# ACM International Collegiate Programming Contest, Asia-Amritapuri Site, 2012

## Problem F: Denethor's Decryption of Dequeue permutations

*'Though the Stewards deemed that it was a secret kept only by themselves, long ago I guessed that here in the White Tower, one at least of the Seven Seeing Stones was preserved. In the days of his wisdom Denethor did not presume to use it, nor to challenge Sauron, knowing the limits of his own strength. But his wisdom failed; and I fear that as the peril of his realm grew he looked in the Stone and was deceived: far too often, I guess, since Boromir departed. He was too great to be subdued to the will of the Dark Power, he saw nonetheless only those things which that Power permitted him to see.'* - Gandalf.

Sauron and Saruman have been communicating from large distances using the Seeing Stones. Denethor, with great difficulty has been able to break into their channel of communication using his strength of will. Despite this however, it seems that their communication has been encrypted. Gondor's spies in Isengard have found out the encryption algorithm they use and have reported back to Denethor.

The algorithm is as follows :
We refer to a dequeue as a double-ended queue. We define a "dequeue permutation of N" as a permutation of 1 to N that can be got by starting from a dequeue having elements 1, 2, 3, ..., N (in that order with 1 at the front and N at the back) and performing any sequence of N pop_front() or pop_back() operations.

Note that not all permutations of 1 to N are dequeue permutations. For example, with N = 3, you have 3-1-2, 1-3-2 etc. as dequeue permutations whereas 2-1-3, 2-3-1 aren't (you can't have 2 right at the beginning since its not at any end of the dequeue).

If Sauron wants to encrypt the number K and send it to Saruman, he would instead send the K'th lexicographically smallest (0-based indexed) dequeue permutation of N. That is, if Sauron wanted to send '0' to Saruman, he would just send 1-2-3-...-N (since this is clearly the smallest lexicographic dequeue permutation of N).

Sauron is transmitting the size of his army to Saruman, so that they can coordinate an attack on the Men of Rohan and Gondor. Since Sauron's will is so powerful, Denethor is able to get only vague glimpses of the numbers, and he is able to remember only the first half (floor(N/2) elements) of the permutation. Further, since these images are so vague, his understanding of the numbers happens out of order. For example, Denethor may understand that the 5th number of the permutation is 4, and later on only understand that the 3rd number was 3.

Help him estimate the minimum and maximum possible size of Sauron's army (value of K), given the number N, and incremental understanding of the first half of the permutation, not necessarily in order.

NOTE 1: Since the values to be output can be rather large, output the values modulo 1,000,000,007.

NOTE 2: It may be the case that Denethor's understanding of the permutation is flawed. If it is not possible to have a dequeue permutation satisfying all the conditions seen so far, output -1.

NOTE 3: A permutation p1 is lexicographically smaller than p2 if at the first position where they differ, p1's value is smaller than p2's.

## Input (STDIN):

The first line consists of the integer T, the number of test cases.
Each test case begins with a single integer N.
This is followed by exactly floor(N/2) lines containing 2 integers each: i and j, denoting that Denethor has understood that the i'th element (1-based) of the supposed permutation is j.

## Output (STDOUT):

For each testcase, output exactly floor(N/2) lines, one for each (i,j) pair. If the recollections till now cannot all be feasible, output '-1' on a line. Else output two space-separated integers: the minimum and the maximum possible value of Sauron's army, K, that are consistent with all the observations seen so far.
Between successive test cases, there should not be any blank lines in the output.

## Constraints:

1 <= T < 3
2 <= N <= 100,000
1 <= i <= floor(N/2)
1 <= j <= N
All (i, j) pairs are distinct. And for two pairs (i1, j1) and (i2, j2), you have that i1 != i2 and j1 != j2.

Time Limit: 8s
Memory Limit: 64MB

## Sample Input:

```
2
3
1 3
32
1 1
3 2
2 32
4 4
5 5
6 6
7 7
8 8
9 9
10 10
11 11
12 12
13 13
14 14
15 15
16 3
```

## Sample Output:

```
2 3
0 73741816
536870912 805306367
536870912 805306367
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
-1
```

## Notes/Explanation of Sample Input:

For the first test case, for N = 3, there are 4 dequeue permutations, lexicographically ordered as 1-2-3, 1-3-2, 3-1-2, and 3-2-1. Denethor sees the first number of the dequeue permutation as 3, and concludes that the permutation can be either 3-1-2, or 3-2-1.

In the second test case, we see that

(a) the 1st number is 1.

(b) the 3rd number is 2 ... this means that the 2nd number has to be 32.

(c) the 2nd number is 32 ... this does not add any new information.

(d) the 4th number is 4. But this is not possible, since the 4th number can now either be 3 or 31.

Hence it is inconsistent (and none of the further observations can make it consistent).

Also notice that in the 2nd test-case, the values have been output modulo 1,000,000,007.