Educational Codeforces Round 55 (Rated for Div. 2)

A. Vasya and Book

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

Vasya is reading a e-book. The file of the book consists of n pages, numbered from 1 to n. The screen is currently displaying the contents of page x, and Vasya wants to read the page y. There are two buttons on the book which allow Vasya to scroll d pages forwards or backwards (but he cannot scroll outside the book). For example, if the book consists of 10 pages, and $d = 3$, then from the first page Vasya can scroll to the first or to the fourth page by pressing one of the buttons; from the second page — to the first or to the fifth; from the sixth page — to the third or to the ninth; from the eighth — to the fifth or to the tenth.

Help Vasya to calculate the minimum number of times he needs to press a button to move to page y.

Input

The first line contains one integer t ($1 \le t \le 10^3$) — the number of testcases.

Each testcase is denoted by a line containing four integers n, x, y, d ($1 \le n, d \le 10^9$, $1 \le x, y \le n$) — the number of pages, the starting page, the desired page, and the number of pages scrolled by pressing one button, respectively.

Output

Print one line for each test.

If Vasya can move from page x to page y, print the minimum number of times he needs to press a button to do it. Otherwise print -1.

Example

input
3 10 4 5 2 5 1 3 4 20 4 19 3
output
4 -1 5

Note

In the first test case the optimal sequence is: $4 \to 2 \to 1 \to 3 \to 5$.

In the second test case it is possible to get to pages 1 and 5.

In the third test case the optimal sequence is: $4 \to 7 \to 10 \to 13 \to 16 \to 19$.

B. Vova and Trophies

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

Vova has won n trophies in different competitions. Each trophy is either golden or silver. The trophies are arranged in a row.

The *beauty* of the arrangement is the length of the longest subsegment consisting of golden trophies. Vova wants to swap two trophies (not necessarily adjacent ones) to make the arrangement as beautiful as possible — that means, to maximize the length of the longest such subsegment.

Help Vova! Tell him the maximum possible beauty of the arrangement if he is allowed to do at most one swap.

Input

The first line contains one integer n ($2 \le n \le 10^5$) — the number of trophies.

The second line contains n characters, each of them is either G or S. If the i-th character is G, then the i-th trophy is a golden one, otherwise it's a silver trophy.

Output

Print the maximum possible length of a subsegment of golden trophies, if Vova is allowed to do at most one swap.

Examples

input
10 GGGSGGGSGG
output
7

input
4 GGGG
output
4

input
3 SSS
output
0

Note

In the first example Vova has to swap trophies with indices 4 and 10. Thus he will obtain the sequence "GGGGGGGSGS", the length of the longest subsegment of golden trophies is 7.

In the second example Vova can make no swaps at all. The length of the longest subsegment of golden trophies in the sequence is 4.

In the third example Vova cannot do anything to make the length of the longest subsegment of golden trophies in the sequence greater than 0.

C. Multi-Subject Competition

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

A multi-subject competition is coming! The competition has m different subjects participants can choose from. That's why Alex (the coach) should form a competition delegation among his students.

He has n candidates. For the i-th person he knows subject s_i the candidate specializes in and r_i — a skill level in his specialization (this level can be negative!).

The rules of the competition require each delegation to choose some subset of subjects they will participate in. The only restriction is that the **number of students from the team** participating in each of the **chosen** subjects should be the **same**.

Alex decided that each candidate would participate only in the subject he specializes in. Now Alex wonders whom he has to choose to maximize the total sum of skill levels of all delegates, or just skip the competition this year if every valid non-empty delegation has negative sum.

(Of course, Alex doesn't have any spare money so each delegate he chooses must participate in the competition).

Input

The first line contains two integers n and m ($1 \le n \le 10^5$, $1 \le m \le 10^5$) — the number of candidates and the number of subjects.

The next n lines contains two integers per line: s_i and r_i ($1 \le s_i \le m$, $-10^4 \le r_i \le 10^4$) — the subject of specialization and the skill level of the i-th candidate.

Output

Print the single integer — the maximum total sum of skills of delegates who form a valid delegation (according to rules above) or 0 if every valid non-empty delegation has negative sum.

Examples

input
6 3 2 6 3 6 2 5 3 5 1 9 3 1
output
22

input
5 3 2 6 3 6 2 5 3 5 1 11
output
23

input
5 2 1 -1 1 -5 2 -1 2 -1 1 -10
output
0

Note

In the first example it's optimal to choose candidates 1, 2, 3, 4, so two of them specialize in the 2-nd subject and other two in the 3-rd. The total sum is $6 + 6 + 5 + 5 = 22$.

In the second example it's optimal to choose candidates 1, 2 and 5. One person in each subject and the total sum is $6 + 6 + 11 = 23$.

In the third example it's impossible to obtain a non-negative sum.

D. Maximum Diameter Graph

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

Graph constructive problems are back! This time the graph you are asked to build should match the following properties.

The graph is connected if and only if there exists a path between every pair of vertices.

The diameter (aka "longest shortest path") of a connected undirected graph is the maximum number of edges in the **shortest** path between any pair of its vertices.

The degree of a vertex is the number of edges incident to it.

Given a sequence of n integers $a_1, a_2, \ldots, a_n$ construct a **connected undirected** graph of n vertices such that:

- the graph contains no self-loops and no multiple edges;
- the degree d_i of the i-th vertex doesn't exceed a_i (i.e. $d_i \leq a_i$);
- the diameter of the graph is maximum possible.

Output the resulting graph or report that no solution exists.

Input

The first line contains a single integer n ($3 \leq n \leq 500$) — the number of vertices in the graph.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq n - 1$) — the upper limits to vertex degrees.

Output

Print "NO" if no graph can be constructed under the given conditions.

Otherwise print "YES" and the diameter of the resulting graph in the first line.

The second line should contain a single integer m — the number of edges in the resulting graph.

The i-th of the next m lines should contain two integers v_i, u_i ($1 \leq v_i, u_i \leq n$, $v_i \neq u_i$) — the description of the i-th edge. The graph should contain no multiple edges — for each pair (x, y) you output, you should output no more pairs (x, y) or (y, x).

Examples

input
3 2 2 2
output
YES 2 2 1 2

2 3

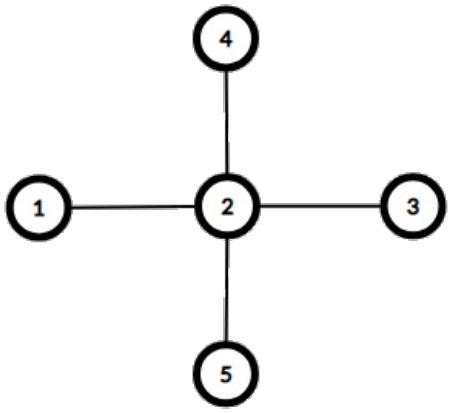
input
5 1 4 1 1 1
output
YES 2 4 1 2 3 2 4 2 5 2

input
3 1 1 1
output
NO

Note

Here are the graphs for the first two example cases. Both have diameter of 2.

$$d_1 = 1 \leq a_1 = 2$$
$$d_2 = 2 \leq a_2 = 2$$
$$d_3 = 1 \leq a_3 = 2$$



$$d_1 = 1 \leq a_1 = 1$$
$$d_2 = 4 \leq a_2 = 4$$
$$d_3 = 1 \leq a_3 = 1$$
$$d_4 = 1 \leq a_4 = 1$$

E. Increasing Frequency

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

You are given array a of length n. You can choose one segment $[l, r]$ ($1 \leq l \leq r \leq n$) and integer value k (positive, negative or even zero) and change $a_l, a_{l+1}, \ldots, a_r$ by k each (i.e. $a_i := a_i + k$ for each $l \leq i \leq r$).

What is the maximum possible number of elements with value c that can be obtained after one such operation?

Input

The first line contains two integers n and c ($1 \leq n \leq 5 \cdot 10^5$, $1 \leq c \leq 5 \cdot 10^5$) — the length of array and the value c to obtain.

The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \leq a_i \leq 5 \cdot 10^5$) — array a.

Output

Print one integer — the maximum possible number of elements with value c which can be obtained after performing operation described above.

Examples

input
6 9 9 9 9 9 9 9
output
6

input
3 2 6 2 6
output
2

Note

In the first example we can choose any segment and $k = 0$. The array will stay same.

In the second example we can choose segment $[1, 3]$ and $k = -4$. The array will become $[2, -2, 2]$.

F. Speed Dial

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

Polycarp's phone book contains n phone numbers, each of them is described by s_i — the number itself and m_i — the number of times Polycarp dials it in daily.

Polycarp has just bought a brand new phone with an amazing *speed dial* feature! More precisely, k buttons on it can have a number assigned to it (not necessary from the phone book). To enter some number Polycarp can press one of these k buttons and then finish the number using usual digit buttons (entering a number with only digit buttons is also possible).

Speed dial button can only be used when no digits are entered. No button can have its number reassigned.

What is the minimal total number of **digit number presses** Polycarp can achieve after he assigns numbers to *speed dial* buttons and enters each of the numbers from his phone book the given number of times in an optimal way?

Input

The first line contains two integers n and k ($1 \le n \le 500$, $1 \le k \le 10$) — the amount of numbers in Polycarp's phone book and the number of *speed dial* buttons his new phone has.

The i-th of the next n lines contain a string s_i and an integer m_i ($1 \le m_i \le 500$), where s_i is a non-empty string of digits from 0 to 9 inclusive (the i-th number), and m_i is the amount of times it will be dialed, respectively.

It is guaranteed that the total length of all phone numbers will not exceed 500.

Output

Print a single integer — the minimal total number of **digit number presses** Polycarp can achieve after he assigns numbers to *speed dial* buttons and enters each of the numbers from his phone book the given number of times in an optimal way.

Examples

input
3 1 0001 5 001 4 01 1
output
14

input
3 1 0001 5 001 6 01 1
output
18

Note

The only *speed dial* button in the first example should have "0001" on it. The total number of digit button presses will be $0 \cdot 5$ for the first number + $3 \cdot 4$ for the second + $2 \cdot 1$ for the third. 14 in total.

The only *speed dial* button in the second example should have "00" on it. The total number of digit button presses will be $2 \cdot 5$ for the first number + $1 \cdot 6$ for the second + $2 \cdot 1$ for the third. 18 in total.

G. Petya and Graph

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

Petya has a simple graph (that is, a graph without loops or multiple edges) consisting of n vertices and m edges.

The weight of the i-th vertex is a_i.

The weight of the i-th edge is w_i.

A subgraph of a graph is some set of the graph vertices and some set of the graph edges. The set of edges must meet the condition: both ends of each edge from the set must belong to the chosen set of vertices.

The weight of a subgraph is the sum of the weights of its edges, minus the sum of the weights of its vertices. You need to find the maximum weight of subgraph of given graph. **The given graph does not contain loops and multiple edges**.

Input

The first line contains two numbers n and m ($1 \le n \le 10^3, 0 \le m \le 10^3$) - the number of vertices and edges in the graph, respectively.

The next line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 10^9$) - the weights of the vertices of the graph.

The following m lines contain edges: the i-e edge is defined by a triple of integers v_i, u_i, w_i ($1 \le v_i, u_i \le n, 1 \le w_i \le 10^9, v_i \ne u_i$). This triple means that between the vertices v_i and u_i there is an edge of weight w_i. It is guaranteed that the graph does not contain loops and multiple edges.

Output

Print one integer — the maximum weight of the subgraph of the given graph.

Examples

input
4 5 1 5 2 2 1 3 4 1 4 4 3 4 5 3 2 2 4 2 2
output
8

input
3 3 9 7 8 1 2 1 2 3 2 1 3 3
output
0

Note

In the first test example, the optimal subgraph consists of the vertices $1, 3, 4$ and has weight $4 + 4 + 5 - (1 + 2 + 2) = 8$. In the second test case, the optimal subgraph is empty.