**WIA1002/WIB1002/WXES1117 Data Structure**

**Lab : ADT**

**Question 1**

Given the interface BagInterface below:

/\*\*

An interface that describes the operations of a bag of objects.

\*/

**public** **interface** BagInterface<T>

{

/\*\* Gets the current number of entries in this bag.

**@return** the integer number of entries currently in the bag \*/

**public** **int** getCurrentSize();

/\*\* Sees whether this bag is full.

**@return** true if the bag is full, or false if not \*/

**public** **boolean** isFull();

/\*\* Sees whether this bag is empty.

**@return** true if the bag is empty, or false if not \*/

**public** **boolean** isEmpty();

/\*\* Adds a new entry to this bag.

**@param** newEntry the object to be added as a new entry

**@return** true if the addition is successful, or false if not \*/

**public** **boolean** add(T newEntry);

/\*\* Removes one unspecified entry from this bag, if possible.

**@return** either the removed entry, if the removal was successful,

or null \*/

**public** T remove();

/\*\* Removes one occurrence of a given entry from this bag.

**@param** anEntry the entry to be removed

**@return** true if the removal was successful, or false if not \*/

**public** **boolean** remove(T anEntry);

/\*\* Removes all entries from this bag. \*/

**public** **void** clear();

/\*\* Counts the number of times a given entry appears in this bag.

**@param** anEntry the entry to be counted

**@return** the number of times anEntry appears in the bag \*/

**public** **int** getFrequencyOf(T anEntry);

/\*\* Tests whether this bag contains a given entry.

**@param** anEntry the entry to locate

**@return** true if this bag contains anEntry, or false otherwise \*/

**public** **boolean** contains(T anEntry);

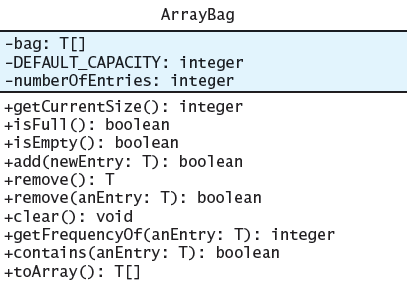
/\*\* Retrieves all entries that are in this bag.

**@return** a newly allocated array of all the entries in the bag \*/

**public** T[] toArray();

} // end BagInterface

Write a ArrayBag class that implement the BagInterface, as shown in the UML diagram below.

****

**Question 2**

The union of two collections consists of their contents combined into a new collection. Add a method union to the interface BagInterface for the ADT bag that returns as a new bag the union of the bag receiving the call to the method and the bag that is the method’s one argument. Include sufficient comments to fully specify the method.

Note that the union of two bags might contain duplicate items. For example, if object x occurs five times in one bag and twice in another, the union of these bags contains x seven times. Specifically, suppose that bag1 and bag2 are Bag objects, where Bag implements BagInterface; bag1 contains the String objects a, b, and c; and bag2 contains the String objects b, b, d, and e. After the statement

BagInterface<String> everything = bag1.union(bag2);

executes, the bag everything contains the strings a, b, b, b, c, d, and e. Note that union does not affect the contents of bag1 and bag2.

Implement the union method in ArrayBag class.

**Question 3**

The intersection of two collections is a new collection of the entries that occur in both collections. That is, it contains the overlapping entries. Add a method intersection to the interface BagInterface for the ADT bag that returns as a new bag the intersection of the bag receiving the call to the method and the bag that is the method’s one argument. Include sufficient comments to fully specify the method.

Note that the intersection of two bags might contain duplicate items. For example, if object x occurs five times in one bag and twice in another, the intersection of these bags contains x twice. Specifically, suppose that bag1 and bag2 are Bag objects, where Bag implements BagInterface; bag1 contains the String objects a, b, and c; and bag2 contains the String objects b, b, d, and e. After the statement

BagInterface<String> commonItems = bag1.intersection(bag2);

executes, the bag commonItems contains only the string b. If b had occurred in bag1 twice, commonItems would have contained two occurrences of b, since bag2 also contains two occurrences of b. Note that intersection does not affect the contents of bag1 and bag2.

Implement the intersection method in ArrayBag class.

**Question 4**

The difference of two collections is a new collection of the entries that would be left in one collection after removing those that also occur in the second. Add a method difference to the interface BagInterface for the ADT bag that returns as a new bag the difference of the bag receiving the call to the method and the bag that is the method’s one argument. Include sufficient comments to fully specify the method.

Note that the difference of two bags might contain duplicate items. For example, if object x occurs five times in one bag and twice in another, the difference of these bags contains x three times. Specifically, suppose that bag1 and bag2 are Bag objects, where Bag implements BagInterface; bag1 contains the String objects a, b, and c; and bag2 contains the String objects b, b, d, and e. After the statement

BagInterface leftOver1 = bag1.difference(bag2);

executes, the bag leftOver1 contains the strings a and c. After the statement

BagInterface leftOver2 = bag2.difference(bag1);

executes, the bag leftOver2 contains the strings b, d, and e. Note that difference does not affect the contents of bag1 and bag2.

Implement the difference method in ArrayBag class.