Problem B. Useful Algorithm

Input file:	`standard input`
Output file:	`standard output`
Time limit:	6 seconds
Memory limit:	1024 megabytes

Putata is learning some useful algorithm called Binary Adding these days. This algorithm allows you to calculate the sum of two m-bit binary integers.

An integer x is called to be a m-bit binary integer, if and only if $0 \le x < 2^m$. The binary representation of m is a 0-indexed sequence v of length m, where $\forall 0 \le i < m, v_i \in \{0, 1\}$, and $x = \sum\limits_{i=0}^{m-1} v_i 2^i$. It is guaranteed that for all m-bit binary integers, each one have its unique binary representation.

The Binary Adding algorithm for calculating the sum of two m-bit binary integers with binary representation $\{a_i\}_{i=0}^{m-1}, \{b_i\}_{i=0}^{m-1}$ is shown below:

Algorithm 1 BinaryAdding(a, b)

Input: input parameters $a[0, \ldots, m-1]$, $b[0, \ldots, m-1]$
Output: output result $sum[0, \ldots, m], carry[0, \ldots, m]$

```
carry[0] := 0
for i := 0 to m − 1 do
    sum[i] := a[i] ⊕ b[i] ⊕ carry[i]                                  ▷ ⊕ denotes xor operation
    carry[i + 1] := (a[i] ∧ b[i]) ∨ (a[i] ∧ carry[i]) ∨ (b[i] ∧ carry[i])   ▷ ∧ denotes and operation, ∨ denotes or operation.
sum[m] := carry[m]
return sum[0, ..., m], carry[0, ..., m]
```

In order to test if Putata really mastered this algorithm, Budada is going to prepare tests for Putata. Before preparing his tests, Budada devised a way to calculate the difficulty of a problem. Assume the problem is to calculate the sum of $\{a_i\}_{i=0}^{m-1}, \{b_i\}_{i=0}^{m-1}$, then we define the carry set of a, b, $S(a, b) = \{x | \text{when calling BinaryAdding}(a, b), carry_x = 1\}$. Budada has an integer sequence $\{w_i\}_{i=1}^{m}$, denoting the difficulty of calculation when a carry occurred at the corresponding bit. The **Carry Difficulty** is the maximum difficulty among all bits where a carry occurred. If there's no carry occurred, the **Carry Difficulty** is 0.

The problem database of Budada is an integer sequence $\{c_i\}_{i=1}^{n}$. Each integer has a corresponding **Numerical difficulty**, which is also an integer sequence $\{d_i\}_{i=1}^{n}$. Please notice that c_i are **not necessarily** pairwise distinct, and for some $c_i = c_j$, d_i **might not be equal to** d_j. A Binary Adding problem consists of two integers, so when Budada chooses c_i and c_j to set a test, the **Numerical difficulty** of this problem is $d_i + d_j$.

Budada wants to prepare the most difficult test for Putata, so he will choose two integers i, j such that $1 \le i, j \le n$ (not necessarily distinct), and use c_i, c_j to set a test. He wants to maximize the **Test Difficulty**, which is the product of the **Carry Difficulty** and **Numerical difficulty** of this test, and he asked you to tell him this production. Formally, the maximum **Test Difficulty** is $\max\limits_{1 \le i,j \le n} \{\max\{\max\{w_x | x \in S(c_i, c_j)\}, 0\} \cdot (d_i + d_j)\}$.

Budada also has q updates for his problem database, each time he will select an integer i, and modify c_i, d_i. You are asked to answer the maximum **Test Difficulty** of $q + 1$ versions of the problem database.

Input

The first line contains three integers n, m, q ($1 \le n, q \le 10^5$, $1 \le m \le 16$), corresponding to the meaning described above.

The second line contains m integers, the i-th integer is w_i ($1 \le w_i \le 10^9$).

The third line contains n integers, the i-th integer is c_i ($0 \le c_i < 2^m$).

The fourth line contains n integers, the i-th integer is d_i ($1 \le d_i \le 10^9$).

For the following q lines, each line contains three integers x, u, v, let $lastans$ denotes the maximum **Test Difficulty** of the last version of the problem database, and $x' = x \oplus lastans$, $u' = u \oplus lastans$, $v' = v \oplus lastans$ ($1 \le x' \le n$, $0 \le u' < 2^m$, $1 \le v' \le 10^9$), this represents Budada changes $c_{x'}$ to u', and $d_{x'}$ to v'. Here, "$\oplus$" denotes the bitwise XOR operator. **Please notice that you should use 64-bit integer to store x, u, v.**

Output

Output $q+1$ lines, denoting the maximum **Test Difficulty** of the $q+1$ versions of the problem database.

Example

standard input	standard output
5 3 3 1 2 4 0 0 1 2 7 10 10 5 3 1 27 24 29 20 16 19 13 8 9	24 16 8 0

Note

The decrypted operations are:

$x' = 3, u' = 0, v' = 5.$

$x' = 4, u' = 0, v' = 3.$

$x' = 5, u' = 0, v' = 1.$