# CE Merge CODE

MISSING 10 Possible Points

**| 10/3/2022**

Attempt 1

IN PROGRESS
Next Up: Submit Assignment

Add Comment

**Unlimited Attempts Allowed**

**Details**

Sorting

## CE: Merge Sort CODE

## Learning Objectives

- Sort an array using the merge sort algorithm
- Modify an existing implementation of merge sort and demonstrate the impact on the performance.

## Overview

In this CE, you demonstrate that seemingly small changes in the implementation can lead to vast differences in the performance.
To do so, you will create a modified version of class Merge and you will measure the performance of the two implementations.

Previous (https://slcc.instructure.com/courses/817632/modules/items/18752979)

Submit Assignment

Next (https://slcc.instructure.com/courses/817632/modules/items/18752983)

# Instruction

- Create a package called **ceMerge**
  Add class **MergeSlow** and class **MergeComparison** that includes the main method.

- **Class MergeSlow:**
  - Copy the implementation from class **Merge** (https://algs4.cs.princeton.edu/22mergesort/Merge.java.html) as a starter code, and make the necessary changes so that it compiles.
  - Find the public method sort. It calls the recursive method sort. However, before it does so, it creates an auxiliary array and passes it as the second argument. This is a very important step. It allows the recursive method to reuse the same auxiliary array over and over again.
  - In this exercise, we explore what happens when we don't pass the auxiliary array. Change the recursive sort method header and remove the second parameter (the auxiliary array). To make the method compile, remove the argument aux when calling the methods sort and also when calling merge.
    This impacts the method merge. Remove the second parameter (the auxiliary array) from the method merge. Since the array aux is needed for merging, create it as the first statement in the body of the method merge.
    There is still a syntax error we need to fix. Now that we changed the signature of the recursive method sort, the method call in the public sort method no longer compiles. Fix the problem by removing the second argument (aux) and delete the statement above that creates the auxiliary array. It is no longer needed in the public sort method. At this point, class MergeSlow should compile.

- **Class MergeComparison**
  In this class, we will time how long it takes to sort multiple number arrays of various sizes using Merge sort. Then we'll repeat the experiment using the modified class MergeSlow.

  1. Write a method called **getNumbers**. It has one parameter, which is the array size, and it returns an array that is initialized with random 6-digit numbers.
  2. Create an Integer array and initialize it with 1024 random 6-digit numbers.
  3. Use the method **nanoTime** from class System to measure how long it takes to sort the array with the original merges sort from class Merge.

Previous (https://slcc.instructure.com/courses/817632/modules/items/18752979)

Submit Assignment

Next (https://slcc.instructure.com/courses/817632/modules/items/18752983)

header that indicates that we used Professor Sedgewick's class Merge to sort the array.

7. Repeat 2 - 6.
   However, this time use the sort method from class MergeSlow. Use white space to group related output as shown in the Sample Output.
   Notice how the performance starts to deteriorate as the array size increases. If you can't see a significant performance difference in your output, increase the initial number of array elements.

**Take-Away:**
Even an efficient algorithm can have a dismal performance if we don't implement it efficiently.

## Sample Output

```
         n     Merge
---------- ------------
     1,024     0.0067s
     2,048     0.0024s
     4,096     0.0034s
     8,192     0.0069s
    16,384     0.0311s
    32,768     0.0460s
    65,536     0.2324s
   131,072     0.3225s
   262,144     0.2653s

         n     MergeSlow
---------- ------------
     1,024     0.0398s
```

Previous (https://slcc.instructure.com/courses/817632/modules/items/18752979)

Submit Assignment

Next (https://slcc.instructure.com/courses/817632/modules/items/18752983)

```
  65,536     2.1249s
 131,072    49.8824s
 262,144   212.2430s
```

## Submission

Create a screen recording following the **guidelines for lab recordings (https://slcc.instructure.com/courses/817632/pages/guidelines-for-ce-recordings)** .
The video should be **25-50 seconds** long.
Post the video.

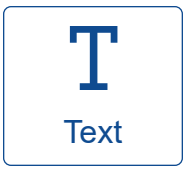
- **(https://slcc.instructure.com/api/v1/courses/817632/module_item_redirect/18752970)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752971)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752972)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752974)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752975)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752976)**
- **(https://slcc.instructure.com/api/v1/courses/817632/module_item_redirect/18752978)**
- **(https://slcc.instructure.com/api/v1/courses/817632/module_item_redirect/18752979)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752981)**
  - **(https://slcc.instructure.com/courses/817632/modules/items/18752983)**

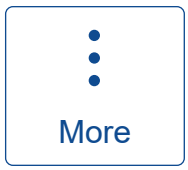
C

Previous (https://slcc.instructure.com/courses/817632/modules/items/18752979)

Submit Assignment

Next (https://slcc.instructure.com/courses/817632/modules/items/18752983)






Previous (https://slcc.instructure.com/courses/817632/modules/items/18752979)

Submit Assignment

Next (https://slcc.instructure.com/courses/817632/modules/items/18752983)