ID: 22141026

Task 1

(a)

In this code, I used nested loop to compare all possible pairs of numbers, resulting in a time complexity of $O(n^2)$. It checks for pairs of numbers that sum up to the desired output and break the loop if a valid pair is found.

(b)

I used two pointer approach to efficiently search for a pair of number in the list that sum up the target value (val). The pointer traverse list only once, which has the O(n) complexity.

Task 2

(a)

It reads two list of integers and using (+) I merged them into one. As I used default Sort function which gives O(nlogn).

(b)

It uses two pointers i and j to iterate both list exactly once. The merging process has a linear time complexity of O(N).

Task 3

The main part of the code sorts a list of tasks by their end times and then iterates through the sorted list to find the maximum number of non-overlap. Here for the sorting is $O(n \log(n))$

Task 4

Here, Just like task 3, I solved the task by sorting the list of tasks by their start time and efficiently assigning them to m people while tracking completed task. It has a time complexity of $O(N \log(N))$