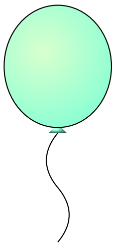
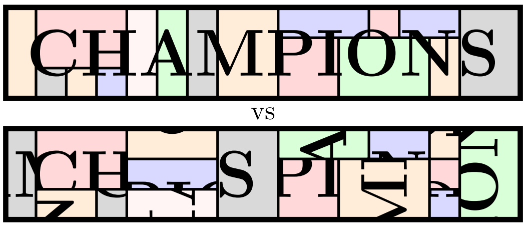


L: Broken trophy

Time limit: 1 second



Coming back home after triumphally winning your long-coveted trophy, you discover that it was shattered to pieces in your trunk. It just remains to repair it.

Your trophy had the shape of a rectangle of size $3 \times N$, for some integer $N \geqslant 1$, thereby consisting of 3 lines and N columns, containing a total of $3N$ unit squares. It was broken into K pieces, the k^{th} piece being a rectangle of size $A_k \times B_k$ for some integers A_k and B_k such that $1 \leqslant A_k \leqslant B_k \leqslant 3$. Such pieces may have been rotated, or even flipped, in the havoc that is your trunk.

As the first step towards repairing your trophy, you should reassemble them in the form of a rectangle of size $3 \times N$. More precisely, you have drawn, on a sheet of paper, a $3 \times N$ rectangle on which you will place your K pieces, and you need to know, for all integers $i \leqslant 3$ and $j \leqslant N$, which piece will cover the unit square on the i^{th} line and j^{th} column of your rectangle.

Input

The input consists of three lines, each one containing space-separated integers. The first line contains the numbers K and N. The second line contains the numbers $A_1, A_2, \ldots, A_K$. The third line contains the numbers $B_1, B_2, \ldots, B_K$.

Output

The output should contain three lines, each one consisting of N space-separated integers. If you plan to cover the unit square on the i^{th} line and j^{th} column with the k^{th} piece, the j^{th} number on the i^{th} output line should be the integer k.

In case there are several ways to reassemble your pieces in the form of a rectangle of size $3 \times N$, every output representing one of these ways is considered correct.

Limits

- $1 \leqslant K \leqslant 300\,000$;
- $1 \leqslant N \leqslant 100\,000$;
- $1 \leqslant A_k \leqslant B_k \leqslant 3$ for all $k \leqslant K$;
- the pieces described in the input can be reassembled in the form of a rectangle of size $3 \times N$.

Sample Input 1

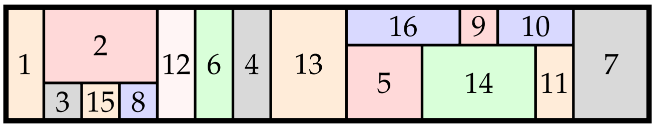
```
16 17
1 2 1 1 2 1 2 1 1 1 1 1 2 2 1 1
3 3 1 3 2 3 3 1 1 2 2 3 3 3 1 3
```

Sample Output 1

```
1 2 2 2 12 6 4 13 13 16 16 16 9 10 10 7 7
1 2 2 2 12 6 4 13 13 5 5 14 14 14 11 7 7
1 3 15 8 12 6 4 13 13 5 5 14 14 14 11 7 7
```

Sample Explanation 1

This output represents the following reassembling:



Sample Input 2

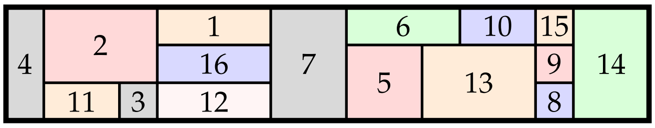
```
16 17
1 2 1 1 2 1 2 1 1 1 1 1 2 2 1 1
3 3 1 3 2 3 3 1 1 2 2 3 3 3 1 3
```

Sample Output 2

```
4 2 2 2 1 1 1 7 7 6 6 6 10 10 15 14 14
4 2 2 2 16 16 16 7 7 5 5 13 13 13 9 14 14
4 11 11 3 12 12 12 7 7 5 5 13 13 13 8 14 14
```

Sample Explanation 2

This output represents the following reassembling:



Both reassemblings are valid, even though sample inputs 1 and 2 are the same.