Qiyao Zhou

Z5379852

Question 3

From this problem, we can use dynamic programming in the question.

Subproblems: First, we can assume that H= $[h_1, h_2, h_3 \ldots h_\mathrm{n}]$ and $h_\mathrm{i} < h_{\mathrm{i}+1}$. We can solve this problem by considering the subproblem sum[x][y] for 0<=x<=n and 0<=y<=k which means the number of arrangements where x people are placed in the queue and the annoyance value is y.

Recurrence: $\mathrm{sum[x][y]} = \sum_{z=\max(y-x+1,0)}^{y} \mathrm{sum[x-1][z]}.$

In simple terms, for every person added to the queue (i people in the queue for ease of description), there are a total of i+1 scenarios, and each arrangement must yield a different annoyance value.

For example, adding h_3 to $[h_1, h_2]$ with a current annoyance value of 1, and arranging h_3 into the queue with a total of 3 scenarios, the resulting new annoyance value could be 1($[h_3, h_1, h_2]$), 2($[h_1, h_3, h_2]$), 3($[h_1, h_2, h_3]$). At this point the number of newly generated number of arrangements is inherited from previous number.

Thus, the number of queue scenarios with annoyance value j when person i is added is equal to the total number of scenarios when person i-1 is added with annoyance value less than or equal to j and not less than j-i+1(ensure accessibility).

Base case: sum [i][0] = 1 and sum [i][j] = 0 for 0<=i<=n,1<=j<=k.

Final answer: At the end of the algorithm we get sum [n][k] which is the number of arrangements of the queue resulting in a total annoyance of exactly k.

Time complexity: 0<=x<=n, 0<=y<=k. So, we need to compute nk times. Every time we need o (1). So, the overall time complexity is o (nk).