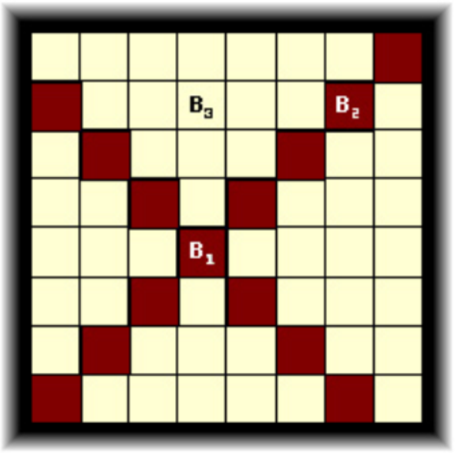
A - Little Bishops

Source file name: `bishops.py`
Time limit: 15 seconds

A bishop is a piece used in the game of chess which can only move diagonally from its current position. Two bishops attack each other if one is on the path of the other. In the figure below, the dark squares represent the reachable locations for bishop B_1 from its current position. Bishops B_1 and B_2 are in attacking position, while B_1 and B_3 are not. Bishops B_2 and B_3 are also in non-attacking position.



Given two numbers n and k, determine the number of ways one can put k bishops on an $n \times n$ chessboard so that no two of them are in attacking positions.

Input

The input file may contain multiple test cases. Each test case occupies a single line in the input file and contains two integers n ($1 \le n \le 6$) and k ($0 \le k \le n^2$).

A test case containing two zeros terminates the input.

Output

For each test case, print a line containing the total number of ways one can put the given number of bishops on a chessboard of the given size so that no two of them lie in attacking positions. You may safely assume that this number will be less than 10^{15}.

It's not allowed to include the test cases solutions in the source code. The code sent to the judge should calculate the solutions for all test cases at least once, probably, implementing a backtracking algorithm.

Sample Input

```
6 4
4 4
0 0
```

Sample Output

```
16428
260
```