COT6936 – Project 1

Objective

Given a problem and different algorithmic methods to solve it, students will be able to perform a comparative study of the performance of the methods by conducting an algorithm experiment.

Assignment Problem

Consider a robotics application involving a sensor that generates a continuous data stream of integer numbers. Suppose you are tasked to implement a data structure in Java that will be used to store values read from the sensor. The data structure should allow for insertions, removals, and searches of the values generated by the sensor. You are not given specifics on the data distribution, but you are told that data will be in the order of a few tens to millions of numbers. Number duplicates are possible. You think your problem can be solved with implementing a simple bag data structure (i.e., a multiset), and after careful analysis you think you are left with two choices, 1) implement the bag with arrays, or implement the bag by using the ArrayList class in the package java.util. While the Java array-based list is a data structure ready for use, you wonder if it would be more efficient to design your own solution by implementing the bag from scratch with arrays. Efficiency is deemed extremely important because values will be generated continuously at a fast pace.

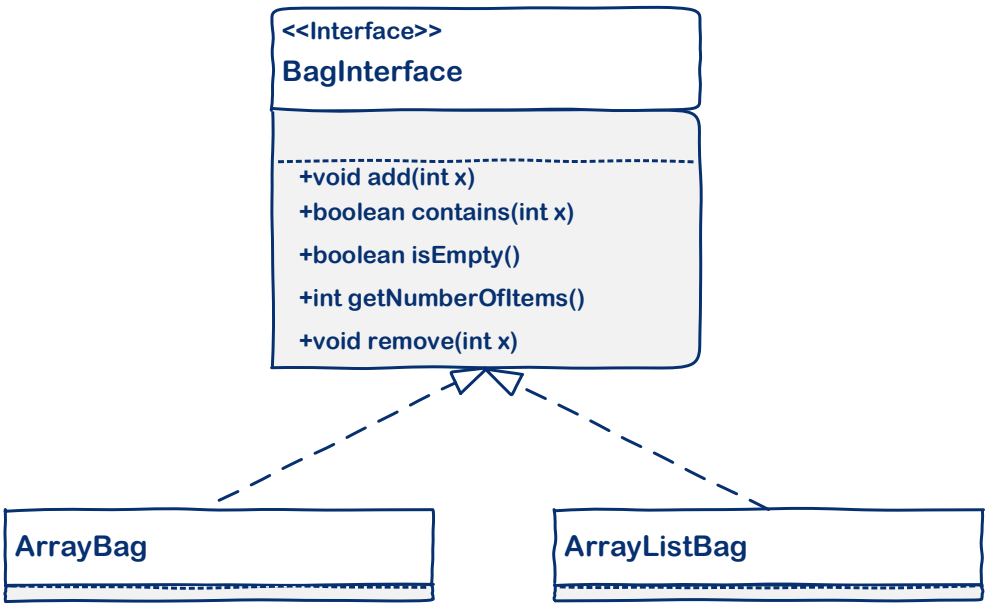
The problem is therefore to design and implement a computational experiment to contrast the performance of both, the array-based bag and the java.util.ArrayList-based bag. The following requirements must be used in your study.

Please consult the class slides for a discussion on experimental algorithmics.

Requirements

- Consider several input classes: data is random (use these two distributions: uniform distribution, normal distribution), data is generated sorted or almost sorted; data is not sorted.
- You want to study the problem with both small sizes and large sizes of the data set.
- Running times of each approach will be measured against the input classes above, for each of the operations mentioned: addition of a new data value, removal of a given data value, and search of a given data value.
- Note that values to be added, removed, or searched are generated according to the distributions specified above.
- Running times will be saved in a .csv file to be open in Excel later.
- Plot, using Excel, the times obtained and save the graphs as Excel files.
- Comment all classes and methods with javadoc-style comments, including class/method purpose, method parameters, and method return value.

UML Diagram



(*) the remove method removes one occurrence of x, if x is present.

Guidelines

The assignment is to be completed in teams of up-to three students. The given problem is based on the content studied in class on experimental algorithms. You are allowed to use the code discussed or given in the lectures. Where appropriate, credit its source.

Deliverables

- A compressed folder, *names_Project 1* (e.g. *Smith_John_Project 1*), containing:
  1. all of the source code of the exercise
  2. Conclusions (Word or PDF file): a document explaining what you observed in the experiment and your conclusions. This document will include the text of your explanations and the picture(s), chart(s), or diagram(s) obtained in Excel
  3. The .csv files and the corresponding Excel files
- Include **only** the .java files (source code); do not include other files or folders generated by the IDE.
- Make sure you write name(s) and PID(s) in the class comment section of each Java file, in the Conclusions document, as well as in the comment window of the project dropbox.
- Only one member of a team will submit the project.

Grading Rubric

The assignment is worth 20 points (out of 100 total course points). Grade components:

<table>
<tr><th>Component</th><th>Points</th><th>Description</th></tr>
<tr><td>Submission</td><td>1</td><td>The student has submitted the project solution using the requirements for deliverables specified in the Deliverables section.</td></tr>
<tr><td>Organization</td><td>1</td><td>Code is expected to be neat, organized, and readable.</td></tr>
<tr><td>Content</td><td>18</td><td><table><tr><th>Deliverable</th><th>Points</th></tr><tr><td>source code</td><td>12 pts</td></tr><tr><td>conclusions</td><td>4 pts</td></tr><tr><td>.csv files</td><td>1 pts</td></tr><tr><td>Excel files</td><td>1 pts</td></tr></table></td></tr>
</table>