -g, which indicates to add the debugging info.

The second note is to make sure that at the end of the program every new pointer that is created is destroyed since otherwise you will have memory leaks in a language without garbage collection like C++. Valgrind should help catch some of these, but ultimately it depends on your code.