In combinatorial optimization, the Hamming distance is a common way to calculate the distance of two binary patterns. The definition of Hamming distance is the number of different bits between the two patterns. For example, the two patterns A = 0010 and B = 1001 have Hamming distance 3.
Write a program to enumerate all possible binary patterns which have length N and Hamming distance H to the origin, a string contains N zeros. This can be completed by using a recursive function with following recursive equations:

$$S_{N,H} = \begin{cases} 0 + S_{N-1,H} \text{ and } 1 + S_{N-1,H-1} & N > H > 0 \\ 0 \ldots 0 \; (N \text{ zeros}) & N > H = 0 \\ 1 \ldots 1 \; (N \text{ ones}) & N = H > 0 \end{cases}$$

Input

The input has several cases indicated by the first input and ends with EOF. Each case contains two integers which in turn represent N and H.

Output

For each case, output all length-N binary patterns with Hamming distance H to the origin in ascending order. Each two consecutive cases should be separated by a newline character.

Sample Input

```
4
1 1
2 1
2 2
3 1
```

Sample Output

```
1

01
10

11

001
010
100
```