# CSC 232: Data Structures and Algorithms

**Assignment 4: Singly Linked List**

**Due: See Blackboard**

*Submit all programs through BB by midnight on the due date. Email your programs to me* as a last resort *if you experience problems with BB.*

The purpose of this assignment is to:

* Implement a formal data structure using a C++ class and a friend class.
* Practice use of class templates.
* Learn the use of pointers.
* Use good memory management techniques.

**Customer Requirements**

You are a motivated software developer and budding entrepreneur advertising for part-time work on the Internet. A small business called Widgets-R-Us has decided to give you an opportunity, but you will have to prove yourself first so they are only giving you the initial part of their overall project. Their hope is that you will use some Computer Science techniques to help them simplify their product.

Widgets-R-Us manufactures a small PCB (printed circuit board), called the WRU-1 that they program with one of three options:

* a simple computer game, Tic-Tac-Toe
* a four-function calculator, 4Calc
* a fun/whimsical program, 8-Ball Oracle

Widgets-R-Us sells the WRU-1 to other companies that put them into cheap plastic enclosures that can be used in a variety of ways, like keychain decorations, “the prize” in cereal boxes, party favors, or on the tips of pencils. Recently, they started having trouble getting new customers because of the limited number of programs that run on the WRU-1.

The rudimentary operating system they have been using allocates memory sequentially and is based on the array. In order to expand into the types of programs that their customers are requesting, they need a more flexible allocation scheme. If they can increase the number of programs that run on their PCB, they can increase their potential market.

The engineers have come up with a prototype, the WRU-2a. They modified the old firmware to return a random memory address rather than a sequential address. What they need from you is a data structure that can take advantage of this random memory function. In the prototype, the data stored at each memory location is a game instruction encoded as a floating point number. This is likely to change in the future.

**Assignment Requirements**

* Implement a singly linked list (SLL) abstract data type using the book’s implementation (Section 3.2.4) with two classes. Call the first class **Node** and the second class **SLinkedList**. Both classes will be templated using the typename **E** which stands for element, the field that holds the data stored in the list. Use of the above names is required, as is using the exact same method names used in the book.
* Implement a copy constructor and an assignment constructor for both classes.
* Instantiate a list of type **float**. Generate 10 random numbers between 0.01 and 9.99. Use them for the element field and call **addFront()** to add 10 new nodes to your list. These random numbers must not have more than two digits beyond the decimal or there will be a buffer overflow in the WRU-2a.
* Overload the **<<** operator to print the list like this: {4.31, 6.22, 9.15}. Do not include a trailing **std::endl** in this method.
* Add a method to sort the list using the insertion sort algorithm in DSA 3.1.2 Follow the algorithm as closely as possible. To achieve this, it will be necessary to have two overloads for the subscript operator [ ], one that is const like in Lab5 and one that is not const. To get full credit for this part of the assignment, your program must structurally move nodes around in the list. Half credit will be awarded if your program does not move nodes but only modifies the element field of each node to accomplish the sort.
* Using the Lab5 code as an example (or create a copy and modify) will likely be helpful in making sure you have correct template syntax in your classes and methods.
* Use good coding style as documented in the Coding Style Cheat Sheet (in the Resources folder in BB). Include the comment template at the top of your program.
* You may use only language features discussed in class or presented in the book up to the date the assignment is due. Your submission must be your own work. You may not utilize any code outside of that provided in class or in the book, and you may not post any provided code on publicly accessible websites. Submit only what is requested.