Algorithms Dispute

A person sitting at a desk

Description automatically generated

There is a big problem between three different algorithms. Each of them claimed to be the best algorithm in sorting. Due to the tense public atmosphere, they went to court to resolve the dispute between them.

**Judge:** Could you introduce yourself to me?

**Algorithm A:** I am a sorting algorithm that counts the frequency of each number in the array and uses that information to place the numbers in their correct sorted positions. I do believe that I am the best algorithm in sorting.

**Judge:** Could you please show me an example to understand how you sort the numbers?

**Algorithm A:** I sort a collection of numbers. First of all, I determine the range of the given numbers in an array by finding the maximum and the minimum values.

A black and white rectangular with numbers

Description automatically generated

The range is 9, Min=4 and Max=13.

Then I create an Index array with size equal to the range +1. This array holds the frequency of each number in the original array.

A number in a row

Description automatically generated

After that, I create a new array to hold the sum of frequencies for a given index.

A number in a row

Description automatically generated

Then, I create a new array, 'Sorted Original Array,' with the same size as the original array. To populate it, I traverse the original array in reverse order and place each element into its correct sorted position in the 'Sorted Original Array' based on the values in the 'Sum Frequency' array. After placing an element, I decrement its frequency in the 'Sum Frequency' array. For instance, starting with element 13 (the last element in the original array), I position it at index 6 in the 'Sorted Original Array', I determined the correct position based on the value at index 13 in the sum frequency array.

A number and a number

Description automatically generated with low confidence

**Judge:** Thank you! May I have the next algorithm please. Can you please introduce yourself to me?

**Algorithm B:** How can’t you recognize me? I'm a famous algorithm!

**Judge:** I'm really sorry, I didn't recognize you. Can you please show me an example to understand how you sort numbers?

**Algorithm B:** Uh... I won't give you an example, but I will give you a hint.

If you apply me on the following array:



And if you stop the run before I finish sorting, the array will look like this:



**Judge:** Thank you! May I have the next algorithm, please? Could you please introduce yourself to me?

**Algorithm C:** I am a sorting algorithm based on the principle of resolving a problem into two simpler sub-problems. Each of these sub-problems may be resolved to produce yet simpler problems. The process is repeated until all the resulting problems are found to be trivial. These trivial problems may then be solved by known methods, thus obtaining a solution to the original more complex problem.

**Algorithm B:** Excuse me! Judge. But Algorithm C is describing me too.

**Algorithm C:** I have not finished yet; If you apply me on the following array:



And if you stop the run before I finish sorting, the array will look like this:



**Judge:** Oh! You are so down to earth. Thank you!

**\*Silence roomed in the court\***

**Judge:** The court is adjourned.

Your Task

After reading the above dialogue, as a Judge, you should wisely decide who is the best algorithm for sorting. You have to conduct theoretical and experimental analysis for the three algorithms to make a good decision. To do so:

**1.** Implement the three algorithms in **main.cpp**.

**2.** Perform experimental analysis to measure the running time each of the three algorithms take. Click on the **Run** button to run the experiments and report your results in **experiments.txt** . The input sizes and types you need to test are provided in experiments.txt .