**Lab 11 - IntList intro**

### Goals

By the end of this lab you should:

* have a good head start on the IntList assignment
* have fully tested the functions required by this lab
* be confident these functions work correctly

**Collaboration policy:**

For this assignment, collaboration **IS ALLOWED**. You will be working on the first stages of a larger programming assignment.

## IntList Assignment Specifications:

You will begin the implementation of the singly-linked list I demonstrated in lecture.

You are required to come up with a single header file (IntList.h) that declares and implements the IntNode class (just copy it exactly as it is below) as well as declares the IntList Class interface only. You are also required to come up with a separate implementation file (IntList.cpp) that implements the member functions of the IntList class. While developing your IntList class you must write your own test harness (main function). Never implement more than 1 or 2 member functions without fully testing them with your own test harness.

## IntNode class

I am providing the IntNode class you are **required** to use. Place this class definition within the IntList.h file exactly as is. Make sure you place it above the definition of your IntList class. Notice that you will not code an implementation file for the IntNode class. The IntNode constructor has been defined inline (within the class declaration). Do not write any other functions for the IntNode class. Use as is.

struct IntNode  
{  
 int data;  
 IntNode \*next;  
 IntNode( int data ) : data(data), next(0) {}  
};

## IntList class

### Encapsulated (Private) Data Fields

* head: IntNode \*
* tail: IntNode \*

### Public Interface (Public Member Functions)

* IntList()
* ~IntList()
* void display() const
* void push\_front( int value )
* void pop\_front()

Constructor and Destructor

#### IntList() - the default constructor

Initialize an empty list.

#### ~IntList()

This function should deallocate all remaining dynamically allocated memory (all remaining IntNodes).

### Accessors

#### void display() const

This function displays to a single line all of the int values stored in the list, each separated by a space. It should **NOT** output a newline or space at the end.

### Mutators

#### void push\_front( int value )

This function inserts a data value (within a new node) at the front end of the list.

#### void pop\_front()

This function removes the value (actually removes the node that contains the value) at the front end of the list. Do nothing if the list is already empty. In other words, do not call the exit function in this function as we did with the IntVector's pop\_front.

## What to Submit

Submit in Canvas the following files (**case sensitive**):

* main.cpp (test harness)
* IntList.h
* IntList.cpp