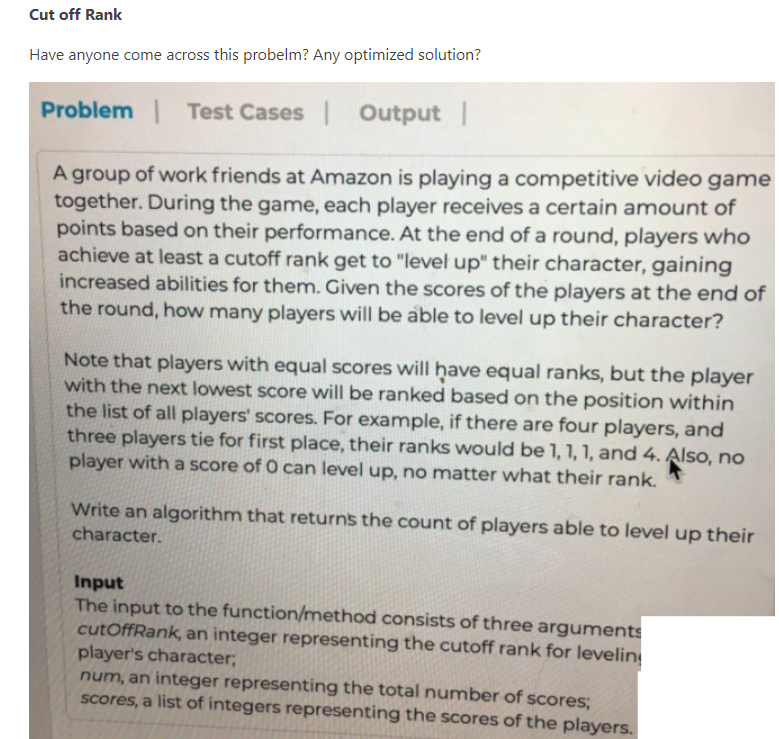
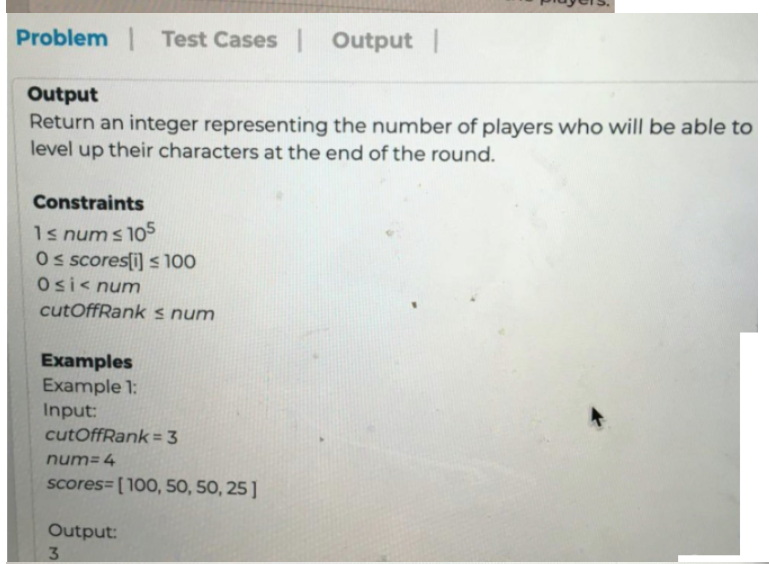
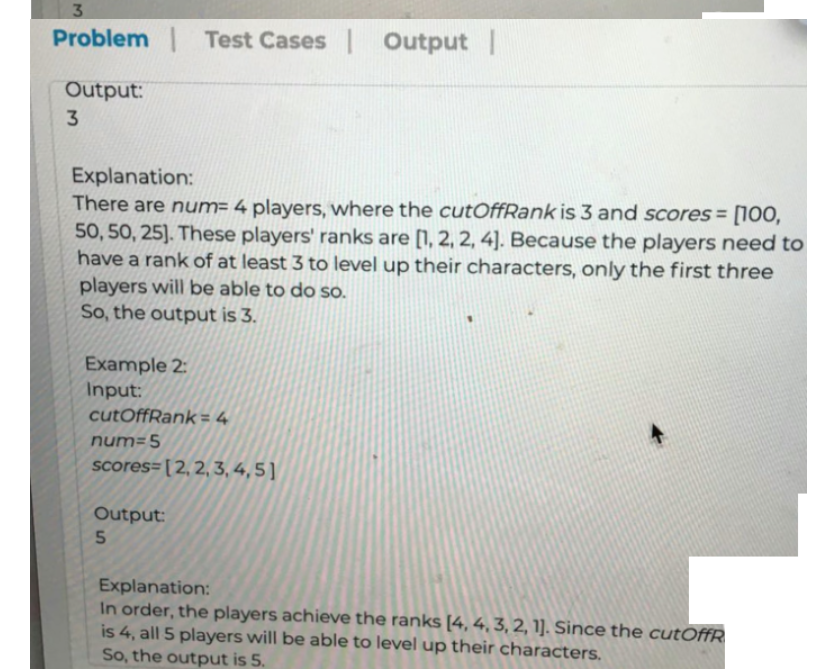
Amazon | OA 2020 | Cut - off Rank

LEET CODE DISCUSS : <https://leetcode.com/discuss/interview-question/890290/>







Time complexity: O(nlogn)  
Space complexity: O(logn)/O(n)

public static int cutOffRank(int cutOffRank, int num, int[] scores) {

int rank = 1;

int position = 1;

Arrays.sort(scores);

for (int i = num - 1; i >= 0; i--) {

if (i == num - 1 || scores[i] != scores[i + 1]) {

rank = position;

if (rank > cutOffRank) return position - 1;

}

position++;

}

return num;

}

2nd possible ans :

Time complexity: O(N)  
Space complexity: O(1). O(1) because count array of constant size is used

public int CutOffRank(int cutOffRank, int num, int[] scores)

{

int[] count = new int[101];

foreach(int s in scores) // count numbers of players for each score 0 - 100

{

count[s]++;

}

int rank = 1;

// start from score rank and assign rank based on number of players found for a particular rank

for(int i = count.Length - 1; i >= 0 ;i --)

{

if(count[i] != 0)

{

int counts = count[i];

count[i] = rank;

rank += counts; // increment current rank by number of players for this score

}

}

int numPlayers = 0;

foreach(int s in scores) // for each score check whether rank is below threshold

{

if(count[s] <= cutOffRank && s > 0)

numPlayers++;

}

return numPlayers;

}