2015 ACM South China Normal University Collegiate Programming Contest Final Round

Problem A. Goddess Hunter

It is well known that *Great Master* is a **pickup artist**.

Therefore N boys ask *Great Master* to help them to fall in love with the girls they admire. If they fall in love, they will give a large number of money to *Great Master*. Obviously, one girl can only fall in love with one boy, and one boy can only fall in love with one girl.

Now there are M girls. Because Great Master is good at love, he clearly know which boy and which girl can fall in love with each other. Because not all boys can find his girlfriend, *Great Master* want to know how much money he can get at most. Can you help him?

In addition, If *Great Master* think two people can fall in love with each other, Great Master always has way to help them to go together.

Input

The first line is a positive integer $T(1 \le T \le 20)$, represents the number of case. The next T blocks follow each indicates a case.

Each case starts with two positive integers N and M, which indicates the number of boys and girls respectively. Boys are numbered from 1 to N. Girls are numbered from 1 to M.

In the next line, contains N positive integers c, the i^{th} integer indicates the money that the boy with number i will give to Great Master if he can fall in love with a girl.

In next N lines, each line contains M integers, which is 0 or 1. If the number of the j^{th} integer in i^{th} line is 1, it means the boy i can fall in love with girl j. Otherwise, they will never go together.

For 80% of cases, $0 < N, M \le 100$.

For 100% of cases, $0 < N, M \le 300$, $c \le 100000$.

Output

For each case, output the number of money that *Great Master* can get at most.

Sample input and output

Sample Input	Sample Output
2 2 2 100000 1 1 0 1 0 2 2 100000 1 1 0 0 1	100000 100001

Problem B. No Title

Give you a sequence $a_1, a_2, \cdots, a_n$.

You can divided a integer into two pairs, but one integer should be divided only once and each pair of it must be greater than 0. For example, you can divide 3 into (1 | 2), but you can't divide it into (1 | 1 | 1) or (0 | 3). Dividing an integer x cost x unit of money.

After all you dividing operator, all the adjacent integer will merge into one integer and become a new sequence b. If one integer divided into two pairs, they are considered non-adjacent.

For example, sequence 2 5 3 7. We split 3 and 2, cost 5 unit of money. And then it become 1 | 1 5 2 | 1 7, and then become sequence b : 1 8 8.

If the maximum integer of b must be smaller than or equal to a given integer M, what's the minimum cost? If it's no valid solution, output -1 as the answer.

Input

The first line is a positive integer $T (T \leq 100)$, represents the number of case. The next T blocks follow each indicates a case.

Each case starts with two positive integers n and M.

In the next line contains n positive integers indicate the sequence of a.

For 100% of cases, $n \leq 1000$ and $a_i \leq 1000$, $M \leq 1000000000$.

Output

For each case, output one line contains the answer of the problem.

Sample input and output

Sample Input	Sample Output
2 4 8 2 5 3 7 3 2 3 3 3	5 -1

Problem C. Rank of Sum

Give you a sequence $a_1, a_2, \cdots, a_n$.
We define $sum(i,j) = \sum_{x=i}^{j} a_x, i \le j$

If we put all of $sum(i,j)$ in a sequence B with $\frac{n(n+1)}{2}$ elements and sort them from small to large. Now give you a positive integer K. Can you tell me the K^{th} element in B?

Input

The first line is a number $T(1 \le T \le 20)$, represents the number of case. The next T blocks follow each indicates a case.

Each case starts with two positive integers n and K.
In the next line, contains n integers $a_1, a_2, \cdots, a_n$.

For 80% of cases, $N \le 1000$, $-1000 \le a_i \le 1000$
For 100% of cases, $N \le 50000$, $1 \le K \le \frac{n(n+1)}{2}$, $-10000 \le a_i \le 10000$

Output

For each case, output the K^{th} smallest number of B in a line.

Sample input and output

Sample Input	Sample Output
2 5 15 1 2 3 4 5 2 2 -1 -2	15 -2

Problem D. String Counter

You are given a string s with N lower case letters. The index of s starts with 0. Your task is to deal with M operations of 2 types:

1. Given an integer p and a lower case letter q, you should change the p^{th} character of the string s to q.
2. Given a lower case letter q, you should print the number of q in string s.

Input

The first line is a positive integer $T (1 \le T \le 20)$, which represents the number of case. The next T blocks follow each indicates a case.

Each case starts with a string s.

In next line contains an integer M.

In next M lines, each line begin with an integer $op (1 \le op \le 2)$.

If $op = 1$, there are an integer p and a lower case letter q after op, means that you should change the p^{th} character of s to q.

If $op = 2$, there is a lower case letter q after op, means that you should print the number of q in string s.

For 70% of cases, the length of $s \le 1000$, $M \le 1000$.

For 100% of cases, the length of $s \le 100000$, s only contains lower case letter, $0 \le M \le 100000$, $0 \le p < N$, q must be lower-case letter.

Output

For every operation 2 in the input, print the answer in a line.

Sample input and output

Sample Input	Sample Output
2 acmer 2 2 a 2 b greatmaster 5 2 a 2 t 1 4 a 2 a 2 t	1 0 2 2 3 1

Problem E. String Game

You are given n string $s_1, s_2, \cdots, s_n$, which only contain 'a' $\cdots$ 'e'. String s_i has a value v_i. There are three kinds of operation:

1. Cv x v': change v_x into v'
2. Cs x a': change s_x into a'
3. Q: calculate $ans = \max \sum_{x=1}^{pn} s_{p_x}$, satisfying that s_{p_x} is a prefix of $s_{p_{x+1}} (1 \le i < n)$.

Please note that $pn \le n$, $1 \le p_i \le n$ and $p_i \ne p_j (i \ne j)$

String $a = a_1 a_2 \cdots a_{an}$ is a prefix of $b = b_1 b_2 \cdots b_{bn}$ if and only if $an <= bn$ and $a_i = b_i (1 \le i \le an)$.

Input

The first line is a positive integer $T (T \le 2)$, which represents the number of cases. The next T blocks follow each indicates a case.

Each case starts with two positive integers n and m.

In the next n lines, the i^{th} line contains a string s_i.

In the next line, contains n integers. The i^{th} integer indicates v_i.

In next m lines, each line contains a operation. There are three kinds of operation whose means is shown in description.

Please note that some operation of input are encrypted. If the last operation Q print *last-ans*(you can assume that *last-ans* is equal to 0 in the beginning), all a' in operation Cs must add *last-ans*(from 'a' to 'e' circulation).

For example, the operation is "Cs 1 abcde". If *last-ans* is 1, You must translate it into "Cs 1 bcdea". If *last-ans* is 2, You must translate it into "Cs 1 cdeab".

For 100% of cases, $n \le 50000$, $m \le 100000$, $0 \le v_i < 100000$, $\sum s_i + \sum a' \le 1000000$.

Output

For every operation Q in the input, print ans in a line.

Sample input and output

Sample Input	Sample Output
2 5 5 aba ab babb abaa abab 0 1 4 2 3 Q Cv 1 2 Q Cs 3 eaeea Q 2 2 aa bb 1 2 Cs 2 aa Q	4 6 9 3

Problem F. Villa Builder

Yuki wants to build a villa. She find a rectangle field. It can see as a rectangle with N * M cells, with bottom-left (1,1) and upper-right cell (N, M). The villa *Yuki* want to build must cover a rectangle with W * H cells. However, some cells of the field are broken that villa can't be built on it. And the villa rectangle's edge must parallel to the rectangle field's edge.

There are K broken cells and the coordinates of these broken cells. Please tell *Yuki* how many ways to select a W * H rectangle on this field without broken cells. Two ways are considered different if and only if two villa's bottom-left coordinate or upper-right coordinate are different respectively.

Input

The first line is a positive integer $T(T \leq 20)$, which represents the number of case. The next T blocks follow each indicates a case.

Each case starts with four integers N, M, W and H.

The next line contains a integer K.

In the next K lines, each line contains two positive integers x, y indicates a broken cell's coordinate (x, y). Two broken cells may have same coordinate.

For 80% of cases, $0 < N, M, W, H \leq 1000$.

For 100% of cases, $0 < N, M, W, H \leq 50000$, $0 \leq K \leq 50000$, $0 < x \leq N$, $0 < y \leq M$.

Output

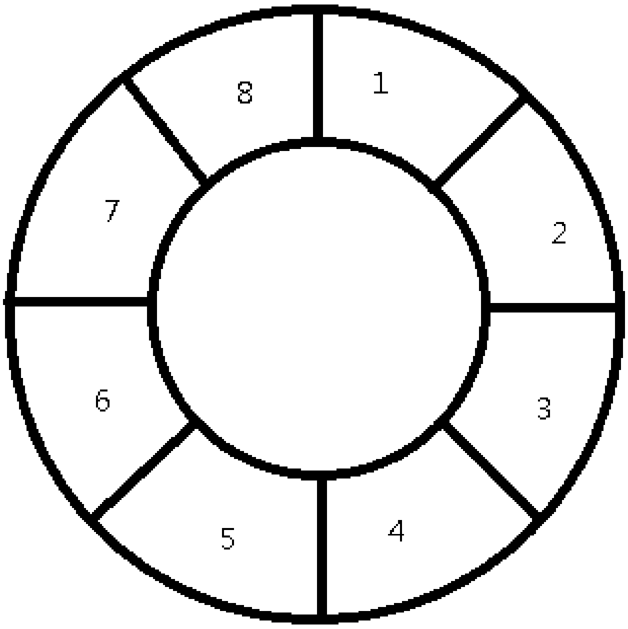
For each case, output the number of ways to select the rectangle.

Sample input and output

Sample Input	Sample Output
2 3 3 2 3 2 1 1 1 1 3 3 1 2 1 2 2	2 8

Problem G. Oyk Cut Paper Again

There is a cyclic paper tape . Now divide the tape into N grids as shown below.



$N = 8$

Oyk and *LGH* take turns to play a cut paper game. In one's turn , the player can take away smaller than K continues grids(some he need to cut the paper tape). Who take away the last grid win the game. We assume *Oyk* always play in first turn.

Input

There are multiple test cases. Each test case contains two integer N and K in one line. $(1 < N, K < 10^9)$

Output

As we all know , *Oyk* and *LGH* is clever. If *Oyk* can win the game , print *Yes* . Otherwise print *No*. Each test case in one line.

Sample input and output

Sample Input	Sample Output
20 7 100 5	No No

Problem H. Matrix

Now you have an $n * n$ board. Every grid has two positions, *UP* and *DOWN*. In this game you can push some amazing buttons to exchange any two rows or any two columns. You will win if you got the grids in the main diagonal line all *UP*.

But you are so clever that you realize for some board, no matter how many times you tries, you cannot get the grids in the main diagonal line all *UP*. So you want to know whether she can win this board or not.

Input

There are several test cases.

For each test case: The 1st line contains 1 integer n, indicating the size of the board. $(1 \leq n \leq 200)$ The next n lines, each contains n characters. *'U'* indicates the position *UP*, and *'D'* indicates the position *DOWN*.

There is no separation line between any two test cases.

Output

For each test case, you should print one line. You should print $'YES'$ if Alice can win, print $'NO'$ if not.

Sample input and output

Sample Input	Sample Output
3 DUD UDD DDU 3 DUD DUD UDD	*YES* *NO*

Problem I. Nightmare

Now we have two arrays of integers A and B. By multiplying each other elements between A and B array, we can get an array C with the size of $|A| * |B|$. Can you tell me the value of the k^{th} largest elements of C?

Input

The first line is a positive integer $T (T \leq 20)$, represents the number of case. The next T blocks follow each indicates a case.

Each case starts with two positive integers n and k. The integer n indicates the size of A and B, and the integer k indicates the k^{th} largest element is required. In the following two lines, each line contains n integers. The first line includes the elements of A, and the second line includes the elements of B.

For all of cases, $1 \leq n \leq 100000$,$1 \leq k \leq n * n$, the value range of each element in array A and B is $(0, 10000]$.

Output

For each case, output one line contains the value of the k^{th} largest element of C.

Sample input and output

Sample Input	Sample Output
2 2 1 3 4 5 6 2 3 2 1 4 8	24 8

Problem J. The xor-longest Path

In an edge-weighted tree, the xor-length of a path p is defined as the xor sum of the weights of edges on p($\oplus$ is the xor operator.):

$$length_{xor}(p) = \oplus_{e \in p} weight(e)$$

We say a path the xor-longest path if it has the largest xor-length. Given an edge-weighted tree with n nodes, can you find the xor-longest path?

Input

The input contains several test cases. The first line of each test case contains an integer $n(1 \leq n \leq 100000)$, The following $n-1$ lines each contains three integers u$(0 \leq$ u $<$ n), $v(0 \leq v < n)$, $w(0 \leq w < 2^{31})$, which means there is an edge between node u and v of length w.

Output

For each test case output the xor-length of the xor-longest path.

Sample input and output

Sample Input	Sample Output
4 0 1 3 1 2 4 1 3 6	7