Problem A. Coloring

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

You are given n elements numbered from 1 to n. Element i has value w_i and color c_i. Each element also has a pointer a_i to some other element.

Initially, the color of element s is 1, while the color of all the other elements is 0. More formally, $c_s = 1$ and $c_i = 0$ for all $i \neq s$ $(1 \leq i \leq n)$.

You can perform the following operation for any number of times:

- Assign $c_i \leftarrow c_{a_i}$ at a cost of p_i.

Your score is equal to the sum of values of all the elements with color 1 after the operations minus the sum of costs of the operations.

Find the maximum possible score you can obtain.

Input

The first line contains two integers n, s $(1 \leq s \leq n \leq 5 \times 10^3)$ — the number of elements and the element with color 1 initially.

The second line contains n integers $w_1, w_2, \ldots, w_n$ $(-10^9 \leq w_i \leq 10^9)$ — the value of the elements.

The third line contains n integers $p_1, p_2, \ldots, p_n$ $(0 \leq p_i \leq 10^9)$ — the cost of changing the color of each element.

The fourth line contains n integers $a_1, a_2, \ldots, a_n$ $(1 \leq a_i \leq n,\ a_i \neq i)$.

Output

Output one integer representing the answer in one line.

Examples

standard input	standard output
3 1 -1 -1 2 1 0 0 3 1 2	1
10 8 36175808 53666444 14885614 -14507677 -92588511 52375931 -87106420 -7180697 -158326918 98234152 17550389 45695943 55459378 18577244 93218347 64719200 84319188 34410268 20911746 49221094 8 1 2 2 8 8 4 7 8 4 (There won't be extra line breakers in the actual test cases.)	35343360

Note

In the first sample, you can successively perform the following operations:

1. Assign $c_2 \leftarrow c_{a_2}$ at a cost of p_2, then $c = [1, 1, 0]$;
2. Assign $c_1 \leftarrow c_{a_1}$ at a cost of p_1, then $c = [0, 1, 0]$;
3. Assign $c_3 \leftarrow c_{a_3}$ at a cost of p_3, then $c = [0, 1, 1]$;
4. Assign $c_2 \leftarrow c_{a_2}$ at a cost of p_2, then $c = [0, 0, 1]$.

After the operations, only the color of element 3 is 1, so your score is equal to $w_3 - (p_2 + p_1 + p_3 + p_2) = 1$. It can be shown that it is impossible to obtain a score greater than 1.

Problem B. Binary String

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

You are given a binary string $a_0a_1a_2\ldots a_{n-1}$ arranged on a cycle. Each second, you will change every 01 to 10 simultaneously. In other words, if $a_i = 0$ and $a_{(i+1) \bmod n} = 1$, you swap a_i and $a_{(i+1) \bmod n}$. For example, we will change `100101110` to `001010111`.

You need to answer how many different strings will occur in infinite seconds, modulo 998244353.

Note: Two strings $a_0a_1\ldots a_{n-1}$ and $b_0b_1\ldots b_{n-1}$ are different if there exists an integer $i \in \{0, 1, \ldots, n-1\}$ such that $a_i \neq b_i$. Thus, the cyclic shifts of a string may be different from the original string.

Input

The first line contains an integer T ($1 \leq T \leq 10^6$) – the number of test cases.

For each test case, the first line contains a binary string $a_0a_1\ldots a_{n-1}$ ($a_i \in \{0, 1\}$).

It is guaranteed that the sum of lengths of strings over all test cases does not exceed 10^7.

Output

For each test case, output one integer representing the answer in one line.

Example

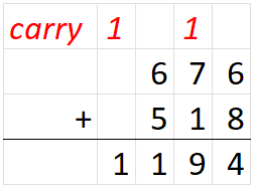
standard input	standard output
3 1 001001 0001111	1 3 9

Problem C. Best Carry Player 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	4 seconds
Memory limit:	1024 megabytes

Given a positive integer x, find the minimum positive integer y such that the number of **carries**[1] of $x+y$ is exactly k.

We adds numbers **by column addition in base-ten**, just like what we normally do in primary school. For example, there are two carries in the following addition.



Input

The first line contains an integer T ($1 \le T \le 10^5$) – the number of test cases.

For each test case, the first line contains two integers x, k ($1 \le x < 10^{18}, 0 \le k \le 18$).

Output

For each test case, output one integer representing the answer in one line. If there is no solution, output -1 instead.

Example

standard input	standard output
4 12345678 0 12345678 5 12345678 18 990099 5	1 54322 999999999987654322 9910

[1] which means "进位" in Chinese

Problem D. Minimum Suffix

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

For a string s of length n, we define $p_j = x$ if $s[x \ldots j]$ is the minimum suffix of $s[1 \ldots j]$, for all $j = 1, \ldots, n$. (A suffix is the minimum suffix of a string if it is lexicographically smaller than any other suffix of that string.)

You are to recover s from $p_1, \ldots, p_n$. If there are multiple answers, find the lexicographically smallest one.

Input

The first line contains a single integer T ($1 \le T \le 10^5$) representing the number of test cases.

For each test case, the first line contains a single integer n ($1 \le n \le 3 \times 10^6$) representing the length of s. The next line contains n integers $p_1, \ldots, p_n$ ($1 \le p_i \le i$ for all $1 \le i \le n$).

It is guaranteed that the sum of n over all test cases does not exceed 3×10^6.

Output

For each test case, output one line. If there is no solution, output -1. Otherwise, output the lexicographically smallest s. Characters of s are represented by positive integers. Smaller integers represent smaller characters in the lexicographical order.

Example

standard input	standard output
6 3 1 1 1 3 1 1 2 3 1 1 3 3 1 2 1 3 1 2 2 3 1 2 3	1 2 2 -1 1 2 1 1 1 2 2 1 2 1 1 1

Note

As the input/output can be huge, it is recommended to use fast input/output methods.

Problem E. Map

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

It is a famous math fact that if you drop a map of a park completely inside the park, then there exists a point on the map which overlays with the point it represents.

Mio likes this fact a lot so she drops a map of her favorite park completely inside the park. The park P can be represented by a rectangle. A map of the park is just a smaller (or equal) version of the park printed on paper. The map is similar to the original rectangle. Each point on the map corresponds to a point in the park by the similarity transformation.

We can define a map formally: A map is a rectangle M (of smaller or equal size) together with a positive real number r and a bijective function $f : M \to P$ satisfying

- For every pair of different points $a, b \in M$, $|f(a) - f(b)|/|a - b| = r$.

$|x - y|$ represents the Euclidean distance between points x and y.

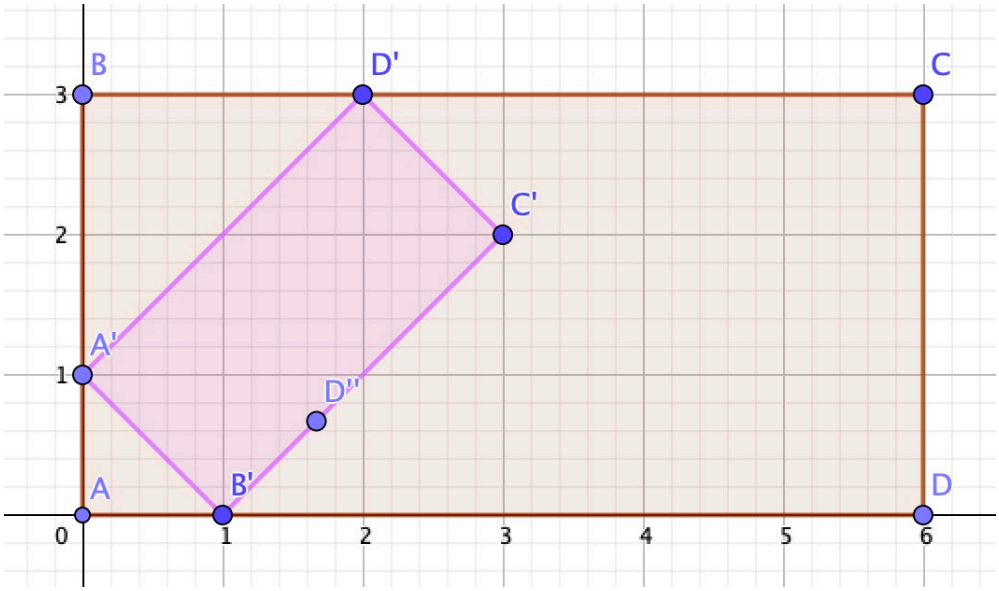
Like in many games, Mio can teleport using the map. Precisely, when Mio is at some point x on the map (including the boundary), she may teleport to the corresponding point $f(x)$ in the park. She may also choose not to teleport. The reverse is also true. When she is at point y in the park (including the boundary), she may teleport to the point $f^{-1}(y)$ on the map representing her current location. And she may also choose not to teleport.

Mio can teleport at most n (and at least 0) times. Each teleport takes k seconds. Mio can also walk on her foot at a speed of 1 unit per second.

Given two points s and t, find the minimum time Mio needs to reach t from s.

Each teleport can be in any direction (from the map to the park, or from the park to the map). The map may be placed upside down. Since the map is inside the park, it is possible that Mio is on the map and in the park simultaneously. In this case, she may teleport in either direction.

For example, in the following figure, the park is $ABCD$, and the map is $A'B'C'D'$. When Mio is inside the map, she is on the map and in the park simultaneously. When she is at point D', she can teleport from the map to the park (reaching D), and from the park to the map (reaching D'').



Input

The first line contains a single integer T ($1 \le T \le 100$) denoting the number of test cases.

For each test case, the first line contains the 4 corners of the rectangle representing the park. The corners are given in clockwise or counterclockwise order. It is guaranteed that the 4 corners are distinct.

The second line contains the 4 corners of the rectangle representing the map. The i-th corner of the map corresponds to the i-th corner of the park for all $1 \le i \le 4$. Note that you can figure out whether the map is placed upside down or not by the order of the corners. The corners are given in clockwise or counterclockwise order. It is guaranteed that the map is inside the park. (The boundary of the map may intersect with the boundary of the park at 1 or more points.) It is guaranteed that the map is valid, i.e., there is a positive real number and a bijective function from the map to the park satisfying the definition above.

The third line contains two points s and t. It is guaranteed that s and t are inside (or on the boundary of) the park.

The fourth line contains two integers k, n ($0 \le k \le 2 \times 10^6, 0 \le n \le 100$), the time each teleport needs, and the maximum number of teleports.

Each point in the input is represented by a pair of integers whose absolute values are no more than 2×10^6. Integers are separated by single spaces.

Output

For each test case, output one number representing the answer in one line. Your answer is considered correct if its absolute or relative error does not exceed 10^{-9}.

Example

standard input	standard output
2 0 0 0 2 4 2 4 0 0 0 0 1 2 1 2 0 2 1 4 2 1 1 0 0 0 3 6 3 6 0 0 1 1 0 3 2 2 3 0 0 4 2 0 3	1.0000000000 1.2272623352

Problem F. Inversion

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

This is an interactive problem.

There is a hidden permutation $p_1, p_2, \ldots, p_n$ of $\{1, 2, \ldots, n\}$. You want to find it by asking the parity of the number of inversions of $p_l, \ldots, p_r$.

You can query in the format "? l r", and the interactor will respond you $\left(\sum_{l \le i < j \le r} [p_i > p_j]\right) \bmod 2$. $[p_i > p_j]$ is 1 when $p_i > p_j$ and 0 when $p_i \le p_j$.

Interaction Protocol

Firstly, you should read the integer n ($1 \le n \le 2000$).

After that, you can make no more than 4×10^4 queries. To make a query, output "? l r" ($1 \le l \le r \le n$) on a separate line, then you should read the response from standard input.

To give your answer, print "! p_1 p_2 ... p_n" on a separate line. The output of the answer is **not** counted towards the limit of 4×10^4 queries.

After that, your program should terminate.

After printing a query, do not forget to output end of line and flush the output. To do this, use `fflush(stdout)` or `cout.flush()` in C++, `System.out.flush()` in Java, `flush(output)` in Pascal, or `stdout.flush()` in Python.

It is guaranteed that the permutation is fixed in advance.

Example

standard input	standard output
3 0 0 1	? 1 2 ? 1 3 ? 2 3 ! 2 3 1

Problem G. Rectangle

Input file:	`standard input`
Output file:	`standard output`
Time limit:	8 seconds
Memory limit:	1024 megabytes

Prof. Pang has n rectangles, the coordinate of the lower left corner of the i-th rectangle is $(x_{i,1}, y_{i,1})$, and the coordinate of the upper right corner is $(x_{i,2}, y_{i,2})$. Rectangles may overlap.

You need to choose three straight lines such that:

- Each line should be parallel to the x-axis or the y-axis, which means its formula is $x = a$ or $y = a$.
- In the formula $x = a$ or $y = a$, a should be an integer in $[1, 10^9]$.
- These three lines should be distinct.
- Each rectangle is **touched** by at least one line. A line touches a rectangle if it intersects with the boundary and/or the interior of the rectangle.

You need to compute the number of ways to choose three lines. Since the answer can be very large, output it modulo 998244353. Two ways are considered the same if only the order of three lines differs in these two ways.

Input

The first line contains a single integer T ($1 \le T \le 10^5$), denoting the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 10^5$). The i-th line of the next n lines contains four integers $x_{i,1}, y_{i,1}, x_{i,2}, y_{i,2}$ ($1 \le x_{i,1} < x_{i,2} \le 10^9, 1 \le y_{i,1} < y_{i,2} \le 10^9$).

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5.

Output

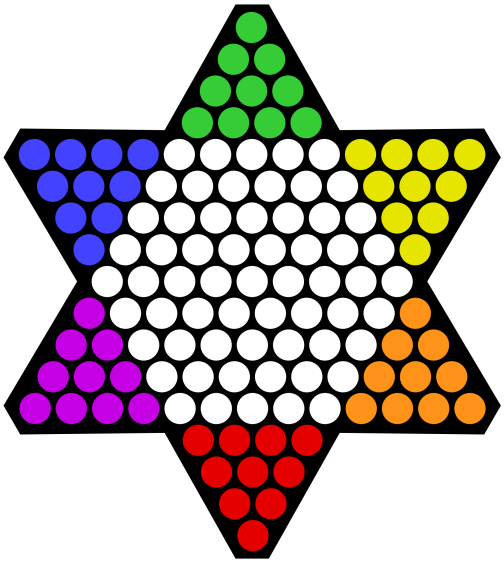
For each test case, output one integer representing the answer in one line.

Example

standard input	standard output
3 1 1 1 1000000000 1000000000 3 1 1 2 2 3 3 4 4 5 5 6 6 5 581574116 47617804 999010750 826131769 223840663 366320907 613364068 926991396 267630832 51913575 488301124 223957497 217461197 492085159 999485867 913732845 28144453 603781668 912516656 993160442	230616300 64 977066618

Problem H. Chinese Checker

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes



Prof. Pang is playing Chinese Checkers. The game board is the same as the figure shown above. There are n checkers on the board. Prof. Pang wants to know how many different moves there are on the current board.

One **move** consists of several **steps**. At first, Prof. Pang needs to choose one checker a to move. In each step, Prof. Pang needs to choose another checker b as the pivot, and move the checker a to the symmetrical position about the checker b. (In one move, Prof. Pang cannot change his choice of a between steps. **If after a step, the checker a will return to its position before the move, this step is not allowed.**) There are several conditions about the pivot b:

- The segment connecting a and b needs to be parallel to one of the coordinate axis. Note: There are three axes on the hexagonal board. One of them is horizontal and any pair of axes intersect at an angle of $\pi/3$.
- a and b **do not** need to be adjacent.
- There cannot be extra checkers other than b on the segment connecting a and its symmetrical position.
- The symmetrical position should be on the hexagonal board and is not occupied by any other checker.

A move must have at least one step. After the first step, Prof. Pang can stop at any time he wants. And Prof. Pang can choose any checker on the board as the moving checker. Output the number of different moves Prof. Pang can make. Two moves are different if and only if the sets of positions of all checkers are different after these two moves, i.e., the checkers are indistinguishable.

Input

The first line contains an integer T ($1 \le T \le 100$) – the number of test cases.

For each test case, the first line contains an integer n ($1 \le n \le 121$) – the number of checkers.

Each of the following n lines contains two integers, indicating the position of a checker. The first number indicates which row it is in, and the second number indicates which one of this row it is. They are counting from top to bottom and left to right, starting from 1.

It is guaranteed that checkers' positions are different.

Output

For each test case, output one integer in a line – the number of different moves.

Example

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

Problem I. Chase Game

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof. Shou is being chased by Prof. Pang on an undirected unweighted simple graph. Initially, Prof. Shou is at vertex 1. His destination is vertex n. Prof. Pang is at vertex k.

In each second, Prof. Shou may choose an adjacent vertex and walk to that vertex. Then Prof. Shou is attacked by Prof. Pang. The damage of this attack is equal to $d - dis$ where d is Prof. Pang's attack range and dis is the distance (number of edges in the shortest path) between Prof. Shou and Prof. Pang on the graph. However, when dis is greater than or equal to d, Prof. Pang cannot deal any positive damage. In this case, instead of attacking with non-positive damage, he will teleport to the vertex where Prof. Shou is and then deal d damage. (When dis is less than d, Prof. Pang will stay at his current vertex.)

Please find the minimum sum of damage Prof. Shou will take to reach vertex n from vertex 1. Prof. Shou will take the last attack at vertex n.

Input

The first line contains 4 integers n, m, k, d ($2 \le n \le 10^5, n-1 \le m \le 2 \times 10^5, 1 \le k \le n, 1 \le d \le 2 \times 10^5$).

Each of the next m lines contains two integers a, b ($1 \le a, b \le n, a \ne b$) representing an edge of the graph. The edges are distinct. (a b and b a represents the same edge. Thus, only one of these two lines may appear in the input.)

It is guaranteed that the graph is connected.

Output

Output one integer representing the answer in one line.

Examples

standard input	standard output
5 5 3 1 1 2 2 4 4 5 1 3 3 5	2
13 17 12 3 1 2 2 3 3 4 4 13 5 13 7 8 7 9 7 10 7 11 7 6 12 7 1 8 8 9 9 10 10 11 11 6 6 13	7

Problem J. Chase Game 2

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Prof. Pang and Prof. Shou like playing a chase game.

The game map consists of n rooms and $n-1$ bi-directional channels. The game map is connected. That means, the map forms a tree.

At first, Prof. Pang is in room u, while Prof. Shou is in room v ($u \neq v$). Prof. Pang and Prof. Shou take turns to play, and Prof. Shou goes first. In one's turn, the player knows his own position and the other player's position and can decide either to stay in the current room or to move to another room which is connected with the current room directly by a channel. When Prof. Pang and Prof. Shou are in the same room, Prof. Shou is caught by Prof. Pang.

Prof. Pang and Prof. Shou are smart enough. Prof. Pang wants to catch Prof. Shou in a finite number of turns. Prof. Shou does not want to be caught by Prof. Pang in any finite number of turns.

Prof. Shou gets tired of being caught every time and finds Prof. Fei for help. Prof. Shou asks Prof. Fei to add some channels so that Prof. Pang cannot catch him in finite number of turns for any pair of initial rooms (u, v). Prof. Fei is lazy, so he hopes to add as few channels as possible. If no matter how to add the channels there is always a pair of rooms (u, v) such that Prof. Pang can catch Prof. Shou, output -1.

Input

The first line contains a single integer T ($1 \leq T \leq 10^4$) denoting the number of test cases.

For each test case, the first line contains a single integer n ($2 \leq n \leq 10^5$) denoting the number of rooms.

For the next $n-1$ lines, each line contains two integers u and v ($1 \leq u, v \leq n$) denoting a channel connecting room u and room v.

It is guaranteed that the sum of n over all test cases does not exceed 2×10^5, and the rooms and channels always form a tree.

Output

For each test case, print a number denoting the smallest number of added channels, or just print -1.

Example

standard input	standard output
4 2 1 2 4 1 2 2 3 3 4 4 1 2 2 3 2 4 5 1 2 2 3 3 4 3 5	-1 1 -1 2

Problem K. Magic

Input file:	`standard input`
Output file:	`standard output`
Time limit:	3 seconds
Memory limit:	16 megabytes

Warning: Unusual memory limit!

You are given a sequence $a_0, \ldots, a_{2n}$. Initially, all numbers are zero.

There are n operations. The i-th operation is represented by two integers l_i, r_i ($1 \le l_i < r_i \le 2n, 1 \le i \le n$), which assigns i to $a_{l_i}, \ldots, a_{r_i-1}$. It is guaranteed that all the $2n$ integers, $l_1, l_2, \ldots, l_n, r_1, r_2, \ldots, r_n$, are distinct.

You need to perform each operation exactly once, in arbitrary order.

You want to maximize the number of i ($0 \le i < 2n$) such that $a_i \neq a_{i+1}$ after all n operations. Output the maximum number.

Input

The first line contains an integer n ($1 \le n \le 5 \times 10^3$).

The i-th line of the next n lines contains a pair of integers l_i, r_i ($1 \le l_i < r_i \le 2n$). It is guaranteed that all the $2n$ integers, $l_1, l_2, \ldots, l_n, r_1, r_2, \ldots, r_n$, are distinct.

Output

Output one integer representing the answer in one line.

Example

standard input	standard output
5 2 3 6 7 1 9 5 10 4 8	9

Problem L. Aqre

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	1024 megabytes

Given an $n \times m$ matrix, you need to fill it with 0 and 1, such that:

- There cannot be **four** consecutive horizontal or vertical cells filled with the same number.
- The cells filled with 1 form a connected area. (Two cells are adjacent if they share an edge. A group of cells is said to be connected if for every pair of cells it is possible to find a path connecting the two cells which lies completely within the group, and which only travels from one cell to an adjacent cell in each step.)

Please construct a matrix satisfying the conditions above and has as many 1s as possible. Output the maximum number of 1s, and the matrix.

Input

The first line contains an integer T ($1 \le T \le 10^3$) – the number of test cases.

For each test case, the first line contains two integers n, m ($2 \le n, m \le 10^3$).

It is guaranteed that the sum of $n \cdot m$ over all test cases does not exceed 10^6.

Output

For each test case, output the maximum number of 1s in the first line. Then output the matrix in the following n lines. If there are multiple solution, output any.

Example

standard input	standard output
3 2 2 3 4 3 8	4 11 11 9 1110 1110 1110 18 11101110 10111011 11011011

Problem M. Dining Professors

Input file:	`standard input`
Output file:	`standard output`
Time limit:	1 second
Memory limit:	256 megabytes

Prof. Pang invited n professors to his banquet. The professors sit at a round table. For all i from 1 to n, professor i sits adjacent to professor $(i \bmod n) + 1$ and $((i + n - 2) \bmod n) + 1$.

Prof. Pang prepared n dishes. There are n positions on the table. Position i is in front of professor i. Professor i can access only the dishes placed at positions i, $(i \bmod n) + 1$, and $((i + n - 2) \bmod n) + 1$. Prof. Pang will place exactly one dish at each position.

Among the dishes, a of them are spicy and $n - a$ of them are not spicy. Some (possibly 0) professors are unable to have spicy food. If a professor can have spicy food, his/her **satisfaction level** is the number of dishes (no matter spicy or not) he/she can access. If a professor cannot have spicy food, his/her satisfaction level is the number of not spicy dishes he/she can access.

Prof. Pang knows whether each professor can have spicy food or not. Please help him to arrange the dishes on the table so that the sum of satisfaction levels over all the professors is maximized. Output the maximum sum.

Input

The first line contains two integers n, a $(3 \le n \le 10^5, 0 \le a \le n)$.

The second line contains n integers $b_1, \ldots, b_n$. b_i is 0 or 1. $b_i = 1$ means professor i can have spicy food. $b_i = 0$ means professor i cannot have spicy food.

Output

Output one integer representing the answer in one line.

Example

standard input	standard output
5 2 1 0 1 0 1	13