Computer Science 501
Data Structures & Algorithms
The College of Saint Rose
Fall 2013

Programming Project 1: Timing Java

Due: 6:00 PM, Wednesday, September 25, 2013

Our first programming project is to write a Java program or programs that will perform emprical analysis of a number of operations that you can compare with the theoretical expectations, and to write up a formal report of your results.

You may work individually or with a partner on this project.

Timing Java Code

Read the lab description on pages 115-117 of Bailey. We will not be doing this lab specifically, but it explains the ideas and techniques you will need to be able to write a program that can generate good timing data.

Experiments

Design and implement experiments to analyze the efficiency of each of the following operations:

- The construction of an n-element array of `int`.
- Inserting n numbers into an n-element array of `int`.
- Accessing the element at index 0 of an n-element array of `int`.
- Accessing the element at index $n - 1$ of an n-element array of `int`.
- Adding n integer values to a default `Vector` from the structure package.
- Adding n integer values to a `Vector` whose initial capacity is 1 element, and which increases the size of its internal array by 1 element each time an element is added that does not fit in the internal array. Hint: use the proper constructor of your `Vector` to achieve this behavior.
- Adding n integer values to a `Vector` whose initial capacity is 10 elements, and which increases the size of its internal array by 10 elements each time an element is added that does not fit in the internal array.
- Adding n integer values to a `Vector` whose initial capacity is 1 element, and which increases the size of its internal array by doubling each time an element is added that does not fit in the internal array.

- Adding n integer values to a `Vector` whose initial capacity is n elements. The resizing discipline is not relevant here, as the `Vector` will never need to be resized.
- Inserting n integer values into position 0 of a `Vector` whose initial capacity is n elements. The resizing discipline is not relevant here, as the `Vector` will never need to be resized.
- Accessing the element at index 0 in a `Vector` that contains n integer elements.
- Accessing the element at index $n - 1$ in a `Vector` that contains n integer elements.

In each case, you will need to determine an appropriate range for values of n to use for your experiments to be able to see the trends in the behavior of the operation. You will need to perform multiple runs of each to generate meaningful timings.

Analysis and Discussion

For each experiment, you should do the following in your writeup:

- State and jusify the theoretical expectation for the efficiency (*e.g.*, $O(n^2)$).
- Present your timing results in tabular and graph formats.
- Compare your timing results with the theoretical expectation, explaining any discrepencies to the best of your ability.

Submission

Before 6:00 PM, Wednesday, September 25, 2013, submit your Java program(s) and writeup for grading. Do this through Submission Box at `http://sb.teresco.org` under assignment "TimingJava". Since SubmissionBox can only accept one file per assignment, please package up your files in a "zip" or similar archival format first.

Grading

This assignment is graded out of 50 points, broken down as follows:

Grading Breakdown	
General timing framework	5 points
Java code for specific operations	15 points
Java code style, documentation, and formatting	5 points
Theoretical expectations	10 points
Presentation and analysis of timing results	15 points