Important

There are a few guidelines you must follow in this homework. If you fail to follow any of the following guidelines you will receive a **0** for the entire assignment.

1. All submitted code must compile under **JDK 7**. This includes unused code, so don't submit extra files that don't compile. (Java is backwards compatabile so if it compiles under JDK 6 it *should* compile under JDK 7.)
2. Don't include any package declarations in your classes.
3. Don't change any existing class headers, constructors, or method signatures.
4. Don't import or use anything that would trivialize the assignment. (e.g. don't import `java.util.LinkedList` for a Linked List assignment. Ask if you are unsure.)
5. You must submit your source code, the `.java` files, not the compiled `.class` files.

After you submit your files redownload them and run them to make sure they are what you intended to submit. You are responsible if you submit the wrong files.

Assignment

We have been studying abstract data types in class, in particular the Bag interface. A bag is a finite collection of items with no inherent ordering. It is similar to a set, except that a bag can contain duplicates. In this homework you will be implementing a bag using an ArrayList.

Imagine an old-school ballot box, where people cast their vote by writing the name of the desired candidate on a slip of paper and putting an envelope containing this slip into a physical box. Unfortunately, the voting procedure for elections has become quite corrupt and some of the people determining the election results have been known to adjust vote counts when given large bribes. The voting system has entirely changed, and now a lot of ballots cast include bribes of various amounts.

You will need to create a program and supporting classes for an election with a rigged ballot box. The ballots should contain the name of the person voted for, as well as the dollar amount bribed. The ballot box should allow votes to be added (when someone votes) and removed or completely emptied (because this is a corrupt system where crazy things are happening). The box can also provide a variety of other information, including the current number of votes, if the box is empty, and if it contains a particular name and the frequency of that name.

The election class provides the main way of interaction with the ballot box. In this class, you will code a command-line menu which allows the user to input choices and receive text output. The menu should include the following actions:

1. **Vote for a candidate.** This should first prompt the user for a candidate name and then prompt the user for the bribe amount. Then the ballot should be added to the ballot box.
2. **Count the number of votes for a candidate.** The user should be prompted to enter a name and the number of votes cast for that name will be printed
3. **Remove a vote.** This should print out a list of all of the current votes in the ballot box and then prompt the user to pick a specific one to remove OR to select the option of removing a vote at random.
4. **Get the number of votes in the ballot box.** This should print out the number of votes in the ballot box.

5. **Empty the ballot box.** This should completely empty the ballot box.
6. **Print the votes in the ballot box.** This should print an organized list of all of the votes in the box. For each vote, it should print the name of the candidate and the amount of the bribe.

The menu does not need to be overly complicated. It just needs to allow the user to interact with your program easily. An example of what this should look like:

```
Welcome to the election!  What would you like to do?
1: Vote for a candidate
2: Count the number of votes for a candidate
3: Remove a vote
4: Get number of votes in the ballot box
5: Empty the ballot box
6: Print all votes
7: Quit
Enter your choice (1-7) here:  1
Enter the name of the candidate you would like to vote for: Dhruv Saksena
Enter bribe amount:  200.50
```

You may assume all input will be valid. Be sure the user can opt to quit the program via the menu immediately (having not done any voting and whatnot).

Provided

The following file(s) have been provided to you.

1. `BagInterface.java`

   This is the interface you will implement. All instructions for what the methods should do are in the javadocs. **Do not alter this file.** Note that the version provided with the homework assignment is slightly different than the version used in the book and lecture.

Deliverables

You must submit all of the following file(s). Please make sure the filename matches the filename(s) below.

1. `ArrayListBag.java`

   This class implements the Bag interface. Refer to the javadocs in BagInterface.java for how to implement each method. You should use the Java class ArrayList for the bag backend, and the implementation should be efficient and use generics.

2. `Ballot.java`

   This class represents a Ballot object. It should store a string name for the candidate and a double value for the bribe amount. The bribe is just stored - it is not used for anything else. You will need to **override the .equals method** to make sure ballots are properly compared. Two ballots are equal if they have the same candidate name; the amount of the bribe does not matter. You should also override the .toString method for printing.

3. `BallotBox.java`

   This class represents a Ballot Box object. It should only be a wrapper around the ArrayListBag which specifies that the bag contains Ballot objects. NOTE: The internal reference should be of type BagInterface (this class should have a variable like `private BagInterface<Ballot> ballots`).

4. `Election.java`

   This class should contain the main method and supporting methods for a command-line menu (described in the Assignment section). The menu should print out menu choices and prompt the user to make a numeric selection. Some menu items will print out additional information and/or menus (depending on the task) - but all menus should be setup so that the user only has to enter an integer number to make a selection.

   Note: It must be possible for a user to start the program, see the menu, and immediately chose to quit. Also your user must be able to quit using the quiz menu option. (See the menu given.)

You may attach each file individually, or submit them in a zip archive. Do not include the .class files. Also do not use any archive file format other than a zip.

All of the code you submit should contain descriptive javadocs and comments to receive full credit.