CE Quicksort CODE

MISSING 10 Possible Points

| 10/17/2022

Attempt 1

IN PROGRESS
Next Up: Submit Assignment

Add Comment

Unlimited Attempts Allowed

Details

Sorting

CE: Quicksort CODE

Learning Objectives

- Sort an array using quicksort.
- Measure the time it takes to sort arrays of various sizes.
- Demonstrate the performance differences between O(N log N) and O(N^2).

Overview

In this CE, you will write a method that generates an array of random integers. You will use such integer arrays to compare the performance of the following two sorting algorithms: selection sort and quicksort.

Previous (https://slcc.instructure.com/courses/817632/modules/items/18753004)

Submit Assignment

Next (https://slcc.instructure.com/courses/817632/modules/items/18753007)

- Create a package called **ceQuick**
  Add a class called **SortComparison** that includes the main method.

- Create a private static method called **getRandomNumberArray**. It returns an array of Integer and has 2 parameters:
  arraySize of type int and numberOfDigits of type int.
  This method should create an array of the specified size that is filled with random numbers where each random number has the number of digits specified in the second parameter (no leading zeros).
  Write some test code to verify that the method getRandomNumberArray works as expected. Once you are confident that this is the case, remove the test code.
  Because of the limited range of integer values, the number of digits needs to be a value from the range [1,10]. If the second argument is not within this range, throw an IllegalArgumentException that includes the message: "The number of digits needs to be a value between 1 and 10."

- In your **main method** (or additional private methods) do the following:
  Repeatedly execute selection sort and quicksort on an array of numbers
  - Create an array of n 7-digit numbers, where n is 1000. If the performance of your computer significantly differs from the sample output, you might need to start with a smaller or larger n.
  - To ensure a meaningful comparison, both algorithms need to sort the exact same sequence of random numbers (e.g. if I run selection sort on [1, 5, 2, 7, 6, 4, 9, 8] then I need to run quicksort on [1, 5, 2, 7, 6, 4, 9, 8].
    Create a copy of the array to ensure that both sorting methods are called on the identical array.
  - Use the method **nanoTime** from class System to measure how long it takes to sort with Selection sort and how long it takes to sort with quicksort.
  - Print the results side-by-side in tabular form as shown in the sample output
  - After the two sorting algorithms have been executed and the elapsed time has been printed, the size of n is doubled so that in the next iteration a new array of twice the size can be created.
  - Print the output in tabular form as shown in the output

< Previous
(https://slcc.instructure.com/courses/817632/modules/items/18753004)

Submit Assignment

Next >
(https://slcc.instructure.com/courses/817632/modules/items/18753007)

The table should have eight to ten rows, and the size of n needs to be chosen to show the following patterns that indicate the big O of the algorithm.

- If the duration after doubling the size of n is approximately 4 times bigger than before, that indicates $O(N^2)$
- If the duration after doubling the size of n is about twice as big as before, that indicates big O(N)
- If the duration after doubling the size of n is a bit more than twice as big as before, that indicates big O(N log N)

Sample Output

The actual numbers will differ. However, the importance is the demonstration of the following two patterns: That the measured time of Selection sort approximately quadruples as n is doubled and the measured time of Quicksort slightly more than doubles. The measurements are less precise when n is small, so run the sorting algorithms often enough to display decisive results. Depending on your computer, you might need to adjust n by starting with a smaller or larger number.

```
         n | Selection | Quick     |
----------|-----------|-----------|
     1,000 |  0.0191s  |  0.0088s  |
     2,000 |  0.0471s  |  0.0011s  |
     4,000 |  0.0660s  |  0.0045s  |
     8,000 |  0.0860s  |  0.0116s  |
    16,000 |  0.3063s  |  0.0538s  |
    32,000 |  1.4400s  |  0.0202s  |
    64,000 |  5.5242s  |  0.0346s  |
   128,000 | 42.9131s  |  0.0736s  |
   256,000 | 154.2390s |  0.1510s  |
   512,000 | 690.5412s |  0.3169s  |
```

< Previous (https://slcc.instructure.com/courses/817632/modules/items/18753004)

Submit Assignment

Next > (https://slcc.instructure.com/courses/817632/modules/items/18753007)

Create a screen recording following the **guidelines for lab recordings (https://slcc.instructure.com/courses/817632/pages/guidelines-for-ce-recordings)** .
Show the code and the first few lines of the output as the program starts to execute. Pause the video while the remaining time measurements are taken. Then turn it back on and show the completed table.

The video should be **35-70 seconds** long.
Post the video.

(https://slcc.instructure.com/courses/817632/modules/items/18752995)
(https://slcc.instructure.com/courses/817632/modules/items/18752996)
(https://slcc.instructure.com/courses/817632/modules/items/18752997)
(https://slcc.instructure.com/courses/817632/modules/items/18752998)
(https://slcc.instructure.com/courses/817632/modules/items/18752999)
(https://slcc.instructure.com/courses/817632/modules/items/18753000)
(https://slcc.instructure.com/api/v1/courses/817632/module_item_redirect/18753002)
(https://slcc.instructure.com/api/v1/courses/817632/module_item_redirect/18753004)
(https://slcc.instructure.com/courses/817632/modules/items/18753006)
(https://slcc.instructure.com/courses/817632/modules/items/18753007)
(https://slcc.instructure.com/courses/817632/modules/items/18753009)

Choose a submission type

< Previous **(https://slcc.instructure.com/courses/817632/modules/items/18753004)**

Submit Assignment

Next > **(https://slcc.instructure.com/courses/817632/modules/items/18753007)**

Previous
(https://slcc.instructure.com/courses/817632/modules/items/18753004)

Submit Assignment

Next
(https://slcc.instructure.com/courses/817632/modules/items/18753007)