CSIS 211 – Data Structures
Spring 2017
Lab #3 – Due Date: 04/16/17 at 11:55 pm
Remember to submit ALL files (.cpp,.h etc.) on Moodle – NO EMAILS – no exceptions
You may work in a group of 1-3 for this assignment
EACH group member should submit just to make sure I don't miss any while grading
IMPORTANT! – put EACH group member's name at the TOP of the CODE in comments
If you are a group of 1 put your name only at the top
Make sure and use good design techniques
This will be worth 50 points

For all Programs:
Do NOT forget to put ALL your groups names at the TOP of EACH FILE in comments.
Example:
//LAB 3
//Group: Kristina Shroyer, John Smith, Jane Doe
Do this EVEN if you have a group of 1
Since this is very important

Lab #3 – Pointers, Dynamic Memory Allocation, Linked Bags

This is an object oriented program – you should use the following files from the Chapter 4 Example programs as a starting point for this assignment: LinkedBag.h, BagInterface.h, Node.h

For this assignment you are going to make a modification to the LinkedBag, add some new function templates to the LinkedBag and then test out your new LinkedBag in a client program. Note BagInterface.h and Node.h should NOT change:

1. **First modify the add method of LinkedBag (you already did this in HW #5 so you are just coding it in C++ if you used pseudocode or adding your C++ code to the LinkedBag class:**
    1. Modify the add function (a function template) of the LinkedBag class so that the new node is inserted at the end of the linked chain instead of at the beginning
2. **Add a union method to the LinkedBag**
    - The union of two bags is a new bag containing the combined contents of the two original bags. Design and implement a union member function for the LinkedBag class. Don't forget all member functions of a template class must be function templates.
    - Note the union may contain duplicate items. For example if object x occurs in bag1 five times and occurs in bag2 two times, object x occurs in the union of bag1 and bag2 seven times.
3. **Add an intersection method to the LinkedBag**
    - The intersection of two bags ins a new bag containing the entries that occur in both of the original bags. So the intersection of bag1 that contains a 4 and 5 and bag2 that contains a 5 and 6 would be a new bag containing just 5. (This example assumes the bags are bags of ints)
    - Note that the intersection of two bags might contain duplicate items. For example if object x occurs in bag1 five times and occurs in bag2 two times the bag that is the intersection of these two bags should contain the item two times.

Finally create a client program to test the methods you added to LinkedBag

Make sure and test the add method, the union method and the intsersection method on various lists. You will need to use toVector to print out results and test and you should think about using remove and clear just to make sure union and intersection still work when an entry is removed and when the list is empty.