README

Problem

1. This function complexity is O(n log n)
   because i use firstly break the list into small parts and than i use a loop
   In this function firstly divide list or array into small parts and than check the condition if there is duplicate than return True otherwise its check the other element and if it was not find any duplicate in list than return False

   merge sort time complexity

   $T(n) = 2\,T(n/2) + n$

   $= 2\,[2\,T(n/4) + n/2] + n$

   $= 4\,T(n/4) + 2n$

   $= 4\,[2\,T(n/8) + n/4] + 2n$

   $= 8\,T(n/8) + 3n$

   $= 2^k\,T(n/2^k) + k\,n$

   $= 2^k\,T(n/2^k) + k\,n$

   $= 2^{\log_2 n}\,T(1) + (\log_2 n)\,n$

   $= n + n \log_2 n$

   $= O(n \log n)$

   and than i used a loop and than the complexity is n + n log n + n but the big is n log n than we consider the n log n

2. Complexity of this function is O(n log n)

because i use firstly break the list into small parts and than i use a loop
In this function firstly i sorted list and than put i in first position of the list and than i checked difference the next element is equal or more than key or not and i increase continousely and put condition if there is duplicate than return True otherwise return False
merge sort time complexity

$T(n) = 2\,T(n/2) + n$

$= 2\,[2\,T(n/4) + n/2] + n$

$= 4\,T(n/4) + 2n$

$= 4\,[2\,T(n/8) + n/4] + 2n$

$= 8\,T(n/8) + 3n$

$= 2^k\,T(n/2^k) + k\,n$

$= 2^k\,T(n/2^k) + k\,n$

$= 2^{\log_2 n}\,T(1) + (\log_2 n)\,n$

$= n + n \log_2 n$

$= O(n \log n)$

and than i used a loop and than the complexity is n + n log n + n but the big is n log n than we consider the n log n

Explaination

divide, solve, and combine

Divide

Break the given problem into subproblems of same type

In divide part we divide the list into small parts like in my solution i have a list than i divide my list firstly into two parts and than again to divided part divide into two parts and we divide that till the list is not small into two elements now we have only two element in list

Solve

Recursively solve these subproblems

And now i have small list and i have only two element in list than i compare that element with each other and put it into small to big order now we do same thing with all small lists

combine

Appropriately combine the answers

And now we have many small list with already sorted and now we have to combine the list firstly we take one list and check the first element of both list and which is small that element we append to the list and like this we will compare all the list

optional

exercise 3.

1. No we cannot solve the 2 dimensional space with a similar approach because in this firstly i used the merge sort and in that function its break the into small list

2. Yes firstly we have to make the size of the list is two

because it is two dimensional and second think is calculating the distance because in 1 dimensional space we can divide the element and than we compare that in 2 dimensional we use the technique we have to keep two element and than compare

3.
No