Notes 1/14/11

So far we have been working exclusively with arrays. Downsides of arrays are that they are not expandable, if you delete something there is a gap, and if you want to add an object in the middle you have to shuffle everything around, all of which are annoying. The advantages of arrays are simplicity and quickness, but let’s talk about another way to arrange objects called a *linked list*.

A linked list is a class where every object has a pointer in it, called a *next* pointer. The first pointer is called the *head* pointer. You can store values in the classes and have them link to each other. Advantages and disadvantages are reversed from classes; it is very easy to add and subtract elements, they are easy to expand, but they are complicated and it is not easy to reference a specific element.

If you are trying to add an element a:

Obj \*curr; //current pointer  
last = head;  
curr = head; //points to the first object  
while (curr && curr->next->val > a->val) //while the current pointer != null and the value of current is less than the value of a  
{  
 last = curr;  
 curr = curr->next;  
}   
last->next = a;  
a->next = curr;

This is one way to implement it, but there are many. You really have to worry about the first and last element of the list. A lot of times people just use the first element as junk because it removes a lot of debugging if you can ignore it. There is also what is known as a *doubly-linked list* where the elements have links both to the next element and the previous element. One step further is the *circular-linked list* where the double linked list also links the last element to the first and the first to the last.

Until now we do not know how to use the debugger, but it is very useful. You can use breakpoints to stop the program at a certain point by clicking in the left margin, making a red dot. The program will stop at this point and you can view the values of all the local variables.

Step into: call the next function  
Step over: skip the next function  
Step out of: go back out of the current function  
  
If you step into, then step out of a line with multiple function calls, if you then step into it again you will access the second function.

Windows: Autos is variables that are currently executing, Locals is all variables locally defined, Watch is your choice of what variables you want to look at. Watch lets you look at specific elements of arrays – and change them if you want.

You can add conditional breakpoints for conditions when you want the program to stop. One of the problems with this is that it can be extremely slow because it breaks and checks every single time it passes the conditional breakpoint.