COMP3121 Homework Q1

Arth Sanskar Patel
z5228942

June 15, 2020

1 Answer a

We can go through all the pairs and compute the sum of their squares and add them to a new array. This will take $O(n^2)$ time since there will be nested for loops to check all the pairs. Now we have an auxiliary array (Lets call it B) with $n(n-1)/2$ elements in it. MergeSort B will take $O(n^2 logn)$ time. Note there that general mergeSort with n elements take $O(nlogn)$ time but since B will have n^2 elements, it will take $O(n^2 logn)$ time. now we go through elements of B and find if we have double or duplicate values. We can do this using a normal binary search with each element which will take $O(n^2 logn)$ time as there are n^2 elements. In the end we return true if we find a duplicate or return false if not.

2 Answer b

For this part we can use a hashmap which has a search time of O(1). We can do the same as part a and make an auxiliary hashmap B in $O(n^2)$ time. Then we can search the hashmap for duplicates in $O(n)$ time and hence the expected time will be $O(n^2)$. There is another way of doing this. We create the hashmap in the same way using $O(n^2)$ time. Instead of having a search outside our nested loops, we can have a search inside the nested for loop which searches for duplicate of the element that we are adding in the hashmap. Since the search will take $O(1)$ time the total time would still be $O(n^2)$ and then return true if we find a duplicate or false otherwise.