# Michael Chillemi

# 07/02/2023

# CS 590 - Algorithms

# M7.B2: Module 7 Sorting and Selection Creativity Exercises

Problem 8.5.14

Suppose we are given a sequence S of n elements, on which a total order relation is defined. Describe an efficient method for determining whether there are two equal elements in S. What is the running time of your method?

Answer:

The time complexity of an effective solution to this issue will be O(n log n). You will first order the components of sequence S. It will take about O(n log n) time to complete that task. Next, go through each step looking for two items that follow each other and are equal. This extra step will take roughly O(n) time. As a result, this method will ultimately take O(n log n).

Problem 9.5.12

Suppose we are given a sequence, S, of n integers in the range from 1 to . Give an

-time method for determining whether there are two equal numbers in S.

Answer:

There are various different ways that one could go about solving this problem. One strategy would be to sort the array before looking for adjacent duplicate values. It would take as long as n logarithm, on average. Another strategy that might be used is to create a hash map of the values in the array. It would take O(n) time to construct this, and O(1) time to seek up the values. Utilizing the brute force method, which compares each value in the array to every other possible value, would be a third tactic. Time required for this would be exponential. Sorting the array and searching for duplicate values in neighboring cells are two methods to determine whether two sets of comparable numbers are displayed in a row. The amount of time needed to complete this process would increase as log(n). Making a hash map of the array's values is another way to check for duplicate integers in a sequence. Building it takes O(n) time, whereas checking up values takes O(n log n) time. The third method checks to see if there are two groups of related integers in the sequence by applying "brute force," which entails comparing each value in the array to every other value. This would require an exponentially long period of time in . There are numerous approaches to address this issue. The array should be sorted to find duplicates. O(n log n) seconds. Values from arrays can also be hashed. Building this takes O(n) time, while looking up values takes O(1) time. A third approach is to compare each value in an array to every other value. speed.