The balloon shooting problem

Samsung

September 18, 2019

A person P is playing a game in which she has to shoot balloons using a game-gun. There are k balloons arranged in a row. P is standing in front of that row far away. Each balloon is marked with a number b_i written on it, $1 \leq i \leq k, b_i \in \mathbb{N}$.

P is given k chances to shoot. At each time a balloon is shot, points are computed. Each time a balloon is shot, all the balloons are replaced a bit in such a way that the distance between them remains same, for all adjacent pairs. If no balloon is shot, total score is zero. The points are computed in the following way.

If a balloon b_i is shot and it has neighbours b_{i-1} and b_{i+1}, then the points $b_{i-1} \times b_{i+1}$ from this shot shall be added to the total score. If a balloon b_i has only one neighbour b_j, then the points b_j shall be added to the total score. This may happen if the shot balloon is the first balloon or the last balloon in the row. If there is only one balloon b_i then the points b_i shall be added to the total score.

For example, if there are $k = 4$ balloons, b_1, b_2, b_3, b_4 sequenced in a row and the numbers $2, 3, 4, 5$ are marked on them respectively. Let us assume that b_2 is shot first. b_2 has two neighbours b_1 and b_3, so $b_1 \times b_3 = 2 \times 4 = 8$ is added to the score. The total score is 8. Now the balloons b_1, b_3, b_4 are left in the row. Now let us assume that the balloon b_4 is shot. b_4 has only one neighbour, b_3, so $b_3 = 4$ points are added to the total score. The total score now is 12. Now balloons, b_1, b_3 are left in the row. Now let us assume that the balloon b_3 is shot. b_3 has only one neighbour, b_1, so $b_1 = 2$ is added to the total score. The total score now is 14. Now there is only one balloon in the row, b_1. Let us assume that b_1 is shot in the fourth shot. b_1 has no neighbour, so $b_1 = 2$ is added to the total score. The final total score in this example is 16.

For the sake of another example, let us assume another shooting sequence for the same sequence of balloons. Let us assume that b_3 is shot first. b_3 has two neighbours b_2 and b_4, so $b_2 \times b_4 = 3 \times 5 = 15$ is added to the score. The total score is 15. Now the balloons b_1, b_2, b_4 are left in the row. Now let us assume that the balloon b_2 is shot. b_2 has two neighbours b_1 and b_4, so $b_1 \times b_4 = 2 \times 5 = 10$ is added to the score. The total score is 25. Now balloons, b_1, b_4 are left in the row. Now let us assume that the balloon b_1 is shot. b_1 has only one neighbour, b_4, so $b_4 = 5$ is added to the total score. The total score now is 30. Now there is only one balloon in the row, b_4. Let us assume that b_4 is shot in the fourth shot. b_4 has no neighbour, so $b_4 = 5$ is added to the total score. The final score in this example is 35.

For this particular of sequence of 4 balloons, 35 is the maximum score that can be scored. The task is to find the maximum score that can be scored while shooting a given sequence of balloons marked on them. Construct a program to complete this task. Use any programming language of your choice.

First T has to be input from the user. There will be T test cases. Each test case shall contain two rows. The first row shall contain a value $K_m, 1 \leq m \leq T, K_m \in \mathbb{N}$, the number of balloons in a row. The second row shall contains the sequence of K_m numbers, $b_i | 1 \leq i \leq K_m, b_i \in \mathbb{N}$, each of which are marked sequentially on the row of balloons.

Sample Input:

```
4
4
1 2 3 4
4
2 3 4 5
5
1 3 10 5 2
6
564 234 874 472 937 373
```

Sample Output: Output for the given sample input

```
#1 20
#2 35
#3 60
#4 2051842
```

In any shot, more than one balloons cannot be shot.

Constrains

(1) Number of test cases $1 \leq T \leq 10$.

(2) Number of balloons b_i in each test case $1 \leq i \leq 10$.

(3) Number marked on each balloon $1 \leq b_i \leq 1000, \forall i$.

(4) Maximum data cache / memory = 254 MB.

(5) Maximum stack cache / memory = 1 MB.

(6) Maximum time to execute all test cases = 3 seconds.

(7) There may be 50 tests done on your program using different input sequences.