Codeforces Round #508 (Div. 2)

A. Equality

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

You are given a string s of length n, which consists only of the first k letters of the Latin alphabet. All letters in string s are uppercase.

A *subsequence* of string s is a string that can be derived from s by deleting some of its symbols without changing the order of the remaining symbols. For example, "ADE" and "BD" are subsequences of "ABCDE", but "DEA" is not.

A subsequence of s called *good* if the number of occurences of each of the first k letters of the alphabet is the same.

Find the length of the longest good subsequence of s.

Input
The first line of the input contains integers n ($1 \le n \le 10^5$) and k ($1 \le k \le 26$).

The second line of the input contains the string s of length n. String s only contains uppercase letters from 'A' to the k-th letter of Latin alphabet.

Output
Print the only integer — the length of the longest good subsequence of string s.

Examples

input
9 3 ACAABCCAB
output
6

input
9 4 ABCABCABC
output
0

Note
In the first example, "ACBCAB" ("**AC**AA**BC**C**AB**") is one of the subsequences that has the same frequency of 'A', 'B' and 'C'. Subsequence "CAB" also has the same frequency of these letters, but doesn't have the maximum possible length.

In the second example, none of the subsequences can have 'D', hence the answer is 0.

B. Non-Coprime Partition

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

Find out if it is possible to partition the first n positive integers into two **non-empty** disjoint sets S_1 and S_2 such that:

$$\gcd(\mathrm{sum}(S_1), \mathrm{sum}(S_2)) > 1$$

Here $\mathrm{sum}(S)$ denotes the sum of all elements present in set S and $\gcd$ means thegreatest common divisor.

Every integer number from 1 to n should be present in **exactly one** of S_1 or S_2.

Input
The only line of the input contains a single integer n ($1 \le n \le 45\,000$)

Output
If such partition doesn't exist, print "No" (quotes for clarity).

Otherwise, print "Yes" (quotes for clarity), followed by two lines, describing S_1 and S_2 respectively.

Each set description starts with the set size, followed by the elements of the set in any order. Each set must be non-empty.

If there are multiple possible partitions — print any of them.

Examples

input
1
output
No

input
3
output
Yes 1 2 2 1 3

Note
In the first example, there is no way to partition a single number into two non-empty sets, hence the answer is "No".

In the second example, the sums of the sets are 2 and 4 respectively. The $\gcd(2, 4) = 2 > 1$, hence that is one of the possible answers.

C. Gambling

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

Two players A and B have a list of n integers each. They both want to maximize the subtraction between their score and their opponent's score.

In one turn, a player can either add to his score any element from his list (assuming his list is not empty), the element is removed from the list afterward. Or remove an element from his opponent's list (assuming his opponent's list is not empty).

Note, that in case there are equal elements in the list only one of them will be affected in the operations above. For example, if there are elements $\{1, 2, 2, 3\}$ in a list and you decided to choose 2 for the next turn, only a single instance of 2 will be deleted (and added to the score, if necessary).

The player A starts the game and the game stops when both lists are empty. Find the difference between A's score and B's score at the end of the game, if both of the players are playing optimally.

Optimal play between two players means that both players choose the best possible strategy to achieve the best possible outcome for themselves. In this problem, it means that each player, each time makes a move, which maximizes the final difference between his score and his opponent's score, knowing that the opponent is doing the same.

Input
The first line of input contains an integer n ($1 \le n \le 100\,000$) — the sizes of the list.

The second line contains n integers a_i ($1 \le a_i \le 10^6$), describing the list of the player A, who starts the game.

The third line contains n integers b_i ($1 \le b_i \le 10^6$), describing the list of the player B.

Output
Output the difference between A's score and B's score ($A - B$) if both of them are playing optimally.

Examples

input
2 1 4 5 1
output
0

input
3 100 100 100 100 100 100
output
0

input
2 2 1 5 6
output
-3

Note
In the first example, the game could have gone as follows:

- A removes 5 from B's list.
- B removes 4 from A's list.
- A takes his 1.
- B takes his 1.

Hence, A's score is 1, B's score is 1 and difference is 0.

There is also another optimal way of playing:

- A removes 5 from B's list.
- B removes 4 from A's list.
- A removes 1 from B's list.
- B removes 1 from A's list.

The difference in the scores is still 0.

In the second example, irrespective of the moves the players make, they will end up with the same number of numbers added to their score, so the difference will be 0.

D. Slime

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

There are n slimes in a row. Each slime has an integer value (possibly negative or zero) associated with it.

Any slime can eat its adjacent slime (the closest slime to its left or to its right, assuming that this slime exists).

When a slime with a value x eats a slime with a value y, the eaten slime disappears, and the value of the remaining slime changes to $x - y$.

The slimes will eat each other until there is only one slime left.

Find the maximum possible value of the last slime.

Input
The first line of the input contains an integer n ($1 \le n \le 500\,000$) denoting the number of slimes.

The next line contains n integers a_i ($-10^9 \le a_i \le 10^9$), where a_i is the value of i-th slime.

Output
Print an only integer — the maximum possible value of the last slime.

Examples

input
4 2 1 2 1
output
4

input
5 0 -1 -1 -1 -1
output
4

Note
In the first example, a possible way of getting the last slime with value 4 is:

- Second slime eats the third slime, the row now contains slimes $2, -1, 1$
- Second slime eats the third slime, the row now contains slimes $2, -2$
- First slime eats the second slime, the row now contains 4

In the second example, the first slime can keep eating slimes to its right to end up with a value of 4.

E. Maximum Matching

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

You are given n blocks, each of them is of the form [color$_1$|value|color$_2$], where the block can also be flipped to get [color$_2$|value|color$_1$].

A sequence of blocks is called *valid* if the touching endpoints of neighboring blocks have the same color. For example, the sequence of three blocks A, B and C is valid if the left color of the B is the same as the right color of the A and the right color of the B is the

same as the left color of C.

The value of the sequence is defined as the sum of the values of the blocks in this sequence.

Find the maximum possible value of the valid sequence that can be constructed from the subset of the given blocks. The blocks from the subset can be reordered and flipped if necessary. Each block can be used at most once in the sequence.

Input
The first line of input contains a single integer n ($1 \le n \le 100$) — the number of given blocks.

Each of the following n lines describes corresponding block and consists of $\text{color}_{1,i}$, value_i and $\text{color}_{2,i}$ ($1 \le \text{color}_{1,i}, \text{color}_{2,i} \le 4$, $1 \le \text{value}_i \le 100\,000$).

Output
Print exactly one integer — the maximum total value of the subset of blocks, which makes a valid sequence.

Examples

input
6 2 1 4 1 2 4 3 4 4 2 8 3 3 16 3 1 32 2

output
63

input
7 1 100000 1 1 100000 2 1 100000 2 4 50000 3 3 50000 4 4 50000 4 3 50000 3

output
300000

input
4 1 1000 1 2 500 2 3 250 3 4 125 4

output
1000

Note
In the first example, it is possible to form a valid sequence from all blocks.

One of the valid sequences is the following:

`[4|2|1] [1|32|2] [2|8|3] [3|16|3] [3|4|4] [4|1|2]`

The first block from the input (`[2|1|4]` $\rightarrow$ `[4|1|2]`) and second (`[1|2|4]` $\rightarrow$ `[4|2|1]`) are flipped.

In the second example, the optimal answers can be formed from the first three blocks as in the following (the second or the third block from the input is flipped):

`[2|100000|1] [1|100000|1] [1|100000|2]`

In the third example, it is not possible to form a valid sequence of two or more blocks, so the answer is a sequence consisting only of the first block since it is the block with the largest value.

F. Wrap Around

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

You are given a binary string s.

Find the number of distinct *cyclical* binary strings of length n which contain s as a substring.

The cyclical string t contains s as a substring if there is some cyclical shift of string t, such that s is a substring of this cyclical shift of t.

For example, the cyclical string "000111" contains substrings "001", "01110" and "10", but doesn't contain "0110" and "10110".

Two cyclical strings are called different if they differ from each other as strings. For example, two different strings, which differ from each other by a cyclical shift, are still considered **different** cyclical strings.

Input

The first line contains a single integer n ($1 \le n \le 40$) — the length of the target string t.

The next line contains the string s ($1 \le |s| \le n$) — the string which must be a substring of cyclical string t. String s contains only characters '0' and '1'.

Output

Print the only integer — the number of distinct cyclical binary strings t, which contain s as a substring.

Examples

input
2 0
output
3

input
4 1010
output
2

input
20 10101010101010
output
962

Note

In the first example, there are three cyclical strings, which contain "0" — "00", "01" and "10".

In the second example, there are only two such strings — "1010", "0101".