D. Bear and Cavalry

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Would you want to fight against bears riding horses? Me neither.

Limak is a grizzly bear. He is general of the dreadful army of Bearland. The most important part of an army is cavalry of course.

Cavalry of Bearland consists of n warriors and n horses. i-th warrior has strength w_i and i-th horse has strength h_i. Warrior together with his horse is called a unit. Strength of a unit is equal to multiplied strengths of warrior and horse. Total strength of cavalry is equal to sum of strengths of all n units. Good assignment of warriors and horses makes cavalry truly powerful.

Initially, i-th warrior has i-th horse. You are given q queries. In each query two warriors swap their horses with each other.

General Limak must be ready for every possible situation. What if warriors weren't allowed to ride their own horses? After each query find the maximum possible strength of cavalry if we consider assignments of all warriors to all horses that no warrior is assigned to his own horse (it can be proven that for $n \geq 2$ there is always at least one correct assignment).

Note that we can't leave a warrior without a horse.

Input

The first line contains two space-separated integers, n and q ($2 \leq n \leq 30\,000$, $1 \leq q \leq 10\,000$).

The second line contains n space-separated integers, $w_1, w_2, ..., w_n$ ($1 \leq w_i \leq 10^6$) — strengths of warriors.

The third line contains n space-separated integers, $h_1, h_2, ..., h_n$ ($1 \leq h_i \leq 10^6$) — strengths of horses.

Next q lines describe queries. i-th of them contains two space-separated integers a_i and b_i ($1 \leq a_i, b_i \leq n$, $a_i \neq b_i$), indices of warriors who swap their horses with each other.

Output

Print q lines with answers to queries. In i-th line print the maximum possible strength of cavalry after first i queries.

Examples

input
4 2 1 10 100 1000 3 7 2 5 2 4 2 4

output
5732 7532

input
3 3 7 11 5 3 2 1 1 2 1 3 2 3

output

```
44
48
52
```

input

```
7 4
1 2 4 8 16 32 64
87 40 77 29 50 11 18
1 5
2 7
6 2
5 6
```

output

```
9315
9308
9315
9315
```

Note

Clarification for the first sample:

```
Warriors: 1 10 100 1000
Horses:   3  7  2    5
```

After first query situation looks like the following:

```
Warriors: 1 10 100 1000
Horses:   3  5  2    7
```

We can get $1 \cdot 2 + 10 \cdot 3 + 100 \cdot 7 + 1000 \cdot 5 = 5732$ (note that no hussar takes his own horse in this assignment).

After second query we get back to initial situation and optimal assignment is $1 \cdot 2 + 10 \cdot 3 + 100 \cdot 5 + 1000 \cdot 7 = 7532$.

Clarification for the second sample. After first query:

```
Warriors:  7 11 5
Horses:    2  3 1
```

Optimal assignment is $7 \cdot 1 + 11 \cdot 2 + 5 \cdot 3 = 44$.

Then after second query $7 \cdot 3 + 11 \cdot 2 + 5 \cdot 1 = 48$.

Finally $7 \cdot 2 + 11 \cdot 3 + 5 \cdot 1 = 52$.