

Forum:[Hack Forums](#)

Thread: [Optimizing this simple method?](#)

Dyst0pia - September 19, 2016, 4:02 pm

I need to make 1 array out of two different arrays and then sort the new array, I've done it this way and it works perfectly but I keep thinking that its not the most effective way to do it.
Does anyone have any input on how I can optimize this code?

PHP Code

```
<?php
public static void fade() {

    int array1[] = {131, 42, 3, 54, 5, 6, 6, 8, 9, 32};
    int array2[] = {5, 6, 741, 55, 25, 22, 666, 6, 1, 5};
    int arrayMerged[] = new int[(array1.length + array2.length)];

    int count = 0;
    int swap;

    // Joins the two arrays together
    for (int i = 0; i < arrayMerged.length; i++) {
        if (i <= array1.length-1) {
            arrayMerged[i] = array1[i];
        } else if (i >= array1.length-1) {
            arrayMerged[i] = array2[count];
            count++;
        }
    }

    // Sorts the new array accordingly
    for (int k = 0; k < arrayMerged.length; k++) {
        for (int j = k + 1; j < arrayMerged.length ; j++) {
            if (arrayMerged[k] > arrayMerged[j]) {
                swap = arrayMerged[k];
                arrayMerged[k] = arrayMerged[j];
                arrayMerged[j] = swap;
            }
        }
        System.out.println(arrayMerged[k]);
    }
}
```

Stealth™ - September 19, 2016, 5:06 pm

With Java 8, you can do it in a very simple way!

Dyst0pia - September 20, 2016, 2:23 am

(Sep 19, 2016 05:06 PM)Stealth™ Wrote: [\[CITE\]](#) With Java 8, you can do it in a very simple way!

Sorry I should have explained better, I'm not allowed to use built in java methods. So that won't be possible.

Sogomn - September 20, 2016, 5:30 am

You could use another sorting algorithms than bubblesort. It has a runtime of $O(n^2)$.

Goodies - September 20, 2016, 8:35 am

(Sep 19, 2016 05:06 PM)Stealth™ Wrote: [\[CITE\]](#) With Java 8, you can do it in a very simple way!

Not to mention java 8 streams, while sexy, are not efficient. At least not yet.

Stealth™ - September 20, 2016, 9:35 am

(Sep 20, 2016 08:35 AM)Goodies Wrote: [\[CITE\]](#) Not to mention java 8 streams, while sexy, are not efficient. At least not yet.

Well there are parallel streams, but I still have no clue how to use them properly yet :P

And efficiency shouldn't really be an issue for what OP is trying to achieve.

Goodies - September 20, 2016, 10:24 am

The route I went for efficiency and maintaining a low time complexity:

~ quicksort for sorting each array individually with $O(n \log n)$ average case. The sorting algorithm was derived straight from the wiki. <https://en.wikipedia.org/wiki/Quicksort>

~ merging the two sorted arrays efficiently with $O(n)$.

I tested it with this. Takes an average of about 43ms on my PC.

PHP Code

```
<?php
import sorting.Dystopia;

import java.security.SecureRandom;
import java.util.Arrays;

/**
 * Created by Austin Archer on 9/20/2016.
 */
public class Main {
    private static final int SIZE = 100000, VARY = 10000;

    public static void main(String args[]){
        SecureRandom rand = new SecureRandom();

        int size1 = rand.nextInt(VARY) + SIZE, size2 = rand.nextInt(VARY) + SIZE;
        int[] arr1 = new int[size1], arr2 = new int[size2];
```

```
for(int i = 0; i < size1; ++i){
    arr1[i] = rand.nextInt();
}

for(int i = 0; i < size2; ++i){
    arr2[i] = rand.nextInt();
}

long start = System.nanoTime();
Dystopia.quicksort(arr1);
Dystopia.quicksort(arr2);
final int[] merged = Dystopia.mergeSorted(arr1, arr2);
long end = System.nanoTime();
System.out.println(String.format("Took %.3fms to sort and merge the arrays.", ((end - start) /
1000000.0)));
}
}
```