A - Operations on a Stack

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 400 points

Problem Statement

You are given an integer sequence of length N: $(A_1, A_2, \ldots, A_N)$. There is also a sequence S, which is initially empty.

For each $i = 1, 2, \ldots, N$ in this order, you perform exactly one of the following two operations:

- Append A_i as an element to the end of S.
- Delete the last element of S. You cannot choose this operation if S is empty.

Print the maximum possible value of the sum of the elements of S after all operations.

Constraints

- $1 \leq N \leq 2 \times 10^5$
- $-10^9 \leq A_i \leq 10^9$
- All input values are integers.

Input

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

$$
\begin{array}{l}
N \\
A_1 \quad A_2 \quad \ldots \quad A_N
\end{array}
$$

Output

Print the answer.

Sample Input 1

```
6
3 -1 -4 5 -9 2
```

Sample Output 1

```
8
```

Starting from the initial state where S is an empty sequence, consider the following operations:

- For $i = 1$, append $A_1 = 3$ to the end of S. Now, $S = (3)$.
- For $i = 2$, append $A_2 = -1$ to the end of S. Now, $S = (3, -1)$.
- For $i = 3$, delete the last element of S. Now, $S = (3)$.
- For $i = 4$, append $A_4 = 5$ to the end of S. Now, $S = (3, 5)$.
- For $i = 5$, append $A_5 = -9$ to the end of S. Now, $S = (3, 5, -9)$.
- For $i = 6$, delete the last element of S. Now, $S = (3, 5)$.

Here, the sum of the elements of S after all operations is $3 + 5 = 8$, which is the maximum possible value.

Sample Input 2

```
1
-1
```

Sample Output 2

```
-1
```

Note that if S is empty, you must choose to append an element.

Sample Input 3

```
20
-14 74 -48 38 -51 43 5 37 -39 -29 80 -44 -55 59 17 89 -37 -68 38 -16
```

Sample Output 3

```
369
```

B - Minimum Cost Sort

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 600 points

Problem Statement

You are given a permutation $P = (P_1, P_2, \ldots, P_N)$ of $(1, 2, \ldots, N)$. Takahashi can repeatedly perform the following operation on P (possibly zero times):

- Choose an integer i satisfying $1 \leq i \leq N-1$. Pay a cost of i, and swap P_i and P_{i+1}.

Find the minimum total cost required to sort P in ascending order.

Constraints

- $2 \leq N \leq 2 \times 10^5$
- $(P_1, P_2, \ldots, P_N)$ is a permutation of $(1, 2, \ldots, N)$.
- All input values are integers.

Input

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

$$
\begin{array}{l}
N \\
P_1 \quad P_2 \quad \ldots \quad P_N
\end{array}
$$

Output

Print the minimum total cost required to sort P in ascending order.

Sample Input 1

```
3
3 2 1
```

Sample Output 1

```
4
```

Takahashi can sort P in ascending order as follows:

- Pay a cost of 1 and swap $P_1 = 3$ and $P_2 = 2$. Now, $P = (2, 3, 1)$.
- Pay a cost of 2 and swap $P_2 = 3$ and $P_3 = 1$. Now, $P = (2, 1, 3)$.
- Pay a cost of 1 and swap $P_1 = 2$ and $P_2 = 1$. Now, $P = (1, 2, 3)$.

The total cost for these operations is 4, which is the minimum possible.

Sample Input 2

```
5
2 4 1 3 5
```

Sample Output 2

```
6
```

Sample Input 3

```
2
1 2
```

Sample Output 3

```
0
```

C - Cost to Flip

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 600 points

Problem Statement

You are given two integer sequences of length N, $A = (A_1, A_2, \ldots, A_N)$ and $B = (B_1, B_2, \ldots, B_N)$, each consisting of 0 and 1.

You can perform the following operation on A any number of times (possibly zero):

1. First, choose an integer i satisfying $1 \leq i \leq N$, and flip the value of A_i (if the original value is 0, change it to 1; if it is 1, change it to 0).
2. Then, pay $\sum_{k=1}^{N} A_k C_k$ yen as the cost of this operation.

Note that the cost calculation in step 2 uses the A after the change in step 1.

Print the minimum total cost required to make A identical to B.

Constraints

- $1 \leq N \leq 2 \times 10^5$
- $A_i, B_i \in 0, 1$
- $1 \leq C_i \leq 10^6$
- All input values are integers.

Input

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

$$
\begin{array}{llll}
N \\
A_1 & A_2 & \ldots & A_N \\
B_1 & B_2 & \ldots & B_N \\
C_1 & C_2 & \ldots & C_N
\end{array}
$$

Output

Print the answer.

Sample Input 1

```
4
0 1 1 1
1 0 1 0
4 6 2 9
```

Sample Output 1

```
16
```

Consider the following procedure:

- First, flip A_4. Now, $A = (0,1,1,0)$. The cost of this operation is $0 \times 4 + 1 \times 6 + 1 \times 2 + 0 \times 9 = 8$ yen.
- Next, flip A_2. Now, $A = (0,0,1,0)$. The cost of this operation is $0 \times 4 + 0 \times 6 + 1 \times 2 + 0 \times 9 = 2$ yen.
- Finally, flip A_1. Now, $A = (1,0,1,0)$, which matches B. The cost of this operation is $1 \times 4 + 0 \times 6 + 1 \times 2 + 0 \times 9 = 6$ yen.

In this case, the total cost is $8 + 2 + 6 = 16$ yen, which is the minimum possible.

Sample Input 2

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

Sample Output 2

```
0
```

A and B are already identical initially, so there is no need to perform any operations.

Sample Input 3

```
20
1 1 1 1 0 0 1 1 0 0 0 1 0 1 0 1 1 0 1 0
0 0 0 1 1 1 0 1 1 0 0 0 0 0 0 1 0 1 0 0
52 73 97 72 54 15 79 67 13 55 65 22 36 90 84 46 1 2 27 8
```

Sample Output 3

```
2867
```

D - Reverse Brackets

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 700 points

Problem Statement

A string is defined to be a **valid parenthesis sequence** if and only if it satisfies one of the following conditions:

- It is an empty string.
- There exists a valid parenthesis sequence A such that the string is obtained by concatenating `(`, A, and `)` in this order.
- There exist non-empty valid parenthesis sequences A and B such that the string is obtained by concatenating A and B in this order.

You are given a valid parenthesis sequence S of length N. You can perform the following operation any number of times:

- Choose a contiguous substring of S that is a valid parenthesis sequence, and reverse it.

Here, reversing the substring of S from the l-th character to the r-th character means the following:

- For every integer i satisfying $l \leq i \leq r$, simultaneously replace S_i with `)` if S_{l+r-i} is `(`, and with `(` if S_{l+r-i} is `)`.(Note that reversing here is different from the usual definition of reversing.)

Find the number, modulo 998244353, of distinct strings S that you can have at the end of the process.

Constraints

- $1 \leq N \leq 5000$
- $|S| = N$
- S is a valid parenthesis sequence.

Input

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

```
N
S
```

Output

Print the answer.

Sample Input 1

```
6
(())()
```

Sample Output 1

```
2
```

For example, you can transform S into `()(())` by doing the following:

- Choose the substring from the 1st to the 6th character of S. This is a valid parenthesis sequence. S becomes `()(())`.

The only other string that can be formed is `(())()`. Thus, the answer is 2.

Sample Input 2

```
2
()
```

Sample Output 2

```
1
```

E - Swap 0^X and 1^Y

Time Limit: 2 sec / Memory Limit: 1024 MiB

Score : 900 points

Problem Statement

You are given two strings S and T, each of length N and consisting of `0` and `1`, as well as two positive integers X and Y. For $i = 1, 2, \ldots, N$, let S_i denote the i-th character of S.

Determine whether it is possible to make S identical to T by repeatedly performing Operations A and B below any number of times (possibly zero) in any order:

- (Operation A) Choose an integer i satisfying $1 \leq i \leq N-(X+Y)+1$, $S_i = S_{i+1} = \cdots = S_{i+X-1} =$ `0`, and $S_{i+X} = S_{i+X+1} = \cdots = S_{i+X+Y-1} =$ `1`, then change each of $S_i, S_{i+1}, \ldots, S_{i+Y-1}$ to `1` and each of $S_{i+Y}, S_{i+Y+1}, \ldots, S_{i+Y+X-1}$ to `0`.
- (Operation B) Choose an integer i satisfying $1 \leq i \leq N-(X+Y)+1$, $S_i = S_{i+1} = \cdots = S_{i+Y-1} =$ `1`, and $S_{i+Y} = S_{i+Y+1} = \cdots = S_{i+Y+X-1} =$ `0`, then change each of $S_i, S_{i+1}, \ldots, S_{i+X-1}$ to `0` and each of $S_{i+X}, S_{i+X+1}, \ldots, S_{i+X+Y-1}$ to `1`.

Constraints

- $1 \leq N \leq 5 \times 10^5$
- $1 \leq X, Y \leq N$
- S and T are strings of length N consisting of `0` and `1`.
- All input values are integers.

Input

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

```
N X Y
S
T
```

Output

If it is possible to make S identical to T, print `Yes`; otherwise, print `No`.

Sample Input 1

```
9 2 1
000111001
011000011
```

Sample Output 1

```
Yes
```

The following procedure can transform S into T:

- First, perform Operation A with $i = 2$. Now, $S =$ `010011001`.
- Next, perform Operation B with $i = 6$. Now, $S =$ `010010011`.
- Finally, perform Operation A with $i = 3$. Now, $S =$ `011000011`.

Thus, print `Yes`.

Sample Input 2

```
1 1 1
0
1
```

Sample Output 2

```
No
```

It is impossible to make S identical to T. Thus, print `No`.