Code Jam 2021 - Qualification Round

Reversort Engineering

Problem

Note: The main parts of the statements of the problems "Reversort" and "Reversort Engineering" are identical, except for the last paragraph. The problems can otherwise be solved independently.

Reversort is an algorithm to sort a list of distinct integers in increasing order. The algorithm is based on the "Reverse" operation. Each application of this operation reverses the order of some contiguous part of the list.

The pseudocode of the algorithm is the following:

```
Reversort(L):
  for i := 1 to length(L) - 1
    j := position with the minimum value in L between i and length(L), inclusive
    Reverse(L[i..j])
```

After $i-1$ iterations, the positions $1,\ 2,\ \ldots,\ i-1$ of the list contain the $i-1$ smallest elements of L, in increasing order. During the i-th iteration, the process reverses the sublist going from the i-th position to the current position of the i-th minimum element. That makes the i-th minimum element end up in the i-th position.

For example, for a list with 4 elements, the algorithm would perform 3 iterations. Here is how it would process $L = [4, 2, 1, 3]$:

1. $i = 1,\ j = 3 \longrightarrow L = [1, 2, 4, 3]$
2. $i = 2,\ j = 2 \longrightarrow L = [1, 2, 4, 3]$
3. $i = 3,\ j = 4 \longrightarrow L = [1, 2, 3, 4]$

The most expensive part of executing the algorithm on our architecture is the Reverse operation. Therefore, our measure for the cost of each iteration is simply the length of the sublist passed to Reverse, that is, the value $j - i + 1$. The cost of the whole algorithm is the sum of the costs of each iteration.

In the example above, the iterations cost 3, 1, and 2, in that order, for a total of 6.

You are given a size $\mathbf{N}$ and a cost $\mathbf{C}$. Find a list of $\mathbf{N}$ distinct integers between 1 and $\mathbf{N}$ such that the cost of applying Reversort to it is exactly $\mathbf{C}$, or say that there is no such list.

Input

The first line of the input gives the number of test cases, $\mathbf{T}$. $\mathbf{T}$ lines follow. Each line describes a test case with two integers $\mathbf{N}$ and $\mathbf{C}$, the size of the wanted list and the desired cost, respectively.

Output

For each test case, if there is no list of size $\mathbf{N}$ such that applying Reversort to it costs exactly $\mathbf{C}$, output one line containing `Case #`x`: IMPOSSIBLE`, where x is the test case number (starting from 1). Otherwise, output one line containing `Case #`x`:` y_1 y_2 `...` $y_\mathbf{N}$, where x is the test case number (starting from 1) and each y_i is a distinct integer between 1 and $\mathbf{N}$, representing the i-th element of one such possible list.

If there are multiple solutions, you may output any one of them. (See "What if a test case has multiple correct solutions?" in the Competing section of the FAQ.) This information about multiple solutions will not be explicitly stated in the remainder of the 2021 contest.

Limits

Time limit: 10 seconds.
Memory limit: 1 GB.
$1 \le \mathbf{T} \le 100$.
$1 \le \mathbf{C} \le 1000$.

Test Set 1 (Visible Verdict)

$2 \le \mathbf{N} \le 7$.

Test Set 2 (Visible Verdict)

$2 \le \mathbf{N} \le 100$.

Sample

Sample Input

```
5
4 6
2 1
7 12
7 2
2 1000
```

Sample Output

```
Case #1: 4 2 1 3
Case #2: 1 2
Case #3: 7 6 5 4 3 2 1
Case #4: IMPOSSIBLE
Case #5: IMPOSSIBLE
```

Sample Case #1 is described in the statement above.

In Sample Case #2, the algorithm runs for only one iteration on the proposed output. In that iteration, reverse is applied to a sublist of size 1, therefore, its cost is 1.

In Sample Case #3, the first iteration reverses the full list, for a cost of 7. After that, the list is already sorted, but there are 5 more iterations, each of which contributes a cost of 1. Another valid output would be `7 5 4 3 2 1 6`. For that output, the first iteration has a cost of 6, the last one has a cost of 2, and all others have a cost of 1.

In Sample Case #4, Reversort will necessarily perform 6 iterations, each of which will have a cost of at least 1, so there is no way the total cost can be as low as required.