CS 445 Lab 3: Array vs. linked implementations

Introduction

In this lab, you explore the implementation of the ADT bag using arrays and linked chains. You will override the `equals` method so that it will determine if two bags are equal based on their contents.

The primary goal of this lab is to practice solving the problems that arise when developing implementations of data structures (alternately array- and linked-based). A secondary goal is to exercise good code reuse practices, as existing Bag methods may be useful in implementing the one you will be writing in the lab.

As we have seen in lecture, a bag is an unordered collection of items that may contain duplicates. Before completing this exercise you should review the methods available to you in the Bag ADT, as well as the implementation techniques we used to construct the array- and linked-based implementations of this ADT.

Your TA will give a lesson reviewing the two main implementations of Bag, and discussing the method that you will write in this exercise.

Exercise

After the TA's lesson, complete the following steps:

1. Download the provided code from the course website. The following Java files are provided in package `cs445.lab3`.
    - `BagInterface.java` is a Java interface representing the ADT Bag
    - `ArrayBag.java` is a dynamic capacity array-based implementation of ADT Bag. It has a nonfunctional stub for the equals method that you must write in this lab.

    - `LinkedBag.java` is a linked implementation of ADT Bag. It also has a non-functional stub for the equals method.
    - `EqualsTest.java` is an example test client of the Bag equals method.
2. Devise an algorithm for comparing two bags to determine if they contain the same contents. Here are some steps you may want to follow:
    - Consider two "equal" bags. What items and frequencies need to be compared to determine that they are equal? What existing bag methods can you reuse to accomplish this?
    - Determine an example of two bags that cannot be equal, yet no item comparisons are needed to make that determination.
    - Write an algorithm for each of `ArrayBag` and `LinkedBag` that returns `true` if two bags contain the same entries. Remember to consider the example from the previous step.
3. Implement your algorithm(s) as method `boolean equals(Object other)` in each class. Since this is overriding a method from the generic `Object` class, you cannot change the method signature, which means you cannot accept a parameter of type `ArrayBag`. This means you must first check if `other instanceof ArrayBag`. If not, return `false`. If so, cast the reference to type `ArrayBag` to complete the rest of the algorithm.
4. Test your `equals` method by running `EqualsTest`. To change which class is tested, change the object type of the instances created for testing (i.e., from `ArrayBag` to `LinkedBag`).

Conclusion

In this lab, you wrote implementation code for the array- and linked-based implementations of the ADT Bag. Throughout the term, you should take the time to practice implementing useful methods for the data structures that we present. You may either choose to implement additional methods, as in this lab, or simply re-implement existing methods without looking at the given code. The techniques demonstrated are best learned through repeated practice