Inaho VII

Time Limit: 5.0s **Memory Limit:** 256M

Inaho is finally home! After being stuck in an N-dimensional for so long, he has started to comprehend the complexities of N dimensions and have thought of a problem. Unfortunately, he does not know of a solution as he is not yet a master of N dimensions, so he has asked you for help.

Given an N-dimensional array, and a type of operation:

Type 1 operation: print the normal array given the Binary Indexed Tree.

Type 2 operation: print the Binary Indexed Tree given the normal array.

Input Specification

The first line will contain two space-separated integers N $(1 \leq N \leq 10)$, and T $(1 \leq T \leq 2)$, the number of dimensions and the type of operation respectively.

The second line will contain L^N $(L = \lfloor \sqrt[N]{5 \times 10^6} \rfloor)$ space-separated integers $(-100 < a_i < 100)$, which are the values in the array. If $T = 1$, this array is the Binary Indexed Tree, and if $T = 2$, this array is the normal array.

a_i represents the value at position p_i where $(p_i = \sum_{n=1}^{N} (p_{i_n} \times L^{N-n}))$.

In other words, the second line will be a flattened N-dimensional grid.

Output Specification

The normal array, if the $T = 1$, or the Binary Indexed Tree, if $T = 2$.

The output should follow the same format as the second line of the input. That is, the i^{th} integer should represent the value at p_i where $(p_i = \sum_{n=1}^{N} (p_{i_n} \times L^{N-n}))$.

Subtasks

Subtask 1 [10%]

$N = 1$

Subtask 2 [20%]

$T = 1$

Subtask 3 [20%]

$T = 2$

Subtask 4 [50%]

No further constraints.

Sample Cases Note

For the sake of not having 10 million characters in the sample cases and for the sample cases to be useful, $L = \lfloor \sqrt[N]{20} \rfloor$. **In the actual test cases**, however, L will follow the constraints as stated in the Input Specification (that is, $L = \lfloor \sqrt[N]{5 \times 10^6} \rfloor$).

Sample Input 1

```
1 1
1 2 1 4 1 2 1 8 1 2 1 4 1 2 1 16 1 2 1 4
```

Sample Output 1

```
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Sample Input 2

```
2 2
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Sample Output 2

```
1 2 1 4 2 4 2 8 1 2 1 4 4 8 4 16
```

Sample Input 3

```
4 1
1 9 3 2 4 7 5 8 2 9 1 8 4 9 2 10
```

Sample Output 3

```
1 8 2 -9 3 -5 -1 9 1 -1 -3 9 -1 3 0 -6
```