


# Instructions for Module 14, Exercise #1 (traverse a linked list)



In this exercise, you will fill in the details for a creating and traversing a linked list of student names..

First, download the [Traverse.cpp \(https://miracosta.instructure.com/courses/31330/files/7025687?wrap=1\)](https://miracosta.instructure.com/courses/31330/files/7025687?wrap=1)   [\(https://miracosta.instructure.com/courses/31330/files/7025687/download?download\\_frd=1\)](https://miracosta.instructure.com/courses/31330/files/7025687/download?download_frd=1) program into a C++ project. This program already has the **LinkedList** structure from the lecture already defined.

The **main** function is partially completed. You should be able to compile and run it now, but all you will see is a report showing the values of the head node of an empty linked list.

In **main** there is also an array of student names. These names are in sorted order. The intention of this lab is to add those names into a linked list, then traverse the list so that the names of the students are listed in the *same* order that they appear in the array.

Since we know only how to add to the front of a list, we'll need to add the names to the list in the reverse order of the names as they appear in the list. To do this, we'll just create an **stringstream** object with the names in reverse order, then peel them off of the stream using a stream extraction operator and add them to the front of the list! (This will allow us to exercise our use of stringstream, and simulate reading data from a file.)

Under the comment which starts "Add the items from the array...", add the following code.

```
int tsize = sizeof(array) / sizeof(*array);  
stringstream iostr;  
for(int ii = size - 1; ii >= 0; ii--){  
    iostr << array[ii] << endl;  
}
```

This code simply reads each name in the array (starting at the end working to the front) and adds it into a **stringstream** object.

Next, we can simulate reading names from a file (except we'll use the **stringstream** object instead of an **fstream** object) and add each name in the stream to the *front* of the linked list. Add

the following code after the declaration and assignment statement of the **head** node variable.

```
stringnamee;;  
whilee(iostr>>>name)){(  
    headd==newwLinkedListNode(name,,head));;  
    cout << "added " << name << endl ;  
}}}
```

Make sure that your program still compiles. (It still will print only the head node information, though.)

Finally add the following code which traverses the list, printing the name of each student as it reaches their node in the linked list. Add this code after the code which prints the information for the head node:

```
///Information for each node in the linked list  
whilee(ptrr!=nullptr)){(  
    cout<<<"    "<<<setw(10))<<<leftt<<<ptr->dataa  
        <<<setw(17))<<<rightt<<<ptrr  
        <<<setw(12))<<<ptr->nextt<<<endl;;  
    ptrr==ptr->nextt;;  
}?  
coutt<<<endl;;
```

Compile and run. The complete list of names should now print in the same order they appear in the array (sorted by name)!

Notice that the "next" pointer in each node points to the address of the next node in the list, and that the "next" node at the end of the list is 0 (or nullptr).