HW5 ALGORİTHMS AND TIME COMPLEXITIES

Q1) Firstly, jobs are sorted in descending as Wi/Ti. Where wi is current job weight and ti is the job time. Wi/Ti provides optimal solution to minimize the weighted sum of the completion times. After jobs are sorted, Ci = Wi * Ti is calculated to first job. For next jobs, same formula is used but Ti is increment by previous job time. For second job Ci = Wi * (Ti-1 + Ti) then it continues. Total sum weight is calculated by this way. Time complexity of this solution is O(nlogn + n) = O(nlogn) nlogn comes from python Tim sort for sorting jobs. n comes from calculating minimum wighted sum. Therefore O(nlogn).

Q2)

a)

	Month 1	Month 2	Month 3	Month 4
NY	2	5	3	12
SF	4	1	8	6

In this example given algorithm calculates total cost = 2 + 1 + 3 + 6 = 12 since it just look to minimum cost in month and it does not include move costs. Algorithm's plan of minimum cost would be the sequence of location : [NY,SF,NY,SF].
Since the location is changed on each month, there would be 3 * M move costs addtion to total cost. In correct approach of this problem, correct answer should be Cost = 4 + 1 + 8 + 6 = 19

b) First, two distinct list defined to indicate the minimum costs for month in NY and SF. Then set first month cost as a min cost for NY and SF to these lists. For each month costs 1 to i, below formulas are used to get minimum costs:

minCostNY[i]= NY[i]+ min(minCostNY[i-1],M+minCostSF[i-1])
minCostSF[i] = SF[i] + min(minCostSF[i-1],M+minCostNY[i-1])

Check the sum of move cost with the cost of city is greater than cost of current city. If the sum of move cost with the cost of other city is greater than the cost of current city, stay in the current city.Then calculate all months minimum cost by this way.
Time complexity of this solution is O(n) where n is number of months.