CS 10C Programming Concepts and Methodologies 2

Skip to Main Content

Assignment 3: Link-Based Implementations

No documentation is required in this assignment.

Assignment 3.1 [20 points]

Do Programming Problem 2 at the end of chapter 4.

Similarly to what you did with the array-based implementation, you'll start with the LinkedBag code from the links below and make the needed edits to change it to a link-based Set.

The vast majority of your code will be copied directly from the LinkedBag class from the links below. You just need to make minor adjustments to reflect the fact that duplicate elements are not allowed. Also, we won't have a frequencyOf() member function.

Your LinkedSet class must be derived from a "SetInterface" abstract class. You should be able to reuse your SetInterface class from the last assignment.

You must provide the big-3. If you are not familiar with the term "big-3," please take a few minutes to review lesson 17. The LinkedBag from the links below provides a copy constructor but does not provide an overloaded assignment operator, so you will need to add that to your class.

I strongly suggest the following as a review of the function of the copy constructor: Compile and run the LinkedBag code provided below. Then comment out the copy constructor from both LinkedBag.h and LinkedBag.cpp, and observe what happens. Study the copyConstructorTester() function to make sure you understand what is happening.

Here's my suggestion about the big-3. Change the given copy constructor into a function named "copy()". This won't require any changes to the code inside the function. Then create a new copy constructor that does nothing except call the copy() function. Once you have tested that carefully, then you can call copy() from the assignment operator to do the work of making the copy. (The assignment operator will also need to call clear(), check for self-assignment, and return the appropriate object.)

A subtle point about the copy() function I'm suggesting: It should be a private function, because it won't be appropriate for the client to call this function, because a copy() function intended for client use would need to deallocate the already existing LinkedBag object that is being copied into.

Here is the source code that you are to use as a starting point:

- **LinkedBag.cpp**
- **LinkedBag.h**
- **Node.h**
- **Node.cpp**
- **bagtester.cpp**

Assignment 3.2 [25 points]

Add member functions setUnion(), setIntersection(), and setDifference().

The functions should all return the resulting LinkedSet object. We won't need to throw an exception in the setUnion() function, because the Set will have unlimited capacity.

You may use contains() and the Node class accessors and mutators, but, other than those, **do not use any explicit function calls in your definitions of these three functions!** The operations must be coded without explicitly calling any other functions to help. The practice manipulating the pointers directly will be extremely valuable.

You may also use calls to constructors and the assignment operator. (The word "explicitly" in the above paragraph is intended to allow for you to call constructors and the assignment operator, since those are not explicit function calls.)

Submit Your Work

Name your source code files SetInterface.h,LinkedSet.h, and LinkedSet.cpp. Don't submit the node class files, the client program that you used for testing, or your output. Use the Assignment Submission link to submit the three files. When you submit your assignment, on the Canvas assignment submission page there will be a text field in which you can add a note to me (called a "comment", but don't confuse it with a C++ comment). In this "comments" section of the submission page let me know whether the class works as required.