You can view this report online at : https://www.hackerrank.com/x/tests/1178714/candidates/28865956/report

Full Name:	Valliammai N
Email:	valliammai.usha@gmail.com
Test Name:	**Assessment 4**
Taken On:	15 Aug 2021 19:02:50 IST
Time Taken:	74 min 52 sec/ 75 min
Work Experience:	> 5 years
Invited by:	Babu
Invited on:	15 Aug 2021 18:57:26 IST
Skills Score:	Problem Solving (Basic) 78/100 Problem Solving (Intermediate) 16/75
Tags Score:	Algorithms 16/75 Dynamic Programming 16/75 Easy 78/100 Greedy Algorithms 28/50 Interviewer Guidelines 78/100 Math 16/75 Medium 16/75 Problem Solving 16/75 Sorting 50/50

53.7%

94/175

scored in **Assessment 4** in 74 min 52 sec on 15 Aug 2021 19:02:50 IST

Recruiter/Team Comments:

No Comments.

Question Description	Time Taken	Score	Status
Q1 Cut Them All > Coding	37 min 18 sec	28/ 50	✓
Q2 K-Subarrays > Coding	32 min 15 sec	16/ 75	✓
Q3 Programming Contest > Coding	5 min 20 sec	50/ 50	✓

QUESTION 1

Correct Answer

Score 28

Cut Them All > Coding

Greedy Algorithms | Easy | Interviewer Guidelines

QUESTION DESCRIPTION

An automated cutting machine is used to cut rods into segments. The cutting machine can only hold a rod of *minLength* or more. A rod is marked with the necessary cuts and their lengths are given as an array in the order they are marked. Determine if it is possible to plan the cuts so the last cut is from a rod at least *minLength* units long.

Example
n = 3
lengths = [4, 3, 2]
minLength = 7

The rod is initially *sum(lengths) = 4 + 3 + 2 = 9* units long. First cut off the segment of length *4 + 3 = 7* leaving a rod *9 - 7 = 2.* Then check that the length *7* rod can be cut into segments of lengths *4* and *3.* Since *7* is greater than or equal to *minLength = 7,* the final cut can be made. Return *"Possible".*

Example
n = 3
lengths = [4, 2, 3]
minLength = 7

The rod is initially *sum(lengths) = 4 + 2 + 3 = 9* units long. In this case, the initial cut can be of length *4* or *4 + 2 = 6.* Regardless of the length of the first cut, the remaining piece will be shorter than *minLength.* Because *n - 1* = 2 cuts cannot be made, the answer is "*Impossible*."

Function Description
Complete the function *cutThemAll* in the editor below.

cutThemAll has the following parameter(s):
int lengths[n]: the lengths of the segments, in order
int minLength: the minimum length the machine can accept

Returns
string: "*Possible*" if all *n-1* cuts can be made. Otherwise, return the string "*Impossible*"

Constraints
- $2 \le n \le 10^5$
- $1 \le t \le 10^9$
- $1 \le lengths[i] \le 10^9$
- The sum of the elements of *lengths* equals the uncut rod length.

▼ Input Format For Custom Testing

The first line contains an integer, *n*, the number of elements in *lengths*.
Each line *i* of the *n* subsequent lines (where $0 \le i < n$) contains an integer, *lengths[i]*.
The next line contains an integer, *minLength*, the minimum length accepted by the machine.

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
4     →   lengths[] size n = 4
3     →   lengths[] =  [3, 5, 4, 3]
5
4
3
9     →   minLength= 9
```

Sample Output

```
Possible
```

Explanation

The uncut rod is *3 + 5 + 4 + 3 = 15* units long. Cut the rod into lengths of *3 + 5 + 4 = 12* and *3.* Then cut

the *12* unit piece into lengths *3* and *5 + 4 = 9*. The remaining segment is *5 + 4 = 9* units and that is long enough to make the final cut.

▼ Sample Case 1

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
3     →   lengths[] size n = 3
5     →   lengths[] =  [5, 6, 2]
6
2
12    →   minLength= 12
```

Sample Output

```
Impossible
```

Explanation

The uncut rod is *5 + 6 + 2 = 13* units long. After making either cut, the rod will be too short to make the second cut.

INTERVIEWER GUIDELINES

▼ Hint 1

Since n - 1 cuts are required, two rods remain and they are consecutive.

▼ Hint 2

For each pair of consecutive rods, find the maximum sum of the lengths of the rods. If it is greater than t, it is possible to cut them.

▼ Solution

Concepts covered: Basic Programming Skills, Loops, Arrays, Greedy, Problem Solving. The problem tests the candidate's ability to use loops, arrays, and greedy techniques. It requires the candidate to come up with an algorithm to check if an array of given lengths of the rod can be cur form left and right exactly n - 1 time and the remaining size of the rod is at least minLength in a constrained time and space complexity.

Optimal Solution:
Sort the whole array in non-descending order and the last k + 1 indexes are the ones that can be forced upon to be chosen.
Time Complexity: O(NlogN)

```
def cutThemAll(lengths, t):

    mx = 0
    # find the length of the longest adjoining pair of sticks
    for i in range(len(lengths) - 1):
        mx = max(mx, lengths[i] + lengths[i + 1])
    # if there is a pair long enough to hold at the end,
    if mx >= t:
        return "Possible"
    else:
        return "Impossible"
```

Brute Force Solution:
Since N - 1 cuts are to be made we need to remove exactly N - 2 elements from both the prefix and suffix of the array. So for each possible lengths of the prefix and suffix remove the N - 2 rods.
Time Complexity: O(N^2)

```
def cutThemAll(lengths, t):
```

```
    n = len(lengths)
    tot = sum(lengths)
    sum1 = 0
    for i in range(n - 1):
        rem = n - 2 - i
        sum2 = 0
        for j in range(n - rem, n):
            sum2 = sum2 + lengths[j]
        if(tot - sum1 - sum2 >= t):
            return "Possible"
        sum1 = sum1 + lengths[i]
    if(tot - sum1 >= t):
        return "Possible"
    else:
        return "Impossible"
```

Error Handling:

1. The maximum needs to be taken only for 2 consecutive elements in the array.
2. The answer is "Possible" only if the maximum consecutive sum is at least t.

▼ Complexity Analysis

Time Complexity - O(N)

We do linear time operations on the whole array.

Space Complexity - O(1)

No extra space required.

CANDIDATE ANSWER

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'cutThemAll' function below.
     *
     * The function is expected to return a STRING.
     * The function accepts following parameters:
     *  1. LONG_INTEGER_ARRAY lengths
     *  2. LONG_INTEGER minLength
     */

    public static String cutThemAll(List<Long> lengths, long minLength) {
    // Write your code here
     int sum = 0;
     int cSum=0;
    for(int i=0;i<lengths.size();i++){
        sum+=lengths.get(i);
    }
        if(minLength<=sum){
            for(int i=0;i<lengths.size();i=i++){

                List<Long> cutRod=lengths.subList(i, lengths.size()-1);

                 for(int j=0;j<cutRod.size();j++){
                        cSum+=cutRod.get(j);
                    }
                if(cSum>= minLength){
                    if(i==lengths.size()-1)
                        return "Possible";
```

```
                    else
                        continue;
                }else
                    return "Impossible";

            }
        }

    return "Impossible";
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Wrong Answer	0	3.7787 sec	199 KB
TestCase 1	Easy	Sample case	Success	1	0.1251 sec	30.1 KB
TestCase 2	Easy	Sample case	Wrong Answer	0	3.7851 sec	199 KB
TestCase 3	Easy	Sample case	Success	3	0.1783 sec	36.2 KB
TestCase 4	Easy	Sample case	Wrong Answer	0	0.1993 sec	35.7 KB
TestCase 5	Easy	Hidden case	Success	4	0.227 sec	35.9 KB
TestCase 6	Easy	Hidden case	Wrong Answer	0	0.1783 sec	34.8 KB
TestCase 7	Easy	Hidden case	Success	4	0.1876 sec	36.1 KB
TestCase 8	Easy	Hidden case	Success	4	0.3952 sec	56.6 KB
TestCase 9	Easy	Hidden case	Wrong Answer	0	0.4586 sec	56.1 KB
TestCase 10	Easy	Hidden case	Success	4	0.4033 sec	57.5 KB
TestCase 11	Easy	Hidden case	Wrong Answer	0	0.3836 sec	56.9 KB
TestCase 12	Easy	Hidden case	Success	4	0.4131 sec	57.1 KB
TestCase 13	Easy	Hidden case	Wrong Answer	0	0.4009 sec	57.2 KB
TestCase 14	Easy	Hidden case	Success	4	0.3574 sec	56.9 KB

No Comments

QUESTION 2

Correct Answer

Score 16

K-Subarrays › Coding

Medium | Math | Dynamic Programming | Algorithms | Problem Solving

QUESTION DESCRIPTION

A *k-subarray* of an array is defined as follows:

- It is a subarray, i.e. made of contiguous elements in the array
- The *sum* of the subarray elements, *s*, is evenly divisible by *k*, i.e.: *sum mod k = 0*.

Given an array of integers, determine the number of *k-subarrays* it contains. For example, *k = 5* and the array *nums = [5, 10, 11, 9, 5]*. The *10 k-subarrays* are: *{5}, {5, 10}, {5, 10, 11, 9}, {5, 10, 11, 9, 5}, {10}, {10, 11, 9}, {10, 11, 9, 5}, {11, 9}, {11, 9, 5}, {5}*.

Function Description

Complete the function *kSub* in the editor below. The function must return a long integer that represents the number of *k-subarrays* in the array.

number of k-subarrays in the array.

kSub has the following parameter(s):

k: the integer divisor of a k-subarray

nums[nums[0],...nums[n-1]]: an array of integers

Constraints

- $1 \le n \le 3 \times 10^5$
- $1 \le k \le 100$
- $1 \le nums[i] \le 10^4$

▼ Input Format For Custom Testing

Input from stdin will be processed as follows and passed to the function.

The first line contains an integer, *k,* the number the sum of the subarray must be divisible by.
The next line contains an integer, *n*, that denotes the number of elements in *nums*.
Each line *i* of the *n* subsequent lines (where $0 \le i < n$) contains an integer that describes *nums[i]*.

▼ Sample Case 0

Sample Input For Custom Testing
Sample Input 0

```
3
5
1
2
3
4
1
```

Sample Output 0

```
4
```

Explanation 0

The *4* subarrays of *nums* having sums that are evenly divisible by *k = 3* are *{3}*, *{1, 2}*, *{1, 2, 3}*, *{2, 3, 4}*.

CANDIDATE ANSWER

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'kSub' function below.
     *
     * The function is expected to return a LONG_INTEGER.
     * The function accepts following parameters:
     *  1. INTEGER k
     *  2. INTEGER_ARRAY nums
     */

    public static long kSub(int k, List<Integer> nums) {
    // Write your code here
    int count=0;
       int[] num=new int[nums.size()];
       int left=0,right=0;
       for(int i=0;i<nums.size();i++){
```

```
            num[i]=nums.get(i);
        }
        while(left<nums.size()){
             int sum=num[left];
             while(right<nums.size()){
                 if(left==right){
                     if(num[right]%k==0) count++;
                 }else{
                     sum+=num[right];
                     if(sum%k==0) count++;
                 }
                 right++;
             }
         left++;
         right=left;
        }
    return count;
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
Testcase 0	Easy	Sample case	Success	1	0.1554 sec	29.7 KB
Testcase 1	Easy	Hidden case	Success	4	0.1281 sec	30.2 KB
Testcase 2	Easy	Hidden case	Success	4	0.1599 sec	31.2 KB
Testcase 3	Easy	Hidden case	Success	5	0.4454 sec	40.6 KB
Testcase 4	Medium	Hidden case	Terminated due to timeout	0	4.0085 sec	99.5 KB
Testcase 5	Medium	Hidden case	Terminated due to timeout	0	4.0045 sec	56.7 KB
Testcase 6	Medium	Hidden case	Terminated due to timeout	0	4.0065 sec	56.6 KB
Testcase 7	Easy	Hidden case	Terminated due to timeout	0	4.0049 sec	56.8 KB
Testcase 8	Easy	Hidden case	Terminated due to timeout	0	4.0057 sec	55 KB
Testcase 9	Easy	Hidden case	Terminated due to timeout	0	4.0054 sec	78.9 KB
Testcase 10	Easy	Sample case	Success	1	0.1243 sec	30.3 KB
Testcase 11	Easy	Sample case	Success	1	0.148 sec	30.1 KB

No Comments

QUESTION 3

Correct Answer

Score 50

Programming Contest › Coding

Easy | Sorting | Interviewer Guidelines

QUESTION DESCRIPTION

A programming organization is planning a contest for several programmers, each of which has a certain rating. (The higher the rating, the better the programmer.) Each programmer is paired with another programmer, and the difference between their ratings is referred to as the "bias amount". Given the ratings

programmer, and the difference between their ratings is referred to as the "bias amount". Given the ratings of all the programmers in the contest, what is the minimum total bias amount that can be achieved by optimally planning the programmer pairs?

Example
n = 4
ratings = [4, 2, 5, 1]

The optimal solution is:
Pair the second programmer (2) with the fourth (1) for a difference of 1.
Pair the first programmer (4) with the third (5) for a difference of 1.
This results in a total bias amount of 1 + 1 = 2.

Function Description
Complete the function *minimizeBias* in the editor below.

minimizeBias has the following parameter:
int *ratings[n]:* the ratings of each of the programmers
Returns:
int: the minimum total bias amount that can be achieved in the contest

Constraints

- $1 \le n \le 10^5$
- $1 \le ratings[i] \le 10^9$
- *n* is even.

▼ Input Format For Custom Testing

The first line contains an integer, *n*, the number of elements in *ratings*.
Each line *i* of the *n* subsequent lines (where $0 \le i < n$) contains an integer, *ratings[i]*.

▼ Sample Case 0

Sample Input For Custom Testing

```
STDIN     Function
-----     --------
4     →   ratings[] size n = 4
1     →   ratings = [1, 3, 6, 6]
3
6
6
```

Sample Output

```
2
```

Explanation
The optimal solution is to pair the first programmer (1) with the second (3) for a difference of 2, and the third programmer (6) with the fourth (6) for a difference of 0. This results in a total bias amount of 2.

▼ Sample Case 1

Sample Input For Custom Testing

```
STDIN    Function
-----    --------
6    →   ratings[] size n = 6
2    →   ratings = [2, 4, 5, 3, 7, 8]
4
5
3
7
8
```

Sample Output

```
3
```

Explanation

The optimal solution is to assign the following pairs: (2,3), (4,5), and (7,8). This results in the least total bias amount, which is 3.

INTERVIEWER GUIDELINES

▼ Hint 1

Sort the ratings of the programmers.

▼ Hint 2

Pair adjacent programmers after sorting their ratings.

▼ Solution

Concepts Covered: Basic Programming Skills, Loops, Sorting, Problem Solving. The problem tests the candidate's ability to use loops and sorting. It requires the candidate to come up with an algorithm to find minimum sum of the difference in the ratings of the programmers when they are paired with one other member in a constrained time and space complexity.

Optimal Solution:

At each step, choose the programmer with the highest rating and then select another programmer with a rating nearest to it. So basically, we can sort the whole array of ratings and pair up adjacent programmers.

Time Complexity: O(NlogN)

Pseudo Code:

```
def minimizeBias(ratings):
    # Write your code here
    ratings.sort();
    sum = 0
    for i in range(0, len(ratings), 2):
        sum += ratings[i + 1] - ratings[i]
    return sum
```

Brute Force Approach: Start with a programmer with the highest rating which is not marked till now and find the programmer with the smallest difference in the rating. Mark both of them as visited and go on doing step1 until there are no more unmarked programmers. Time Complexity: O(n^2).

Error Handling:

1. It's important to sort the given array initially.
2. We must take sum of the differences between the adjacent ratings only.

▼ Complexity Analysis

Time Complexity - O(NlogN).

We require O(NlogN). time for sorting the array and O(N) time to then process the array for the answer.

Space Complexity - O(1) - No extra space is required.

▼ Follow up Question

What if the problem were modified a bit to pair up the programmers in such a way such that the total bias is maximal.

Solution: It's then optimal to sort the array and pair up the most distant programmers, i.e pair up (1, N), (2, N - 1), ... (N / 2, N / 2 + 1).

Psuedo Code -

```
def minimizeBias(ratings):
    # Write your code here
```

```
    ratings.sort();
    n = len(ratings)
    sum = 0
    for i in range(0, n // 2):
        sum += ratings[n - i - 1] - ratings[i]
    return sum
```

CANDIDATE ANSWER

Language used: **Java 8**

```
class Result {

    /*
     * Complete the 'minimizeBias' function below.
     *
     * The function is expected to return an INTEGER.
     * The function accepts INTEGER_ARRAY ratings as parameter.
     */

    public static int minimizeBias(List<Integer> ratings) {
    // Write your code here
        int[] rating=new int[ratings.size()];
        for(int i=0;i<ratings.size();i++){
            rating[i]=ratings.get(i);
        }
        Arrays.sort(rating);
        int count=0;
        for(int i=0;i<rating.length-1;i=i+2){
            count+=rating[i+1]-rating[i];
        }
        return count;
    }

}
```

TESTCASE	DIFFICULTY	TYPE	STATUS	SCORE	TIME TAKEN	MEMORY USED
TestCase 0	Easy	Sample case	Success	1	0.1376 sec	30.2 KB
TestCase 1	Easy	Sample case	Success	1	0.1542 sec	30.2 KB
TestCase 2	Easy	Sample case	Success	1	0.1352 sec	30.1 KB
TestCase 3	Easy	Sample case	Success	2	0.1486 sec	30.3 KB
TestCase 4	Easy	Hidden case	Success	2	0.1377 sec	30.3 KB
TestCase 5	Easy	Sample case	Success	3	0.162 sec	31.2 KB
TestCase 6	Easy	Hidden case	Success	3	0.1634 sec	31.3 KB
TestCase 7	Easy	Hidden case	Success	3	0.163 sec	30.9 KB
TestCase 8	Easy	Hidden case	Success	3	0.1708 sec	31.1 KB
TestCase 9	Easy	Hidden case	Success	3	0.1697 sec	31.3 KB
TestCase 10	Easy	Hidden case	Success	5	0.3977 sec	55.2 KB
TestCase 11	Easy	Hidden case	Success	5	0.4273 sec	55.4 KB
TestCase 12	Easy	Hidden case	Success	6	0.4444 sec	56.5 KB
TestCase 13	Easy	Hidden case	Success	6	0.3852 sec	55.9 KB
TestCase 14	Easy	Hidden case	Success	6	0.4069 sec	57.5 KB

No Comments

PDF generated at: 15 Aug 2021 14:49:26 UTC