Abstract

In this first experiment, the program will create two arrays and add their contents together. The program will time how long it takes to perform this operation. We will be looking at single dimension arrays and two-dimensional arrays.

Experiment

First Experiment:

- Single Dimension Array
- Array Size: 29,997,529
- Operation: Addition

Second Experiment:

- Array Dimensions: 2D
- Array Size: 5,477 x 5,477
- Operation: Addition

Because we cannot make assumptions off just one run through of the experiment, two methods were made called single() and mult() for each experiment. In the main method, we will run these experiments ten times and take the average of their clocked times. Then, we will print the average as calculated.

In the first experiment, we use the single() method that uses single dimension variables. First, we fill two arrays with the values 5 and 2 respectively. After making the result array, we begin the timer. The for loop adds both arrays into the result array and once that operation is finished, the timer stops. The average recorded for this experiment clocked in at 15,492,140 nanoseconds or 15.49214 milliseconds.

The second experiment is largely the same, but since we are using two-dimensional arrays, we have to use a different method to fill it. Importing the java.util.Array library gives us the fill function to make this easier. We use a for each loop to fill each array. Remember that in java, multi-dimensional arrays are always represented as an array of arrays. The average recorded for this experiment clocked in at 15,617,720 nanoseconds or 15.61772 milliseconds.

Conclusion

The two averages that we computed were very similar, meaning that Java handles multi-dimensional arrays as efficiently as it handles single-dimension arrays of equal size.