Problem A: Add Add Add

- Time Limit: 2 sec

Problem Statement

You are given two sequences of positive integers of length N, $(A_1, A_2, \ldots, A_N)$ and $(B_1, B_2, \ldots, B_N)$.
For $k = 2, 3, \ldots, 2N$, compute the value of $\sum_{i+j \le k}(A_i + B_j)$, that is, the sum of $(A_i + B_j)$ for all indices (i, j) such that $i + j \le k$ and $1 \le i, j \le N$.

Input

The input is given in the following format:

$$
\begin{array}{l}
N \\
A_1\ A_2\ \ldots\ A_N \\
B_1\ B_2\ \ldots\ B_N
\end{array}
$$

- $1 \le N \le 200{,}000$
- $1 \le A_i, B_i \le 10^6$ $(1 \le i \le N)$
- All input values are integers.

Output

Output $2N - 1$ lines. On the i-th line ($1 \le i \le 2N - 1$), output the answer for the case where $k = i + 1$.

Sample Input 1

```
3
1 1 1
1 1 1
```

Sample Output 1

```
2
6
12
16
18
```

Sample Input 2

```
5
3 7 1 8 3
7 10 5 3 4
```

Sample Output 2

```
10
37
70
114
165
206
230
248
255
```

Sample Input 3

```
1
3
5
```

Sample Output 3

```
8
```

Problem B: Broken Parentheses

- Time Limit: 2 sec

Problem Statement

Let us define a **correct parenthesis sequence** as a string that satisfies any of the following conditions:

- It is an empty string.
- It is formed by concatenating `(`, A, `)` in this order where A is a **correct parenthesis sequence**.
- It is formed by concatenating A and B in this order where A and B are non-empty **correct parenthesis sequences**.

Given a string S of length N consisting of the characters `(` and `)`.

For each i where $0 \leq i \leq N$, define the string T_i as the string obtained by concatenating the suffix of S of length $N-i$ and the reversed string of the prefix of S of length i, in this order. That is, if we denote the i-th character of S as S_i, the string T_i is formed by arranging the characters $S_{i+1}, S_{i+2}, \ldots, S_N, S_i, \ldots, S_2, S_1$ in sequence.

For each T_i where $0 \leq i \leq N$, solve the following problem:

- Consider an operation where you replace one character in T_i with either `(` or `)`. Find the minimum number of such operations required to make T_i a **correct parenthesis sequence**.

Input

The input is given in the following format:

N
S

- $2 \leq N \leq 200{,}000$
- N is even.
- S is a string of length N consisting only of `(` and `)`.

Output

Output $N+1$ lines. On the $i+1$-th line, output the answer for T_i.

Sample Input 1	Sample Output 1
4 ()()	0 2 2 2 2

Sample Input 2	Sample Output 2
6)))(((	4 2 2 0 0 0 0

Sample Input 3	Sample Output 3
8)())())(	3 1 3 1 1 1 1 1 1

Problem C: Convenient Banknotes

- Time Limit: 2 sec

Problem Statement

In the Kingdom of JAG, only 1-yen banknotes have been issued so far. However, due to the increase in the circulation of banknotes, the kingdom has decided to renew its banknote system entirely. The new banknote system is represented by a sequence of positive integers $X = (X_1, X_2, \dots, X_k)$. This means that the new system uses k types of banknotes with denominations of $X_1, X_2, \dots, X_k$ yen. You can decide the number of banknote types k and their values $X_1, X_2, \dots, X_k$ under the following restrictions:

- k is a positive integer.
- $1 = X_1 < X_2 < \cdots < X_k$.
- X_{i+1} must be a multiple of X_i $(1 \le i \le k-1)$.

In the Kingdom of JAG, goods are often traded at prices of A, B, or C yen. Therefore, the **inconvenience** of the new banknote system is defined as:

(The minimum number of banknotes required to represent A yen) $+$ (The minimum number of banknotes required to represent B yen) $+$ (The minimum number of banknotes required to represent C yen).

Your task is to find the minimum possible value of this inconvenience.

Input

The input is given in the following format:

$$A \; B \; C$$

- $1 \le A < B < C \le 10^8$
- All input values are integers.

Output

Output a single line with the minimum possible value of the inconvenience for the new banknote system.

Sample Input 1	Sample Output 1
`6 11 15`	`6`

Sample Input 2	Sample Output 2
`99999959 99999971 99999989`	`11`

Problem D: Do Make Segment Tree

- Time Limit: 2 sec

Problem Statement

Given an integer sequence $B = (B_1, B_2, \ldots, B_{2^N-1})$ of length $2^N - 1$, define $f(B)$ as follows:

- $f(B)$ is the minimum number of operations required to make the following condition true:
  - **Operation**: Choose one integer i such that $1 \le i \le 2^N - 1$, and either increase B_i by 1 or decrease B_i by 1.
  - **Condition**: For all i where $1 \le i \le 2^{N-1} - 1$, the condition $B_i = B_{2i} + B_{2i+1}$ should be satisfied.

You are given a sequence $A = (A_1, A_2, \ldots, A_{2^N-1})$ of length $2^N - 1$.
Process Q queries. For each query i (where $1 \le i \le Q$):

- Given integers x_i and v_i, update A_{x_i} to v_i and then output $f(A)$.

Input

The input is given in the following format:

$$
\begin{array}{l}
N \\
A_1 \ A_2 \ \ldots \ A_{2^N-1} \\
Q \\
x_1 \ v_1 \\
x_2 \ v_2 \\
\vdots \\
x_Q \ v_Q
\end{array}
$$

- $2 \le N \le 18$
- $1 \le Q \le 100{,}000$
- $-10^9 \le A_i \le 10^9$
- $1 \le x_i \le 2^N - 1$
- $-10^9 \le v_i \le 10^9$
- All input values are integers.

Output

Output Q lines. On the i-th line, output the answer for the i-th query.

Sample Input	Sample Output
3 2 3 0 1 -5 2 1 5 3 1 5 3 6 -1 5 1 1 0	9 5 3 2 4

Problem E: Expression Sum

- Time Limit: 3 sec

Problem Statement

You are given a string S. Each character in S is one of **`0123456789+()?`**.

Let T be a string formed by replacing each **`?`** in S with one of **`0123456789+()`**. Define $\mathrm{eval}(T)$ as follows:

- If T is a **valid expression**, then it is the value obtained by evaluating T as an expression.
- If T is not a **valid expression**, then it is 0.

Compute the sum of $\mathrm{eval}(T)$ for all possible ways to replace each **`?`** in S with one of **`0123456789+()`**, and output the result modulo $998{,}244{,}353$.

A **valid expression** is defined by the following BNF:

```
<expression> ::= <expression> "+" <primary> | <primary>
<primary> ::= "(" <expression> ")" | <number>
<number> ::= <nonzero-digit> <number-sub> | <digit>
<number-sub> ::= <number-sub> <digit> | <digit>
<digit> ::= "0" | <nonzero-digit>
<nonzero-digit> ::= "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

Input

The input is given in the following format:

S

- $1 \leq |S| \leq 3{,}000$
- Each character of S is one of **`0123456789+()?`**.

Output

Output the answer.

Sample Input 1	Sample Output 1
`?1?`	`46306`

Sample Input 2	Sample Output 2
`20???0+2??`	`651059511`

Problem F: Flip Path on Rooted Tree

- Time Limit: 2 sec

Problem Statement

You are given a rooted tree with N vertices, with vertex 1 as the root. The parent of vertex i ($2 \le i \le N$) is vertex p_i. Each vertex has a value of either 0 or 1 written on it, and initially, vertex i ($1 \le i \le N$) has the value a_i written on it.

You need to handle Q queries. The i-th query ($1 \le i \le Q$) is as follows:

- If the value written on vertex x_i is 0, change it to 1; if it is 1, change it to 0. After that, output the answer to the following problem:
  - Find the minimum number of operations required to make all vertices have the value 0 by repeatedly performing the following operation:
    - Select a vertex. For every vertex on the path from vertex 1 to the selected vertex (inclusive), change the value to 1 if it is 0, and to 0 if it is 1.

    It can be proved that this can be achieved in a finite number of operations.

Input

The input is given in the following format:

$$
\begin{array}{l}
N \\
p_2\ p_3\ \ldots\ p_N \\
a_1\ a_2\ \ldots\ a_N \\
Q \\
x_1 \\
x_2 \\
\vdots \\
x_Q
\end{array}
$$

- $2 \le N \le 200{,}000$
- $1 \le Q \le 200{,}000$
- $1 \le p_i < i$ ($2 \le i \le N$)
- $0 \le a_i \le 1$ ($1 \le i \le N$)
- $1 \le x_i \le N$ ($1 \le i \le Q$)
- All input values are integers.

Output

Output Q lines. On the i-th line, output the answer to the i-th query.

Sample Input	Sample Output
4 1 1 3 0 1 1 0 3 2 1 4	2 1 1

Problem G: Give Me a Lot of Triangles

- Time Limit: 2 sec

Problem Statement

You have A_1 sticks of length 1, A_2 sticks of length 2, and A_3 sticks of length 3. You can perform the following operation any number of times:

- Choose 3 sticks such that they can form a triangle. Use these 3 sticks to make a triangle. Once used, these sticks cannot be used to form other triangles.

To "form a triangle", the lengths of the chosen sticks a, b, and c must satisfy the triangle inequality: $a + b > c$, $b + c > a$, and $c + a > b$.

Determine the maximum number of triangles that can be made.

Given T test cases, compute the answer for each.

Input

The input is given in the following format:

$$\begin{array}{l} T \\ \mathrm{case}_1 \\ \mathrm{case}_2 \\ \vdots \\ \mathrm{case}_T \end{array}$$

Here, case_i denotes the i-th test case.

Each test case is given in the following format:

$$A_1 \ A_2 \ A_3$$

- $1 \leq T \leq 10{,}000$
- $0 \leq A_i \leq 10^8$
- All input values are integers.

Output

Output T lines. On the i-th line, output the answer for the i-th test case.

Sample Input	Sample Output
4 3 1 2 4 1 1 0 0 0 31415926 535897 93238462	2 1 0 41730095

Problem H: Half Plane Painting

- Time Limit: 2 sec

Problem Statement

You have a 2D plane that is initially entirely white. You can perform the following operation any number of times:

- Choose a line and the half-plane bounded by this line. Then, perform one of the following actions:
  - Paint the half-plane (excluding the boundary) black.
  - Paint the half-plane and the boundary white.

You are given the polygon P with N vertices, which is not necessarily convex. The vertices of P are given in counterclockwise order as $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, and the i-th edge of P connects vertex (x_i, y_i) to vertex $(x_{(i \bmod N)+1}, y_{(i \bmod N)+1})$.
Determine whether it is possible to use the aforementioned operations to paint only the interior of polygon P black, leaving everything else white.

Input

The input is given in the following format:

$$
\begin{array}{l}
N \\
x_1 \ y_1 \\
x_2 \ y_2 \\
\vdots \\
x_N \ y_N
\end{array}
$$

- $3 \le N \le 4{,}000$
- $-10^7 \le x_i, y_i \le 10^7 \quad (1 \le i \le N)$
- $(x_i, y_i) \ne (x_j, y_j) \quad (i \ne j)$
- The vertices of polygon P are given in counterclockwise order.
- The edges of polygon P do not share any points other than the vertices.
- Each internal angle of polygon P is not 180 degrees.
- All input values are integers.

Output

If it is possible to achieve the desired state with the operations, output **`Yes`**; otherwise, output **`No`**.

Sample Input 1

```
4
10 -5
2 -5
-7 6
-7 -8
```

Sample Output 1

```
Yes
```

Sample Input 2

```
12
17 1
19 3
12 10
19 17
17 19
10 12
3 19
1 17
8 10
1 3
3 1
10 8
```

Sample Output 2

```
No
```

Problem I: I Love Square Number

- Time Limit: 2 sec

Problem Statement

Consider a graph with $\frac{N(N+1)}{2}$ vertices and $\frac{3N(N-1)}{2}$ edges, where N is an integer greater than or equals to 2.

- The set of vertices is $\{(i,j) \mid 1 \le i \le N, 1 \le j \le i\}$.
- There is an edge with weight $a_{i,j}$ between (i,j) and $(i+1,j)$ (for $1 \le i \le N-1$ and $1 \le j \le i$).
- There is an edge with weight $b_{i,j}$ between (i,j) and $(i+1,j+1)$ (for $1 \le i \le N-1$ and $1 \le j \le i$).
- There is an edge with weight $c_{i,j}$ between (i,j) and $(i,j+1)$ (for $2 \le i \le N$ and $1 \le j \le i-1$).

For a simple path in this graph, the weight of the path is defined as the **product** of the weights of the edges that the path traverses.

Determine the number of unordered pairs of distinct vertices $\{s,t\}$ such that any simple path from s to t has a weight that is a square number.

Input

The input is given in the following format:

$$
\begin{array}{l}
N \\
a_{1,1} \\
a_{2,1}\ \ a_{2,2} \\
\vdots \\
a_{N-1,1}\ \ \cdots\ \ a_{N-1,N-1} \\
b_{1,1} \\
b_{2,1}\ \ b_{2,2} \\
\vdots \\
b_{N-1,1}\ \ \cdots\ \ b_{N-1,N-1} \\
c_{2,1} \\
c_{3,1}\ \ c_{3,2} \\
\vdots \\
c_{N,1}\ \ \cdots\ \ c_{N,N-1}
\end{array}
$$

- $2 \le N \le 1{,}000$
- $1 \le a_{i,j}, b_{i,j}, c_{i,j} \le 10^6$
- All input values are integers.

Output

Output the answer.

Sample Input 1

```
2
1
2
2
```

Sample Output 1

```
1
```

Sample Input 2

```
3
1
2 3
4
5 6
7
8 9
```

Sample Output 2

```
0
```

Problem J: Just Believe in Binary Search

- Time Limit: 4 sec

Problem Statement

Alice, who was exploring ruins in search of treasure, arrived at a corridor where the entrances to N rooms were lined up in a row. Upon investigation, she found that the rooms were numbered uniquely from 1 to N, but the exact number of each room was unknown until she entered. She discovered that the treasure was hidden in room K.

Given her remaining stamina, it was difficult for Alice to check all the rooms. However, Alice had a secret strategy to overcome this situation: binary search. Alice had successfully applied binary search to various challenges before. With her last ounce of strength, she decided to use binary search to find room K.

Specifically, she followed these steps:

- Initialize variables l and r with $l = 0$ and $r = N + 1$.
- Repeat steps 1 to 3 as follows:
  1. If $l + 1 = r$, stop the operation as she has not found room K.
  2. Set $m = \lfloor \frac{l+r}{2} \rfloor$. Enter the room positioned m-th from the left, check its number, and let this number be x.
  3. If $x = K$, stop the operation as she has found room K. If $x < K$, update l to m. If $x > K$, update r to m.

There are $N!$ possible mappings between rooms and numbers. You need to determine the number of mappings for which Alice can successfully find room K using the above procedure, modulo $998{,}244{,}353$.

Given T test cases, compute the answer for each.

Input

The input is given in the following format:

$$
\begin{array}{l}
T \\
\mathrm{case}_1 \\
\mathrm{case}_2 \\
\vdots \\
\mathrm{case}_T
\end{array}
$$

Here, case_i denotes the i-th test case.

Each test case is given in the following format:

$$N \ \ K$$

- $1 \le T \le 100{,}000$
- $3 \le N \le 10^6$
- $1 \le K \le N$
- All input values are integers.

Output

Output T lines. On the i-th line, output the answer for the i-th test case.

Sample Input	Sample Output
5 3 1 4 2 5 4 10 5 1000000 314159	4 12 66 1192320 853363991

Problem K: K-th Nondivisor

- Time Limit: 6 sec

Problem Statement

Process Q queries. The i-th query is as follows:

- Given integers L_i, R_i, and K_i, find the K_i-th smallest positive integer x such that x does not divide any of the integers from L_i to R_i (inclusive).

Input

The input is given in the following format:

$$\begin{aligned}&Q\\&L_1\ \ R_1\ \ K_1\\&\vdots\\&L_Q\ \ R_Q\ \ K_Q\end{aligned}$$

- $1 \leq Q \leq 100{,}000$
- $1 \leq L_i \leq R_i \leq 200{,}000$
- $1 \leq K_i \leq 200{,}000$
- All input values are integers.

Output

Output Q lines. On the i-th line, output the answer for the i-th query.

Sample Input	Sample Output
3 10 11 5 12345 23456 789 1 200000 200000	8 23629 400000

Problem L: Linear Time Inversion Number

- Time Limit: 2 sec

Problem Statement

Given a permutation P of length N, Alice uses the inversion number as a measure of how close P is to the permutation $(1, 2, \ldots, N)$, while Bob uses the metric $\frac{1}{2}\sum_{i=1}^{N}|P_i - i|$.

Here, the inversion number is the number of pairs (i, j) such that $i < j$ and $P_i > P_j$.

Given a sequence $A = (A_1, A_2, \ldots, A_K)$ of length K, there are $(N-K)!$ permutations of length N that have A as their prefix.

Find the number of these permutations for which Alice's metric and Bob's metric are equal, and return the result modulo $998{,}244{,}353$.

Input

The input is given in the following format:

$$N \ K$$
$$A_1 \ A_2 \ \ldots \ A_K$$

- $1 \leq N \leq 200{,}000$
- $0 \leq K \leq N$
- $1 \leq A_i \leq N \quad (1 \leq i \leq K)$
- $A_i \neq A_j \quad (i \neq j)$
- All input values are integers.

Output

Output the answer.

Sample Input 1	Sample Output 1
5 3 2 3 5	1

Sample Input 2	Sample Output 2
10 10 3 1 4 5 9 2 6 8 7 10	0

Sample Input 3	Sample Output 3
6 0	132

Problem M: Max Sum of GCD

- Time Limit: 2 sec

Problem Statement

For a sequence of positive integers $X = (X_1, X_2, \ldots, X_M)$ where $M \geq 2$, let $f(X)$ be the answer to the following problem:

- Among the M positive integers $X_1, X_2, \ldots, X_M$, paint at least one and at most $M - 1$ of them red, and paint the rest blue. Let R be the greatest common divisor (GCD) of the integers painted red, and B be the GCD of the integers painted blue. Find the maximum possible value of $R + B$.

You are given a sequence of N positive integers $A = (A_1, A_2, \ldots, A_N)$. You will also be given Q queries. For each query, you will be given two integers l_i and r_i such that $1 \leq l_i < r_i \leq N$. For each query, let $X = (A_{l_i}, A_{l_i+1}, \ldots, A_{r_i})$, and find $f(X)$.

Input

The input is given in the following format:

$$
\begin{array}{l}
N \\
A_1 \ A_2 \ \ldots \ A_N \\
Q \\
l_1 \ r_1 \\
l_2 \ r_2 \\
\vdots \\
l_Q \ r_Q
\end{array}
$$

- $2 \leq N \leq 200{,}000$
- $1 \leq A_i \leq 10^{18}$ for all $1 \leq i \leq N$
- $1 \leq Q \leq 200{,}000$
- $1 \leq l_i < r_i \leq N$ for all $1 \leq i \leq Q$

Output

Print Q lines. On the i-th line ($1 \leq i \leq Q$), output the answer to the i-th query.

Sample Input	Sample Output
6 3 6 2 5 4 4 3 1 3 4 6 1 6	7 9 7

Problem N: Noncoprime Subsequences

- Time Limit: 2 sec

Problem Statement

Given a sequence $A = (A_1, A_2, \ldots, A_N)$, a **good subsequence** of A is defined as a subsequence, that is not necessarily contiguous, where adjacent elements in the subsequence are not coprime.

Find the maximum length L of a **good subsequence** of A. Also, determine the number of **good subsequences** of length L, modulo $998{,}244{,}353$.

Input

The input is given in the following format:

$$N$$
$$A_1 \ A_2 \ \ldots \ A_N$$

- $1 \le N \le 200{,}000$
- $1 \le A_i \le 10^6$
- All input values are integers.

Output

Output 2 lines. On the first line, output L. On the second line, output the number of **good subsequences** of length L of A, modulo $998{,}244{,}353$.

Sample Input 1

```
3
2 3 6
```

Sample Output 1

```
2
2
```

Sample Input 2

```
5
1 1 1 1 1
```

Sample Output 2

```
1
5
```

Sample Input 3

```
10
631932 735902 895728 78537 723857 330739 286918 329211 539679 238506
```

Sample Output 3

```
7
2
```