CSE 2105 – Data Structures
2019 – 2020 Fall Semester Project
The Bag ADT

Write an implementation for abstract data type "**Bag"** as a Java class, which is like a *Set*, but it may contain several instances of the same member. For example, {'to', 'be', 'or', 'not', 'to', 'be'} is a bag of words, which is *equal* to {'be', 'be', 'not', 'or', 'to', 'to'} (since order of members is insignificant), but is *unequal* to {'be', 'not', 'or', 'to'} (since the number of instances is significant). Adding a member increases the number of instances in the bag by one and removing a member decreases the number of instances in the bag by one. *The member is deleted from the bag when it has no instances.*

Think that "how would you represent a bag without actually storing multiple instances of the same member?" Your implementations should be **efficient** as possible.

Don't forget that you **cannot** use any pre-written Java API Class! You should write your own code for data structures. You are allowed to use lab source codes that we examined so far, however it is *preferable* to write your own codes for your data structure.

Your implementation must be **generic**, i.e. your bag can accept any type of instances. (Use Java Generics!)

Your ADT should provide at least these operations (you can add other "public" or "private" methods you want):

- **add(AnyType item):** Inserts item in the bag.
- **clear():** Removes all of the elements from this bag.
- **contains(AnyType item):** Returns true if this bag contains the specified element and false otherwise.
- **distictSize():** Returns the distinct number of elements in this bag. (For example, for the bag {'to', 'be', 'or', 'not', 'to', 'be'} distinct size is 4)
- **equals(Object obj):** Indicates whether some bag object is "equal to" this one. (Returns "true" or "false")
- **elementSize(AnyType item):** Returns the number of this item in this bag. (For example, for the bag {'to', 'be', 'or', 'not', 'to', 'be'} element size('be') is 2)
- **isEmpty() :** Returns true if this collection contains no elements.
- **remove(AnyType item):** Removes a single instance of item from this bag and returns true if it is present; otherwise returns false.
- **size():** Returns the total number of elements in this bag. (For example, for the bag {'to', 'be', 'or', 'not', 'to', 'be'} size is 6)
- **toString() :** Returns a string that displays the elements in the bag. (The style to be used is up to your preference.)

Please also write a Test program (a separate Java class) proves that your container(bag) works properly.

P.S.: You can prepare your project yourself (single person) or as **at most two** people groups. (It means you cannot work as a three/four-people group!) You have to submit a "report" **(report is very important !, especially you have to explain your data structures usages, how and why)**, and "Java code" (your classes) of your program. (You can use any IDE you want.) Please upload your compressed(zip/rar) file (that includes your report(as "pdf" file) and Java source code files (with ".java" extension) – not all the project files !) to the **moodle** page of the course to the appropriate area (projects that are sent via e-mail or other different ways of sending, will not be accepted!) before **22 December 2019, Sunday**, **23:00.** (**Projects that not uploaded to moodle page of the course until this time, will definitely not be accepted!**). One of the group member's project upload is enough, but please write all group members' numbers and names to your report.

Very important note: Every student (each member of the group individually) is going to demonstrate his/her project to the research assistants in a specific time. (the program will be announced later). So, uploading your project is not enough, you are expected to explain your algorithms and codes.