PROB8: PERMS

Count the number of permutations that have a specific number of inversions.

DESCRIPTION

Given a permutation a_1, a_2, a_3,..., a_n of the *n* integers 1, 2, 3, ..., *n*, an inversion is a pair (a_i, a_j) where i < j and $a_i > a_j$. The number of inversions in a permutation gives an indication on how "unsorted" a permutation is. If we wish to analyze the average running time of a sorting algorithm, it is often useful to know how many permutations of *n* objects will have a certain number of inversions.

In this problem you are asked to compute the number of permutations of *n* values that have exactly *k* inversions.

For example, if *n* = 3, there are 6 permutations with the indicated inversions as follows:

123	0 inversions
132	1 inversion (3 > 2)
213	1 inversion (2 > 1)
231	2 inversions (2 > 1, 3 > 1)
312	2 inversions (3 > 1, 3 > 2)
321	3 inversions (3 > 2, 3 > 1, 2 > 1)

Therefore, for the permutations of 3 things

◊ 1 of them has 0 inversions
◊ 2 of them have 1 inversion
◊ 2 of them have 2 inversions
◊ 1 of them has 3 inversions
◊ 0 of them have 4 inversions
◊ 0 of them have 5 inversions
◊ etc.

INPUT: prob8.dat

The input consists one or more problems. The input for each problem is specified on a single line, giving the integer *n* (1 <= *n* <= 15) and a non−negative integer *k* (1 <= *k* <= 200). The end of input is specified by a line with n = k = 0.

An example input file would be

```
       column   1
       1234567890
line 1:3 0[EOL]
     2:3 1[EOL]
     3:3 2[EOL]
     4:3 3[EOL]
     5:4 2[EOL]
     6:4 10[EOL]
     7:13 23[EOL]
     8:18 80[EOL]
     9:0 0[EOL]
      :[EOF]
```

OUTPUT: `prob8.out`

For each problem, output the number of permutations of {1, ..., *n*}with exactly *k* inversions.

The correct output corresponding to the example input file would be

```
       column   11111111112222222223
       123456789012345678901234567890
line 1:Program 8 by team 0[EOL]
     2:1[EOL]
     3:2[EOL]
     4:2[EOL]
     5:1[EOL]
     6:5[EOL]
     7:0[EOL]
     8:46936280[EOL]
     9:184348859235088[EOL]
    10:End of program 8 by team 0[EOL]
      :[EOF]
```

NOTES

Even though only integer arithmetic is performed, use double precision values to represent quantities to avoid overflows.