is calculated like task 3 and the difference is set to a negative number as it not going to be used. If the positive array was not empty we find the ~~task to~~ t' pair with the least difference and assinged to ~~the~~ that ~~task~~ person / inner list. if not positive difference exists if there is empty array then it will be sent there. and we count of how many task ~~ass~~ inged..

the most time consumming step in the algorithm is sorting the task by their end time which is done in O(nlogn) times. using standard sorting algorithm.

Task 3

Here first we insert all the task into a list and then sort them based on ascending order of ending time. Then we iterate through list If the list is empty then task will be added in to a seperate list and the next task will dragged in the way, difference is current task start time minus previous task end time. And the difference is positive then the task is added in a list and this check is done to ensure that the task don't clash with one another.

Task 4

Here we get the number of people from input file and we creat a empty list of len of input inside a larger list. The task are inside a list task and sorted based on their ending time then iterate ones the task. and if the inner list are not empty .iteratively. The ~~empty task~~ difference is