Sashka is going on an excursion. She has prepared backpacks, each of which has a given capacity . Some of them already contain items left over from the previous trip. There are types of items, with an unlimited amount of each type. Item of type brings pleasure if it is placed in backpack and takes up volume (the pleasure of a given item depends on the backpack the item is in). The pleasure of an item may be negative, and it means that it removes  pleasure.

It wasn't long before Sashka got exhausted from too much packing, and then, by pure chance, Kyusho appeared out of nowhere with a special device for just such occasions – a teleportatinator. It supports 4 functions, each having a given cost:

1. Buying an item of type and adding it to backpack for a cost .
2. Removing an item of type from backpack and discarding it for a cost .
3. Swapping two items, one of type from backpack , the other of type from backpack for a cost . The swap takes place immediately - there isn’t a moment when both items are in the same bag together.
4. Taking items of type from backpack and placing them in backpack for cost , where denotes the biggest integer smaller or equal to .

At any time, the sum of the volumes occupied by the items in a backpack should not exceed its capacity, and the items being removed from a backpack should be there in advance. If any of these conditions is not met, the teleportatinator self-destructs and you get Wrong answer.

Let denote the sum of the pleasures that the items from all backpacks bring (for items that bring pleasures 3,5 and -2, S = 3 + 5 + (- 2) = 6) after the teleportatinator executes all functions; denotes the sum of pleasures in the beginning; and  denotes the sum of the costs of the functions used. Sashka wants to maximize the difference . As a good friend of Sashka's, Harry decides to help her pack her backpacks by writing a program that finds the optimal strategy, but as you may have guessed, his laziness got in the way. Help Harry by writing a program teleportatinator that helps Sashka pack her luggage in an optimal way.

**Input**

The first line of teleportatinator.in contains the integers , , , ,, – the number of item types, the number of backpacks and the costs of the functions. The second line of the file contains numbers – the capacities of the backpacks. The next line contains numbers – the volumes of the items. Next, there are lines containing integers each, where the -th number on the -th row is – the pleasure received by placing the -th item type in the -th backpack. Each of the last lines of the file contains a number followed by numbers with values from to – the item types in the backpacks left from the previous trip. It is guaranteed that the volumes they occupy do not exceed the capacities of the backpacks.

**Output**

On the first line of teleportatinator.out print an integer – the number of functions the teleportatinator must execute. On the next lines you should print 3 or 5 numbers, depending on the function type – if it’s the first or second type, the line should be in the 1 or 2 format. If the function is of the third type, the format must be 3 . And if the function is of the fourth type, the format should be 4 .

**Constraints**

|  |  |
| --- | --- |
| **Fraction of test cases in each subtask** | **Constraints** |
| 1/3 | **a=0, b=0** |
| 1/3 | **c=0** |
| 1/3 | **None** |

**Scoring**

The result of the participant is the final difference . Let be your result and be the maximum result among all participants. If , you will receive 0 points for the current test. Otherwise, your result for the test will be .

**Subtasks**

|  |  |  |
| --- | --- | --- |
| **Percent of tests** | **N** | **K** |
| 20% | **≤ 20** | **≤ 5** |
| 20% | **≤ 200** | **≤ 10** |
| 20% | **≤ 103** | **≤ 20** |
| 40% | **≤ 104** | **≤ 50** |

**Time Limit: 5.0 sec.**

**Memory Limit: 1024 MB.**

**Sample test case**

|  |  |
| --- | --- |
| **Input (**teleportatinator.in**)** | **Output (**teleportatinator.out**)** |
| 5 2 6 2 2 3  21 30  6 9 4 13 9  -6 11  -3 8  4 -5  1 8  17 -70  3 2 1 1  1 5 | 4  3 2 1 5 2  4 2 1 1 2  1 5 1  1 1 2 |

**Explanation:** In the end, the first backpack contains two items of type 5, and the second backpack - one of type 2 and three items of type 1. Thus , , . The final result is .