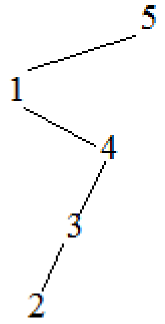
UCF Local Contest (Final Round) — September 19, 2020

Bad Tree

filename: `badtree`
Difficulty Level: `Medium-Hard`
Time Limit: `1 second`

Binary Search Trees are supposed to speed up searches for items but this happens only when the height of the tree is much less than the total number of values stored in the tree. And, in programming contests, judges unfortunately try to make the "worst case data". So invariably, in binary search tree problems, judges will make data where n nodes get inserted into a tree in a particular order so that the height of the tree is the worst case, $n - 1$.

Without loss of generality, let's assume that the n items to be inserted into a binary search tree are 1, 2, 3, …, n. For $n = 5$, if we insert the values in this order: 5, 1, 4, 3, and 2, we get the following binary search tree of height 4:



In fact, there are quite a few orderings of the first n positive integers that, when inserted into a binary search tree, create a binary search tree of height $n - 1$. Since you aspire to be a great judge one day for this contest, write a program to generate the k^{th} lexicographical permutation of the first n positive integers that, when inserted into a binary search tree in that order, generates a binary search tree of height $n - 1$.

The Problem:

Given a positive integer n, and another positive integer k, determine the k^{th} permutation, in lexicographical ordering, that when the items are inserted into a binary search tree in that order, generates a tree of height $n - 1$.

The Input:

There is only one input line; it contains two space separated integers: n ($1 \le n \le 100$), representing the number of nodes in the binary search tree, and k ($1 \le k \le 10^{18}$), where we desire the k^{th} lexicographical permutation of the first n integers which creates a binary search tree of height $n - 1$, when inserted in the order given in the permutation.

The Output:

Print the k^{th} lexicographical permutation of the integers 1 through n of the permutations which, when the values are inserted into a binary search tree, create a tree of height n – 1. Output the permutation on a single line, following each number in the permutation with a space. If no such permutation exists, output –1 instead.

Sample Input	**Sample Output**
`5 12`	`5 1 4 3 2`
`4 2`	`1 2 4 3`
`6 1`	`1 2 3 4 5 6`
`3 50`	`-1`