HAPPINESS

Monetary system in X-land is a bit strange. There are banknotes with values of all integer numbers from 1 to *M*. There is another strange rule in the shops of X-land – the customer can never receive change, but also cannot leave a tip – in other words the customer must always pay the exact value of his purchases. If he does not have the exact sum for his purchase, then he cannot buy. Imagine what inconvenience this creates for the customers.

Niya is a girl from X-land. Like all other persons, she constantly fights against the rules described above. She always knows her set of banknotes – let's assume their values are a_1, a_2,…,a_N. All those values are between 1 and *M* and she may possess more than one banknote of a kind. Also the sequence of values a_1, a_2,……,a_N, is not ordered in any way. Niya feels happy, when entering a shop, she may buy any combination of goods with total price equal to any number between 1 and the total sum of her banknotes $a_1 + a_2 + ... + a_N$. In that case, when she is shopping, she must only consider her total amount of money without making complicated calculations of whether she can buy (or not) with her banknotes.

Remark: Let us sort a_1, a_2,……,a_N in ascending order. Let us denote $S_i = 1+a_1+ a_2+…….+a_i$. Necessary and sufficient condition to be able to represent each number between 1 and $a_1+ a_2+…….+a_N$ as a sum of elements from the multiset a_1, a_2,……,a_N, is that the following inequality $S_i \geq a_{i+1}$ held true for each $i>1$ and $a_1 = 1$.

As expected, Niya's set of banknotes is changing after each purchase and also after each wage she receives – that's why her happiness is variable. You can help the girl with a program. Your program will receive as an input the initial set of Niya's banknotes and all the events that happen – purchases and wages. The program should be able to determine if Niya is happy in the beginning and after each event.

We should note that Niya feels happy also when she doesn't have any money – then she just skips shopping and goes jogging.

Task

Write functions *init()* and *is_happy()*, which will be compiled with jury's grader. These functions should serve to determine Niya's happiness at the beginning and after each event. The functions will receive as parameters the starting set of banknotes and the sets of banknotes that are removed from the set (on purchases) and added to the set (on receiving wage).

Implementation details

You should submit to the grading system a file **happiness.cpp,** which contains functions:

bool init(int coinsCount, long long maxCoinSize, long long coins[]).

bool is_happy(int event, int coinsCount, long long coins[]).

Parameters description:

coinsCount – number of banknotes that are received (starting set or wage) or discarded (shopping).

maxCoinSize – maximum value of one banknote.

coins[] – array, in which in random order are given values of the banknotes (index starts from 0).

event – event's type :

−1 – Shopping;
1 – Receiving wage.

The function *init* is called once by the grader at the beginning to set the starting set of Niya's banknotes and then grader calls *Q* times function *is_happy* with *event* = −1 (shopping) or *event* = 1 (wage). After each call the called function should return *true*, if Niya feels happy with her current set of banknotes or *false* if she doesn't.

File **happiness.cpp** should NOT contain function *main()*, but can contain other declarations and functions, necessary for correct working of functions *init* and *is_happy.* Your program should contain `#include "happiness.h"` in the beginning

Constraints

Let *Nc* denote the number of Niya's banknotes at any given moment and *K* – the number of banknotes, used in any purchase or wage. Then we have:

$0 \le Nc \le 200\,000$
$0 \le Q \le 100\,000$
$1 \le M \le 10^{12}$
$1 \le K \le 5$

It is guaranteed that in any call of *is_happy* with *event* = −1 (shopping) the set of banknotes given in *coins*[] is a subset of current Niya's set of banknotes.

Example

called function	event	coinsCount	maxCoinSize	coins[]	function returns
init		5	100	4 8 1 2 16	true
is_happy	−1(shopping)	2		2 8	false
is_happy	1(receiving wage)	3		7 9 2	true

Subtasks

Subtask	Points	Nc	M	Q
1	10	≤ 300	≤ 100	≤ 100
2	20	≤ 20000	$\le 10^{12}$	≤ 1000
3	30	≤200000	≤ 1000000	≤ 100000
4	40	≤ 200000	$\le 10^{12}$	≤ 100000

Local testing

In order to be able to test your functions *init* and *is_happy* on your local computer, you will get files *lgrader.cpp* and *happiness.h*. Compile them together with your file **happiness.cpp** and you will receive a program that you can use to test your functions.

The program expects following input format:

Single positive integers *N* and *M* are given on the first row – initial count of Niya's banknotes and maximum value of one banknote.

N positive integers are given on the second row, separated by spaces – values of banknotes in the initial set.

Non-negative integer *Q* is given on the third row – event's count.

On each of the next *Q* rows one event is described – first, a value for the event is given: −1 (shopping) or 1 (receiving of a wage). After that a positive integer *K* is given – number of banknotes that are removed or added to Niya's set. Last *K* integers are given, separated by intervals – values for the banknotes which are removed or added.

On the standard output the program will print *Q*+1 lines with 0 or 1 – "happiness" statuses of Niya at the beginning and after each event.

Example for local testing

Input	Output
5 100 4 8 1 2 16 2 -1 2 2 8 1 3 7 9 2	1 0 1