The 3rd Universal Cup

Stage 4: Hongō

July 13-14, 2024

This problem set should contain 17 problems on 28 numbered pages.

Problem A. Again Make UTPC

Input file: `standard input`
Output file: `standard output`
Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given a string S of length N. Each letter of S is either `U`, `T`, `P` or `C`.

You can do the following operation zero or more times:

- Choose a pair of integers (i, j) which meets $1 \leq i \leq j \leq N$. Sort from the i-th letter to the j-th letter of S in ascending alphabetical order.

Find if it is possible to satisfy the following condition, and calculate the minimum number of operations if possible.

- S includes `UTPC` as a consecutive substring.

You have T test cases to solve.

Input

The input is given from Standard input in the following format, where case_i represents the i-th test case:

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

Each case is given in the following format:

$$
\begin{array}{l}
N \\
S
\end{array}
$$

- T, N are integers.
- $1 \leq T \leq 2 \times 10^5$
- $1 \leq N \leq 2 \times 10^5$
- S is a string which consists of `U`, `T`, `P` and `C`, and whose length is N.
- For each input file, the sum of N over all test cases does not exceed 2×10^5.

Output

Print T lines. The i-th line should contain the answer for the i-th test case. In detail, print the minimum number of operations if it is possible to satisfy the condition. If it is not possible, print `-1`.

Example

standard input	standard output
3 10 UCUCTPUCUC 5 UTCUP 12 TUPCTTPCUTPC	2 -1 0

Note

For the first test case, it is possible to satisfy the condition by the following two operations. It is not possible within one operation.

- Choose $(i, j) = (1, 4)$. S becomes `CCUUTPUCUC`.
- Choose $(i, j) = (7, 10)$. S becomes `CCUUTPCCUU`.

For the second test case, there is no way to satisfy the condition.

For the third test case, an operation is not necessary to satisfy the condition.

Problem B. Black or White 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You are given integers N, M, and K. Both of N and M are greater than or equal to 2. You have to color K cells black and the remaining $NM - K$ cells white in a grid of N rows and M columns.

Here, the **loss** for the colored grid is defined as follows:

- The number of 2×2 subgrids that contain exactly 2 black cells and 2 white cells.

Please provide one way to color the grid such that the **loss** is minimized.

You will be given T test cases, and you need to provide a solution for each test case.

Input

The input is given from Standard input in the following format, where case_i represents the i-th test case:

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

Each test case is in the following format:

$$N\ M\ K$$

- All values in the input are integers.
- $1 \le T \le 10^5$
- $2 \le N, M \le 1500$
- $0 \le K \le NM$
- For each input file, the sum of NM over all test cases does not exceed 4×10^6.

Output

Output the answers to each test case in order, line-separated.

For each test case, output a string of length M consisting of `0` and `1` over N lines.

If the j-th character of the string output in line i is `0`, this indicates that the square which is i-th from the top and j-th from the left is painted white, and `1` indicates that the square which is i-th from the top and j-th from the left is painted black.

If there is more than one way to fill the squares with the minimum **loss**, output one of them. Note that do not output the minimum value of **loss**.

Example

standard input	standard output
2 2 2 2 2 3 0	10 01 000 000

Note

- For the first test case, the **loss** is 1, which is the minimum value.
- For the second test case, the **loss** is 0, which is the minimum value.

Problem C. Contour Multiplication

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

There is a sequence of length 2^N given by $A_0, A_1, \ldots, A_{2^N-1}$. Initially, $A_0 = A_1 = \cdots = A_{2^N-1} = 1$.

You will perform K operations. In the i-th operation, for each j (where $0 \le j < 2^N$) such that $\mathrm{popcount}(j \oplus C_i) = D_i$, replace A_j with $(A_j \times X_i) \bmod M$.

Determine the values of $A_0, A_1, \ldots, A_{2^N-1}$ after performing all the operations.

Input

The input is given from Standard Input in the following format:

$N\ M\ K$
$C_1\ D_1\ X_1$
$C_2\ D_2\ X_2$
$\vdots$
$C_M\ D_M\ X_M$

- All input values are integers.
- $1 \le N \le 18$
- $2 \le M \le 10^9$
- $1 \le K \le 5 \times 10^5$
- $0 \le C_i < 2^N$
- $1 \le D_i \le N$
- $2 \le X_i \le 10^9$

Output

Output the values of $A_0, A_1, \ldots, A_{2^N-1}$ in a single line, separated by spaces, after performing all the operations.

Examples

standard input	standard output
3 100 2 0 2 4 3 0 25	1 1 1 0 1 4 4 1
4 998244353 7 0 2 4 3 0 25 9 4 37 4 1 16 6 3 8 1 4 68 13 3 97	1552 8 1 9700 1 64 229696 1 8 4 388 8 64 8 68 1

Problem D. DRD String

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A string S is called **DRD String** if there exist certain non-empty strings D and R, and S is formed by connecting D, R, D in this order. Now we can use M sorts of alphabets. Find the number, modulo 998244353, of DRD strings with a length N.

Input

The input is given from Standard Input in the following format:

N M

- All input values are integers.
- $3 \le N \le 10^6$
- $1 \le M \le 10^6$

Output

Print the answer in a single line.

Examples

standard input	standard output
6 2	40
3017 7801	515391664

Note

If we can use `a` and `b` in example 1, `abbaab` and `aaaaaa` are DRD strings with a length 6, but `abbabb` and `aaabbb` are not.

Problem E. Equally Dividing

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

Given two integers N and M. You have to fill each cell of an $N \times M$ grid with an integer from 1 to NM, each appearing exactly once.

A valid filling is defined as a **fair filling** if it satisfies the following conditions:

- Every integer from 1 to NM is written exactly once in one of the cells.
- The sum of the M integers written in each row is the same for all rows.

Determine if such a **fair filling** exists, and if it does, provide one example of it.

You will be given T test cases, and you need to provide a solution for each test case.

Input

The input is given from Standard input in the following format, where case_i represents the i-th test case:

T
case_1
case_2
$\vdots$
case_T

Each test case is in the following format:

N M

- All values in the input are integers.
- $1 \leq T \leq 10^4$
- $1 \leq N, M$
- $1 \leq NM \leq 3 \times 10^5$
- For each input file, the sum of NM over all test cases does not exceed 5×10^5.

Output

Output the answers to each test case in order, line-separated.

For each test case, if a **fair filling** does not exist, output `No`.

Otherwise, output one example of **fair filling** in the following format:

`Yes`
$S_{1,1}$ $S_{1,2}$ $\ldots$ $S_{1,M}$
$S_{2,1}$ $S_{2,2}$ $\ldots$ $S_{2,M}$
$\vdots$
$S_{N,1}$ $S_{N,2}$ $\ldots$ $S_{N,M}$

Here, $S_{i,j}$ represents the integer which is written in the square which is i-th from the top and j-th from the left.

Example

standard input	standard output
2 2 2 10 1	Yes 1 4 2 3 No

Note

- For the first test case, the sum of the M integers written in each row is $5 = 1 + 4 = 2 + 3$.
- For the second test case, it can be proved that a **fair filling** does not exist.

Problem F. Flip or Not

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

There are N cards arranged in a row. Initially, the i-th card from the left is face-up if the i-th character of the string S is `1`, and face-down if it is `0`. You can perform the following operation up to 10^6 times:

- Move the rightmost card to the leftmost position. If the moved card is face-up, then flip all the cards at positions $A_1,\ A_2,\ \ldots,\ A_P$ from the left. Additionally, you can choose to either flip all the cards at positions $B_1,\ B_2,\ \ldots,\ B_Q$ from the left or do nothing.

As a result of these operations, you want the i-th card from the left to be face-up if the i-th character of the string T is `1`, and face-down if it is `0`. Determine if it is possible to satisfy the condition with 10^6 or fewer operations, and if it is possible, output a sequence of operations that achieves the condition with the minimum number of operations.

Input

The input is provided in the following format from standard input:

$$
\begin{array}{l}
N \\
S \\
T \\
P \\
A_1\ A_2\ \ldots\ A_P \\
Q \\
B_1\ B_2\ \ldots\ B_Q
\end{array}
$$

- All numerical inputs are integers.
- $1 \le N \le 5000$
- Each of S and T is a string of length N consisting of `0` and `1`.
- $S \neq T$
- $1 \le P, Q \le N$
- $1 \le A_1 < A_2 < \cdots < A_P \le N$
- $1 \le B_1 < B_2 < \cdots < B_Q \le N$

Output

If it is impossible to satisfy the conditions with 10^6 or fewer operations, output `-1`. If it is possible to satisfy the conditions, output one operation sequence that minimizes the number of operations in the following format:

$$
\begin{array}{l}
M \\
U
\end{array}
$$

Here, M is the number of operations, and U is a string of length M consisting only of `0` and `1` representing the operation sequence. If the i-th character of U is `1`, it means that in the i-th operation, all cards from $B_1,\ B_2,\ \ldots,\ B_Q$ from the left are flipped. If the character is `0`, no action is taken.

Examples

standard input	standard output
5 00001 00111 3 1 2 3 2 3 5	4 1001
4 0110 1000 2 1 2 4 1 2 3 4	-1

Note

For the first case, during the first operation, moving the rightmost card to the far left changes the card state to 10000 Since the moved card is facing up, we flip the cards at positions $A_1,\ A_2,\ A_3$ (1st, 2nd, 3rd) from the left, resulting in 01100. Next, if we choose to flip the front/back of the cards at positions $B_1,\ B_2$ (3rd, 5th) from the left, the state becomes 01001. Continuing to follow the output example, in the second operation, the state changes to 01000, in the third operation to 00100, and in the fourth operation to 00111. There is no method to achieve this in fewer operations, so the output example is the correct output.

For the second test case, there is no way to satisfy the condition within 10^6 operations.

Problem G. Graph Weighting

Input file:	`standard input`
Output file:	`standard output`
Time limit:	5 seconds
Memory limit:	1024 megabytes

There is a connected undirected graph with N vertices numbered $1, 2 \dots, N$ and M edges. The i-th edge connects the vertex u_i and the vertex v_i. The graph may contain multiple edges between the same pair of vertices, but it does not contain self-loops.

For each $W = 0, 1, \dots, K$, solve the following problem:

Determine if there exists a way to assign a weight $w_i \in \{0, 1, \dots, L\}$ to the i-th edge for each i with $1 \leq i \leq M$, such that the weight of any spanning tree of the graph is exactly W. The weight of a spanning tree is defined as the sum of the weights of all the edges included in the spanning tree. If such an assignment exists, find the minimum value of $(w_1)^2 + (w_2)^2 + \cdots + (w_M)^2$ over all such assignments.

Input

The input is given from Standard Input in the following format:

N M K L
u_1 v_1
$\vdots$
u_M v_M

- All values in the input are integers.
- $2 \leq N \leq 10^5$
- $N - 1 \leq M \leq 2 \times 10^5$
- $1 \leq L, K \leq 10^5$
- $1 \leq u_i, v_i \leq N$
- $u_i \neq v_i$
- A given undirected graph is connected.

Output

For each $W = 0, 1, \dots, K$, output the answer to the problem in this order, separated by spaces. Specifically, if no assignment meets the conditions, output `-1`. If an assignment exists, output the minimum value of $(w_1)^2 + (w_2)^2 + \cdots + (w_M)^2$ over all such assignments.

Examples

standard input	standard output
4 4 3 2 1 2 2 3 2 4 3 4	0 1 3 4
2 3 2 1 1 2 2 1 1 2	0 3 -1
6 7 9 2 1 2 2 3 2 4 4 5 4 6 1 4 3 4	0 1 2 5 6 7 10 13 22 25

Note

In example 1, for instance, when $W = 2$, if we set $(w1, w2, w3, w4) = (0, 1, 1, 1)$, the weight of any spanning tree of the graph will be 2.

In example 2, it is not possible to make the weight of any spanning tree of the graph equal to 2.

Problem H. Huge Segment Tree

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

An interval of integers is considerd *segment-tree-like* if the following condition holds:

- The interval that can be expressed as $[2^i j, 2^i(j+1))$ $(0 \le i \le K, 0 \le j < 2^{K-i})$ with some integers i, j.

For a pair of integer (l, r) that satisfy $0 \le l < r \le 2^K$, it can be proven that it is possible to express the interval $[l, r)$ as the union of segment-tree-like intervals. We denote the minimum numbers of intervals required as $f(l, r)$.

For $k = 1, 2, \ldots, 2K-2$, solve the following problem:

- Find the numbers of pairs of integers (l, r) $(0 \le l < r \le 2^K)$ such that $f(l, r) = k$, modulo 998244353.

Input

The input is given from Standard Input in the following format:

```
K
```

- K is an integer.
- $2 \le K \le 5 \times 10^5$

Output

Print the answer to the problem when $k = 1, 2, \ldots, 2K-2$ in this order.

Examples

standard input	standard output
3	15 14 6 1
5	63 110 132 114 70 30 8 1
10	2047 4975 10896 21772 38360 58724 77184 86312 81448 64324 42112 22576 9744 3304 848 155 18 1

Note

In the first example, when $k = 4$, $f(l, r) = k$ holds only when $l = 1, r = 7$, so 1 is the output.

Problem I. I Love Marathon Contest

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

A marathon event will be held at a pond. The pond is circular and marked with $1, 2, \ldots, 2N$ at equal clockwise intervals along its perimeter. There are $2N$ participants in the marathon event; N people wear red hats and the remaining N people wear white hats.

The marathon event is run as follows:

- For each of the $2N$ marks, exactly one participant takes that position.
- The participant at the position marked with 1 takes the baton and starts running clockwise.
- The i-th ($1 \le i \le 2N - 1$) runner continues running until he/she reaches the position of the first person wearing a hat of a different color than his/hers. After reaching that position, he/she passes the baton to that person and leaves the pond. The person who is passed the baton starts running clockwise.
- The $2N$-th runner finishes the marathon by running to the position marked with 1.

If the length of one lap around the pond is 1, the sum of the distances run by $2N$ participants is an integer, which is L.

There are $(2N)!$ possible arrangements of $2N$ participants. Find the sum of L for all of them modulo 998244353.

Input

The input is given in the following format from standard input:

```
N
```

- All input numbers are integers.
- $1 \le N \le 10^6$

Output

Print the answer on a single line.

Examples

standard input	standard output
1	2
2	40
3	1656
4	112896
5	11750400

Note

For the first example, there are 2 possible arrangements, and $L = 1$ in both case.

For the second example, if the colors of hats worn by the participants who takes the position marked with $1, 2, 3, 4$ are:

- red, red, white, white, then $L = 2$.
- red, white, red, white, then $L = 1$.
- red, white, white, red, then $L = 2$.
- white, red, red, white, then $L = 2$.
- white, red, white, red, then $L = 1$.
- white, white, red, red, then $L = 2$.

There are 4 possible arrangements of participants for each of them. The sum of L for all of 24 arrangements is $(2 + 1 + 2 + 2 + 1 + 2) \times 4 = 40$.

Problem J. Japanese Gift Money

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

There are N types of banknotes, and the i-th type of banknote is the A_i-yen note. You have 10^{100} banknotes of each type. Here, $A_1 < A_2 < \cdots < A_N$ holds, and for each i such that $1 \le i \le N-1$, A_{i+1} is a multiple of A_i.

You are going to select some of these banknotes and put them into an envelope.

A way to put banknotes into the envelope is called **a good way of putting x yen** if the following conditions are satisfied:

- The total amount of money in the envelope is x yen.
- It is impossible to select banknotes from the envelope such that their total amount is exactly $\frac{x}{2}$ yen.

Besides, x yen is called **a good amount of money** if there exists a good way of putting x yen.

Find the number of good amounts of money between L yen and R yen, inclusive.

Input

The input is provided in the following format from standard input:

```
N L R
A_1 A_2 ... A_N
```

- All inputs are integers.
- $1 \le N \le 60$
- $1 \le L \le R \le 10^{18}$
- $1 = A_1 < A_2 < \cdots < A_N \le 10^{18}$
- A_{i+1} is a multiple of A_i. $(1 \le i \le N-1)$

Output

Print the answer on a single line.

Examples

standard input	standard output
3 20 30 1 5 10	8
8 500007484602844543 985892611352151235 1 1971 151767 10927224 87417792 118975614912 263174060185344 43686893990767104	483957600323779237

Note

For instance, 30 yen is a good amount of money, because putting three 10-yen notes is a good way of putting 30 yen.

On the other hand, 20 yen is not a good amount of money, because there is no good way of putting 20 yen.

There are 8 good amounts of money between 20 yen and 30 yen: $21, 23, 25, 26, 27, 28, 29, 30$ yen.

Problem K. Kth Sum

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

You are given three sequences of integers $A = (A_1, A_2, \ldots, A_N)$, $B = (B_1, B_2, \ldots, B_N)$, and $C = (C_1, C_2, \ldots, C_N)$, each of length N.

Consider all possible sums of the form $A_i + B_j + C_k$ where $1 \leq i, j, k \leq N$. Your task is to find the K-th smallest sum among these N^3 sums.

Input

The first line contains two integers N and $K (1 \leq N \leq 50,000, 1 \leq K \leq \min(N^3, 10^9))$.

The second line contains N integers $A_1, A_2, \ldots, A_N (0 \leq A_i \leq 10^9)$.

The third line contains N integers $B_1, B_2, \ldots, B_N (0 \leq B_j \leq 10^9)$.

The fourth line contains N integers $C_1, C_2, \ldots, C_N (0 \leq C_k \leq 10^9)$.

Output

Print the K-th smallest sum among all possible sums of the form $A_i + B_j + C_k$.

Examples

standard input	standard output
2 4 1 2 3 4 5 6	10
10 40 11 9 13 12 15 11 11 2 11 17 3 1 10 2 12 18 9 11 11 15 14 9 4 14 16 9 20 2 1 18	14
1 1 1000000000 1000000000 1000000000	3000000000

Note

For the first test case, all possible sums are $9, 10, 10, 10, 11, 11, 11, 12$ in ascending order. Therefore, the 4-th smallest sum is 10.

Problem L. Largest Triangle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

There are N straight sticks. The length of i-th stick is L_i.

Consider making a (non-degenerate) triangle with 3 sticks selected in them. Determine whether the selection exists, and if it exists, calculate the square of the maximum of surface of the triangle which can be made.

There are T test cases, so answer each of them.

Input

The input is given from Standard Input in the following format. In this, case_i means the i-th test case.

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

Each test case is given in the following format.

$$\begin{array}{l} N \\ L_1\ L_2\ \ldots\ L_N \end{array}$$

- All inputs are integer.
- $1 \le T \le 2 \times 10^5$
- $3 \le N \le 3 \times 10^5$
- $2 \le L_i \le 20000$
- L_i is even.
- In each input, the sum of N of all test cases is equal or less than 2×10^5.

Output

Print T lines. In the i-th line, print the answer of i-th test case.

In each test case, print `-1` if there exists no valid selection of 3 sticks. If it exists, print the square of the maximum of area of the triangle which can be made. Note that this value is integer under the constraints.

Example

standard input	standard output
3 5 2 2 2 2 2 7 2 6 4 10 8 10 20 5 4 16 36 64 100	3 1344 -1

Note

In the 2nd test case, the triangle with 3 sides of length $8, 10, 10$ has a maximum area of $8\sqrt{21}$.The square of this is 1344.

In the 3rd test case, no three bars cannot be combined to form a non-degenerate triangle.

Problem M. Majority and Permutation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You are given an integer sequence $(A_1, A_2, \ldots, A_M)$ consisting of odd numbers between 1 and $2N-1$ (inclusive).

Find the number of permutations $P = (P_1, P_2, \ldots, P_{2N})$ of $(1, 2, \ldots, 2N)$ that satisfy the following condition, modulo 998244353.

- There exists a binary string S of length $2N$ consisting only of `0` and `1` that meets all of the following conditions:
  - The frequencies of `0` and `1` in S are exactly N each.
  - For each $i = 1, 2, \ldots, M$, in the $1, 2, \ldots, A_i$ -th characters of S, the most frequent character is `0`.
  - For each $i = 1, 2, \ldots, M$, in the $P_1, P_2, \ldots, P_{A_i}$ -th characters of S, the most frequent character is `0`.

Input

The input is provided in the following format from standard input:

N M
A_1 A_2 $\ldots$ A_M

- All inputs are integers.
- $1 \le M \le N \le 10^5$
- $1 \le A_1 < A_2 < \cdots < A_M \le 2N-1$
- A_i is odd.

Output

Print the answer on a single line.

Example

standard input	standard output
2 2 1 3	14

Note

For example, if $P = (2, 1, 3, 4)$, then $S =$ `0011` satisfies all three conditions.

On the other hand, if $P = (4, 3, 2, 1)$, there is no string of length 4 that satisfies all three conditions.

Problem N. Number of Abbreviations

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You have a string $S = S_1 S_2 \dots S_N$ of length N consisting of lowercase English letters. Determine the number of distinct strings that can be obtained by performing the following operation exactly once:

- Choose integers l and r such that $1 \le l \le r \le N$, and remove the substring from the l-th to the r-th character of S. The resulting string is $S_1 S_2 \dots S_{l-1} S_{r+1} \dots S_N$.

Input

The input is provided in the following format from standard input:

N
S

- N is an integer.
- $1 \le N \le 5 \times 10^5$
- S is a string of length N consisting of lowercase English letters.

Output

Print the answer on a single line.

Examples

standard input	standard output
5 abbab	11
5 aaaaa	5
4 utpc	10

Note

In the first example, the possible resulting strings are the following 11 types:

- Empty string
- `a`
- `aab`
- `ab`
- `abab`
- `abb`

- `abba`
- `abbb`
- `b`
- `bab`
- `bbab`

Problem O. Optimal Train Operation

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	1024 megabytes

The UTPC Railway has $N+1$ stations arranged along a single line, numbered consecutively from 0 to N from the starting station to the terminal station. For each i $(0 \le i \le N-1)$, station i and station $i+1$ are adjacent, and the congestion level between these stations is C_i. Currently, "rail yards" are located at stations 0 and N.

In the next timetable revision, rail yards will be constructed by repeating the following operation multiple times:

- Select a station i $(1 \le i \le N-1)$ and build a rail yard there. This operation costs A_i.

Next, trains will be operated between rail yards by repeating the following operation multiple times:

- Select stations l and r $(l < r)$ where rail yards are located and operate a train between these stations. This operation decreases the congestion level between stations i and $i+1$ $(l \le i < r)$ by 1. This operation costs $r - l$.

The goal of the timetable revision is to reduce the congestion level between stations i and $i+1$ $(0 \le i \le N-1)$ to 0 or below. Calculate the minimum total cost required for constructing rail yards and operating trains to achieve this goal.

Input

The input is given from Standard Input in the following format:

$$
\begin{array}{llll}
N & & & \\
C_0 & C_1 & \ldots & C_{N-1} \\
A_1 & A_1 & \ldots & A_{N-1}
\end{array}
$$

- All values in the input are integers.
- $2 \le N \le 5 \times 10^5$
- $1 \le C_i \le 10^9 \quad (0 \le i \le N-1)$
- $1 \le A_i \le 10^9 \quad (1 \le i \le N-1)$

Output

Output the answer.

Examples

standard input	standard output
4 3 1 4 1 5 9 2	15
9 28 35 19 27 84 98 78 79 60 40 35 54 63 72 71 27 94	682

Note

In the first example, a rail yard is established at station 3, and 3 trains are set between stations 0 and 3, and 1 train is set between stations 0 and 4. As a result, the congestion level of each section becomes 0 or less, and the total cost is 15. This is the minimum cost.

Problem P. Priority Queue 3

Input file: `standard input`
Output file: `standard output`
Time limit: 2 seconds
Memory limit: 1024 megabytes

You are given a string S of length $N + M$ consisting of N `+` characters and M `-` characters, and a set $A = \{A_1, A_2, \ldots, A_M\}$ consisting of M integers.

Prepare two sets $X = \{\}$ and $Y = \{\}$, and perform the following operations in order for $i = 1, 2, \ldots, N+M$:

- When the i-th character of S is `+`, select one integer from 1 to N that is not included in either X or Y, and add it to X.
- When the i-th character of S is `-`, remove the smallest integer m contained in X from X and add m to Y. From the constraints, it is guaranteed that X is not empty just before this operation.

There are $N!$ ways to determine the order of integers to be added to X. Among them, find the number of ways such that after performing all operations, $Y = A$. Print the answer modulo 998244353.

Input

The input is given in the following format from standard input:

N M
S
A_1 A_2 $\ldots$ A_M

- All input numbers are integers.
- $1 \le M \le N \le 300$
- S is a string of length $N + M$ consisting of N `+` characters and M `-` characters.
- For $i = 1, 2, \ldots, N + M$, the number of `-` characters appearing up to the i-th character does not exceed the number of `+` characters appearing up to the i-th character.
- $1 \le A_1 < A_2 < \cdots < A_M \le N$

Output

Print the answer on a single line.

Examples

standard input	standard output
4 2 ++-++- 1 3	4
6 4 ++-++---++ 2 3 4 6	48
20 10 ++++-++++++--+--+-+++++--+-++- 1 2 3 4 5 6 7 9 12 13	179396825

Note

For the first example, as an example of a sequence of operations that satisfies the conditions, the following can be considered:

- When $i = 1$, add 3 to X. Now $X = \{3\}$ and $Y = \{\}$.
- When $i = 2$, add 4 to X. Now $X = \{3, 4\}$ and $Y = \{\}$.
- When $i = 3$, remove the smallest integer 3 from X and add it to Y. Now $X = \{4\}$ and $Y = \{3\}$.
- When $i = 4$, add 2 to X. Now $X = \{2, 4\}$ and $Y = \{3\}$.
- When $i = 5$, add 1 to X. Now $X = \{1, 2, 4\}$ and $Y = \{3\}$.
- When $i = 6$, remove the smallest integer 1 from X and add it to Y. Now $X = \{2, 4\}$ and $Y = \{1, 3\}$.

For the second example, The end of S is not necessarily `-`.

Problem Q. Quotient Sum

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	1024 megabytes

You are given a sequence $A = (A_1, A_2, \ldots, A_N)$ consisting of N distinct positive integers. Consider rearranging the elements of A to obtain a sequence $B = (B_1, B_2, \ldots, B_N)$. Find the minimum value of the following expression:

$$\left\lfloor \frac{B_2}{B_1} \right\rfloor + \left\lfloor \frac{B_3}{B_2} \right\rfloor + \cdots + \left\lfloor \frac{B_N}{B_{N-1}} \right\rfloor + \left\lfloor \frac{B_1}{B_N} \right\rfloor .$$

Here, $\lfloor x \rfloor$ denotes the largest integer less than or equal to the real number x.

Input

The input is provided in the following format from standard input:

N
A_1 A_2 $\ldots$ A_N

- All inputs are integers.
- $2 \leq N \leq 2 \times 10^5$
- $1 \leq A_i \leq 10^{18}$
- $A_i \neq A_j$ $(i \neq j)$

Output

Print the answer on a single line.

Examples

standard input	standard output
3 2 3 6	3
2 15 4	3
9 284791808 107902 13660981249408 4622332661 13405199 24590921 361 244448137 16077087227955422	4580

Note

In the first example, if we set $(B_1, B_2, B_3) = (6, 2, 3)$, we have

$$\left\lfloor \frac{B_2}{B_1} \right\rfloor + \left\lfloor \frac{B_3}{B_2} \right\rfloor + \left\lfloor \frac{B_1}{B_3} \right\rfloor = \left\lfloor \frac{2}{6} \right\rfloor + \left\lfloor \frac{3}{2} \right\rfloor + \left\lfloor \frac{6}{3} \right\rfloor = 0 + 1 + 2 = 3.$$