Codeforces Round #470 (rated, Div. 2, based on VK Cup 2018 Round 1)

A. Protect Sheep

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Bob is a farmer. He has a large pasture with many sheep. Recently, he has lost some of them due to wolf attacks. He thus decided to place some shepherd dogs in such a way that all his sheep are protected.

The pasture is a rectangle consisting of $R \times C$ cells. Each cell is either empty, contains a sheep, a wolf or a dog. Sheep and dogs always stay in place, but wolves can roam freely around the pasture, by repeatedly moving to the left, right, up or down to a neighboring cell. When a wolf enters a cell with a sheep, it consumes it. However, no wolf can enter a cell with a dog.

Initially there are no dogs. Place dogs onto the pasture in such a way that no wolf can reach any sheep, or determine that it is impossible. Note that since you have many dogs, you do **not** need to minimize their number.

Input

First line contains two integers R ($1 \le R \le 500$) and C ($1 \le C \le 500$), denoting the number of rows and the numbers of columns respectively.

Each of the following R lines is a string consisting of exactly C characters, representing one row of the pasture. Here, '`S`' means a sheep, '`W`' a wolf and '`.`' an empty cell.

Output

If it is impossible to protect all sheep, output a single line with the word "`No`".

Otherwise, output a line with the word "`Yes`". Then print R lines, representing the pasture after placing dogs. Again, '`S`' means a sheep, '`W`' a wolf, '`D`' is a dog and '`.`' an empty space. You are not allowed to move, remove or add a sheep or a wolf.

If there are multiple solutions, you may print any of them. You don't have to minimize the number of dogs.

Examples

input
6 6 ..S... ..S.W. .S.... ..W... ...W.. ......
output
Yes ..SD.. ..SDW. .SD... .DW... DD.W.. ......

input
1 2 SW
output
No

input
5 5 .S... ...S. S.... ...S. .S...
output
Yes .S... ...S. S.D.. ...S.

```
.S...
```

Note

In the first example, we can split the pasture into two halves, one containing wolves and one containing sheep. Note that the sheep at (2,1) is safe, as wolves cannot move diagonally.

In the second example, there are no empty spots to put dogs that would guard the lone sheep.

In the third example, there are no wolves, so the task is very easy. We put a dog in the center to observe the peacefulness of the meadow, but the solution would be correct even without him.

B. Primal Sport

time limit per test: 1.5 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Alice and Bob begin their day with a quick game. They first choose a starting number $X_0 \geq 3$ and try to reach one million by the process described below.

Alice goes first and then they take alternating turns. In the i-th turn, the player whose turn it is selects a prime number smaller than the current number, and announces the smallest multiple of this prime number that is not smaller than the current number.

Formally, he or she selects a prime $p < X_{i-1}$ and then finds the minimum $X_i \geq X_{i-1}$ such that p divides X_i. Note that if the selected prime p already divides X_{i-1}, then the number does not change.

Eve has witnessed the state of the game after two turns. Given X_2, help her determine what is the smallest possible starting number X_0. Note that the players don't necessarily play optimally. You should consider all possible game evolutions.

Input

The input contains a single integer X_2 ($4 \leq X_2 \leq 10^6$). It is guaranteed that the integer X_2 is composite, that is, is not prime.

Output

Output a single integer — the minimum possible X_0.

Examples

input
14
output
6

input
20
output
15

input
8192
output
8191

Note

In the first test, the smallest possible starting number is $X_0 = 6$. One possible course of the game is as follows:

- Alice picks prime 5 and announces $X_1 = 10$
- Bob picks prime 7 and announces $X_2 = 14$.

In the second case, let $X_0 = 15$.

- Alice picks prime 2 and announces $X_1 = 16$
- Bob picks prime 5 and announces $X_2 = 20$.

C. Producing Snow

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Alice likes snow a lot! Unfortunately, this year's winter is already over, and she can't expect to have any more of it. Bob has thus bought her a gift — a large snow maker. He plans to make some amount of snow every day. On day i he will make a pile of snow of volume V_i and put it in her garden.

Each day, every pile will shrink a little due to melting. More precisely, when the temperature on a given day is T_i, each pile will reduce its volume by T_i. If this would reduce the volume of a pile to or below zero, it disappears forever. All snow piles are independent of each other.

Note that the pile made on day i already loses part of its volume on the same day. In an extreme case, this may mean that there are no piles left at the end of a particular day.

You are given the initial pile sizes and the temperature on each day. Determine the total volume of snow melted on each day.

Input
The first line contains a single integer N ($1 \le N \le 10^5$) — the number of days.

The second line contains N integers $V_1, V_2, \ldots, V_N$ ($0 \le V_i \le 10^9$), where V_i is the initial size of a snow pile made on the day i.

The third line contains N integers $T_1, T_2, \ldots, T_N$ ($0 \le T_i \le 10^9$), where T_i is the temperature on the day i.

Output
Output a single line with N integers, where the i-th integer represents the total volume of snow melted on day i.

Examples

input
3 10 10 5 5 7 2
output
5 12 4

input
5 30 25 20 15 10 9 10 12 4 13
output
9 20 35 11 25

Note
In the first sample, Bob first makes a snow pile of volume 10, which melts to the size of 5 on the same day. On the second day, he makes another pile of size 10. Since it is a bit warmer than the day before, the first pile disappears completely while the second pile shrinks to 3. At the end of the second day, he has only a single pile of size 3. On the third day he makes a smaller pile than usual, but as the temperature dropped too, both piles survive till the end of the day.

D. Perfect Security

time limit per test: 3.5 seconds
memory limit per test: 512 megabytes
input: standard input
output: standard output

Alice has a very important message M consisting of some non-negative integers that she wants to keep secret from Eve. Alice knows that the only theoretically secure cipher is one-time pad. Alice generates a random key K of the length equal to the message's length. Alice computes the bitwise xor of each element of the message and the key ($A_i := M_i \oplus K_i$, where $\oplus$ denotes the bitwise XOR operation) and stores this encrypted message A. Alice is smart. Be like Alice.

For example, Alice may have wanted to store a message $M = (0, 15, 9, 18)$. She generated a key $K = (16, 7, 6, 3)$. The encrypted message is thus $A = (16, 8, 15, 17)$.

Alice realised that she cannot store the key with the encrypted message. Alice sent her key K to Bob and deleted her own copy. Alice is smart. Really, be like Alice.

Bob realised that the encrypted message is only secure as long as the key is secret. Bob thus randomly permuted the key before storing it. Bob thinks that this way, even if Eve gets both the encrypted message and the key, she will not be able to read the message. Bob is not smart. Don't be like Bob.

In the above example, Bob may have, for instance, selected a permutation $(3, 4, 1, 2)$ and stored the permuted key $P = (6, 3, 16, 7)$.

One year has passed and Alice wants to decrypt her message. Only now Bob has realised that this is impossible. As he has permuted the key randomly, the message is lost forever. Did we mention that Bob isn't smart?

Bob wants to salvage at least some information from the message. Since he is not so smart, he asks for your help. You know the encrypted message A and the permuted key P. What is the lexicographically smallest message that could have resulted in the given encrypted text?

More precisely, for given A and P, find the lexicographically smallest message O, for which there exists a permutation π such that

$O_i \oplus \pi(P_i) = A_i$ for every i.

Note that the sequence S is lexicographically smaller than the sequence T, if there is an index i such that $S_i < T_i$ and for all $j < i$ the condition $S_j = T_j$ holds.

Input
The first line contains a single integer N ($1 \le N \le 300000$), the length of the message.

The second line contains N integers $A_1, A_2, \ldots, A_N$ ($0 \le A_i < 2^{30}$) representing the encrypted message.

The third line contains N integers $P_1, P_2, \ldots, P_N$ ($0 \le P_i < 2^{30}$) representing the permuted encryption key.

Output
Output a single line with N integers, the lexicographically smallest possible message O. Note that all its elements should be non-negative.

Examples

input
3 8 4 13 17 2 7
output
10 3 28

input
5 12 7 87 22 11 18 39 9 12 16
output
0 14 69 6 44

input
10 331415699 278745619 998190004 423175621 42983144 166555524 843586353 802130100 337889448 685310951 226011312 266003835 342809544 504667531 529814910 684873393 817026985 844010788 993949858 1031395667
output
128965467 243912600 4281110 112029883 223689619 76924724 429589 119397893 613490433 362863284

Note
In the first case, the solution is $(10, 3, 28)$, since $8 \oplus 2 = 10$, $4 \oplus 7 = 3$ and $13 \oplus 17 = 28$. Other possible permutations of key yield messages $(25, 6, 10)$, $(25, 3, 15)$, $(10, 21, 10)$, $(15, 21, 15)$ and $(15, 6, 28)$, which are all lexicographically larger than the solution.

E. Picking Strings

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Alice has a string consisting of characters 'A', 'B' and 'C'. Bob can use the following transitions on any substring of our string in any order any number of times:

- A $\longrightarrow$ BC
- B $\longrightarrow$ AC
- C $\longrightarrow$ AB
- AAA $\longrightarrow$ empty string

Note that a substring is one or more consecutive characters. For given queries, determine whether it is possible to obtain the target string from source.

Input
The first line contains a string S ($1 \le |S| \le 10^5$). The second line contains a string T ($1 \le |T| \le 10^5$), each of these strings consists only of uppercase English letters 'A', 'B' and 'C'.

The third line contains the number of queries Q ($1 \le Q \le 10^5$).

The following Q lines describe queries. The i-th of these lines contains four space separated integers a_i, b_i, c_i, d_i. These represent the i-th query: is it possible to create $T[c_i..d_i]$ from $S[a_i..b_i]$ by applying the above transitions finite amount of times?

Here, $U[x..y]$ is a substring of U that begins at index x (indexed from 1) and ends at index y. In particular, $U[1..|U|]$ is the whole string U.

It is guaranteed that $1 \le a \le b \le |S|$ and $1 \le c \le d \le |T|$.

Output

Print a string of Q characters, where the i-th character is '1' if the answer to the i-th query is positive, and '0' otherwise.

Example

input
AABCCBAAB ABCB 5 1 3 1 2 2 2 2 4 7 9 1 1 3 4 2 3 4 5 1 3
output
10011

Note

In the first query we can achieve the result, for instance, by using transitions $AA\mathbf{B} \rightarrow AAA\mathbf{C} \rightarrow \mathbf{AAA}AB \rightarrow AB$.

The third query asks for changing AAB to A — but in this case we are not able to get rid of the character 'B'.