Parentheses Order

Problem

An **n** parentheses sequence consists of **n** (s and **n**) s.

A valid parentheses sequence is defined as the following:

You can find a way to repeat erasing adjacent pair of parentheses () *until it becomes empty.*

For example, (()) is a valid parentheses, you can erase the pair on the 2nd and 3rd position and it becomes () then you can make it empty.
)()(is not a valid parentheses, after you erase the pair on the 2nd and 3rd position, it becomes)(and you cannot erase any more.

Now, we have all valid **n** parentheses sequences. Find the **k**-th smallest sequence in lexicographical order.

For example, here are all valid 3 parentheses sequences in lexicographical order:

((()))
(()())
(())()
()(())
()()()

Input

The first line of the input gives the number of test cases, **T**. **T** lines follow. Each line represents a test case consisting of 2 integers, **n** and **k**.

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the **k**-th smallest parentheses sequence in all valid **n** parentheses sequences. Output "Doesn't Exist!" when there are less than **k** different **n** parentheses sequences.

Limits

Time limit: 30 seconds per test set.
Memory limit: 1GB.

1 ≤ **T** ≤ 100.

Small dataset (Test set 1 - Visible)

1 ≤ **n** ≤ 10.
1 ≤ **k** ≤ 100000.

Large dataset (Test set 2 - Hidden)

$1 \le n \le 100$.

$1 \le k \le 10^{18}$.

Sample

Sample Input	Sample Output
3 2 2 3 4 3 6	Case #1: ()() Case #2: ()(()) Case #3: Doesn't Exist!