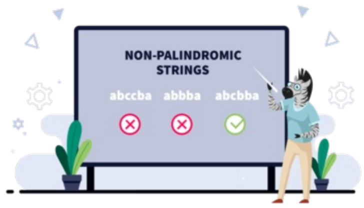
A. String Generation

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

One fall day Joe got bored because he couldn't find himself something interesting to do. Marty suggested Joe to generate a string of length n to entertain him somehow. It didn't seem particularly difficult, but Joe's generated string had to follow these rules:

- the string may only contain characters 'a', 'b', or 'c';
- the maximum length of a substring of this string that is a palindrome does not exceed k.



A string a is a substring of a string b if a can be obtained from b by deletion of several (possibly, zero or all) characters from the beginning and several (possibly, zero or all) characters from the end. For example, strings "a", "bc", "abc" are substrings of a string "abc", while strings "ac", "ba", "cba" are not.

A string is a palindrome if it reads the same from the left to the right and from the right to the left. For example, strings "abccba", "abbba", "aba", "abacaba", "a", and "bacab" are palindromes, while strings "abcbba", "abb", and "ab" are not.

Now Joe wants to find any correct string. Help him! It can be proven that the answer always exists under the given constraints.

Input

Each test contains one or more test cases. The first line contains the number of test cases t ($1 \le t \le 10$).

The only line of each test case contains two integers n and k ($1 \le k \le n \le 1\,000$) — the required string length and the maximum length of a palindrome substring, respectively.

Output

For each test case, print any string that satisfies the conditions from the problem statement. If there are multiple correct answers, you can print any one of them. It can be proven that the answer always exists under the given constraints.

Example

input

```
2
3 2
4 1
```

output

```
aab
acba
```

Note

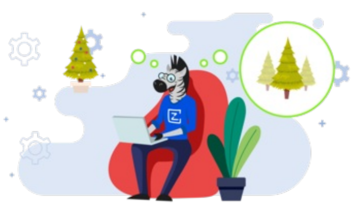
In the first test case of the example, the palindrome substring with the maximum length is "aa". Its length does not exceed 2, so it fits.

In the second test case all palindrome substrings have the length one.

B. Find the Spruce

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

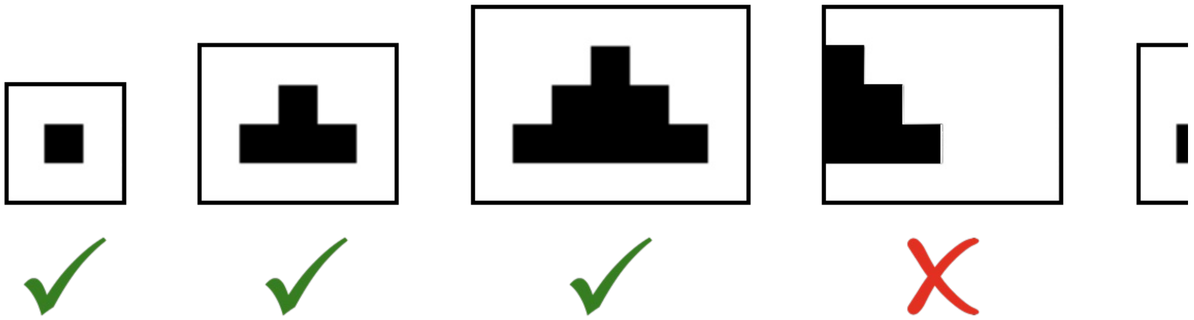
Holidays are coming up really soon. Rick realized that it's time to think about buying a traditional spruce tree. But Rick doesn't want real trees to get hurt so he decided to find some in an $n \times m$ matrix consisting of "*" and ".".



To find every spruce first let's define what a spruce in the matrix is. A set of matrix cells is called a spruce of height k with origin at point (x, y) if:

- All cells in the set contain an "*".
- For each $1 \le i \le k$ all cells with the row number $x+i-1$ and columns in range $[y-i+1, y+i-1]$ must be a part of the set. All other cells cannot belong to the set.

Examples of correct and incorrect spruce trees:



Now Rick wants to know how many spruces his $n \times m$ matrix contains. Help Rick solve this problem.

Input

Each test contains one or more test cases. The first line contains the number of test cases t ($1 \le t \le 10$).

The first line of each test case contains two integers n and m ($1 \le n, m \le 500$) — matrix size.

Next n lines of each test case contain m characters $c_{i,j}$ — matrix contents. It is guaranteed that $c_{i,j}$ is either a "." or an "*".

It is guaranteed that the sum of $n \cdot m$ over all test cases does not exceed 500^2 ($\sum n \cdot m \le 500^2$).

Output

For each test case, print single integer — the total number of spruces in the matrix.

Example

input

```
4
2 3
.*.
***
2 3
.*.
**.
4 5
.***.
*****
*****
*.*.*
5 7
..*.*..
.*****.
*******
.*****.
..*.*..
```

output

```
5
3
23
34
```

Note

In the first test case the first spruce of height 2 has its origin at point $(1, 2)$, the second spruce of height 1 has its origin at point $(1, 2)$, the third spruce of height 1 has its origin at point $(2, 1)$, the fourth spruce of height 1 has its origin at point $(2, 2)$, the fifth spruce of height 1 has its origin at point $(2, 3)$.

In the second test case the first spruce of height 1 has its origin at point $(1, 2)$, the second spruce of height 1 has its origin at point $(2, 1)$, the third spruce of height 1 has its origin at point $(2, 2)$.

C. Random Events

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

Ron is a happy owner of a permutation a of length n.

A permutation of length n is an array consisting of n distinct integers from 1 to n in arbitrary order. For example, $[2, 3, 1, 5, 4]$ is a permutation, but $[1, 2, 2]$ is not a permutation (2 appears twice in the array) and $[1, 3, 4]$ is also not a permutation ($n = 3$ but there is 4 in the array).

Ron's permutation is subjected to m experiments of the following type: (r_i, p_i). This means that elements in range $[1, r_i]$ (in other words, the prefix of length r_i) have to be sorted in ascending order with the probability of p_i. All experiments are performed in the same order in which they are specified in the input data.

As an example, let's take a look at a permutation $[4, 2, 1, 5, 3]$ and an experiment $(3, 0.6)$. After such an experiment with the probability of 60% the permutation will assume the form $[1, 2, 4, 5, 3]$ and with a 40% probability it will remain unchanged.

You have to determine the probability of the permutation becoming completely sorted in ascending order after m experiments.

Input

Each test contains one or more test cases. The first line contains the number of test cases t ($1 \le t \le 100$).

The first line of each test case contains two integers n and m $(1 \le n, m \le 10^5)$ — the length of the permutation and the number of experiments, respectively.

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ $(1 \le a_i \le n)$ — contents of the permutation.

The following m lines of each test case each contain an integer r_i and a real number p_i $(1 \le r_i \le n, 0 \le p_i \le 1)$ — the length of the prefix and the probability of it being sorted. All probabilities are given with at most 6 decimal places.

It is guaranteed that the sum of n and the sum of m does not exceed 10^5 ($\sum n, \sum m \le 10^5$).

Output

For each test case, print a single number — the probability that after all experiments the permutation becomes sorted in ascending order. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Formally, let your answer be a, and the jury's answer be b. Your answer is accepted if and only if $\frac{|a-b|}{\max(1,|b|)} \le 10^{-6}$.

Example

input

```
4
4 3
4 3 2 1
1 0.3
3 1
4 0.6
5 3
4 2 1 3 5
3 0.8
4 0.6
5 0.3
6 5
1 3 2 4 5 6
4 0.9
5 0.3
2 0.4
6 0.7
3 0.5
4 2
1 2 3 4
2 0.5
4 0.1
```

output

```
0.600000
0.720000
0.989500
1.000000
```

Note

Explanation of the first test case: It can be demonstrated that whether the final permutation is sorted or not depends solely on sorting being performed in the $(4, 0.6)$ experiment.

D. Divide and Summarize

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

Mike received an array a of length n as a birthday present and decided to test how pretty it is.

An array would pass the i-th prettiness test if there is a way to get an array with a sum of elements totaling s_i, using some number (possibly zero) of slicing operations.

An array slicing operation is conducted in the following way:

- assume $mid = \lfloor \frac{max(array)+min(array)}{2} \rfloor$, where max and min — are functions that find the maximum and the minimum array elements. In other words, mid is the sum of the maximum and the minimum element of $array$ divided by 2 rounded down.
- Then the array is split into two parts $left$ and $right$. The $left$ array contains all elements which are less than or equal mid, and the $right$ array contains all elements which are greater than mid. Elements in $left$ and $right$ keep their relative order from $array$.
- During the third step we choose which of the $left$ and $right$ arrays we want to keep. The chosen array replaces the current one and the other is permanently discarded.

You need to help Mike find out the results of q prettiness tests.

Note that you test the prettiness of the array a, so you start each prettiness test with the primordial (initial) array a. Thus, the first slice (if required) is always performed on the array a.

Input

Each test contains one or more test cases. The first line contains the number of test cases t $(1 \le t \le 100)$.

The first line of each test case contains two integers n and q $(1 \le n, q \le 10^5)$ — the length of the array a and the total number of prettiness tests.

The second line of each test case contains n integers $a_1, a_2, \dots, a_n$ $(1 \le a_i \le 10^6)$ — the contents of the array a.

Next q lines of each test case contain a single integer s_i $(1 \le s_i \le 10^9)$ — the sum of elements which Mike wants to get in the i-th test.

It is guaranteed that the sum of n and the sum of q does not exceed 10^5 $(\sum n, \sum q \le 10^5)$.

Output

Print q lines, each containing either a "Yes" if the corresponding prettiness test is passed and "No" in the opposite case.

Example

input

```
2
5 5
1 2 3 4 5
1
8
9
12
6
5 5
3 1 3 1 3
1
2
3
9
11
```

output

```
Yes
No
Yes
No
Yes
```

```
No
Yes
No
Yes
Yes
```

Note

Explanation of the first test case:

1. We can get an array with the sum $s_1 = 1$ in the following way:
   1.1 $a = [1, 2, 3, 4, 5]$, $mid = \frac{1+5}{2} = 3$, $left = [1, 2, 3]$, $right = [4, 5]$. We choose to keep the $left$ array.

   1.2 $a = [1, 2, 3]$, $mid = \frac{1+3}{2} = 2$, $left = [1, 2]$, $right = [3]$. We choose to keep the $left$ array.

   1.3 $a = [1, 2]$, $mid = \frac{1+2}{2} = 1$, $left = [1]$, $right = [2]$. We choose to keep the $left$ array with the sum equalling 1.
2. It can be demonstrated that an array with the sum $s_2 = 8$ is impossible to generate.
3. An array with the sum $s_3 = 9$ can be generated in the following way:
   3.1 $a = [1, 2, 3, 4, 5]$, $mid = \frac{1+5}{2} = 3$, $left = [1, 2, 3]$, $right = [4, 5]$. We choose to keep the $right$ array with the sum equalling 9.
4. It can be demonstrated that an array with the sum $s_4 = 12$ is impossible to generate.
5. We can get an array with the sum $s_5 = 6$ in the following way:
   5.1 $a = [1, 2, 3, 4, 5]$, $mid = \frac{1+5}{2} = 3$, $left = [1, 2, 3]$, $right = [4, 5]$. We choose to keep the $left$ with the sum equalling 6.

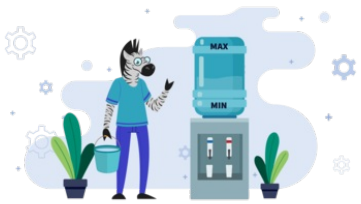
Explanation of the second test case:

1. It can be demonstrated that an array with the sum $s_1 = 1$ is imposssible to generate.
2. We can get an array with the sum $s_2 = 2$ in the following way:
   2.1 $a = [3, 1, 3, 1, 3]$, $mid = \frac{1+3}{2} = 2$, $left = [1, 1]$, $right = [3, 3, 3]$. We choose to keep the $left$ array with the sum equalling 2.
3. It can be demonstrated that an array with the sum $s_3 = 3$ is imposssible to generate.
4. We can get an array with the sum $s_4 = 9$ in the following way:
   4.1 $a = [3, 1, 3, 1, 3]$, $mid = \frac{1+3}{2} = 2$, $left = [1, 1]$, $right = [3, 3, 3]$. We choose to keep the $right$ array with the sum equalling 9.
5. We can get an array with the sum $s_5 = 11$ with zero slicing operations, because array sum is equal to 11.

E. Water Level

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

In recent years John has very successfully settled at his new job at the office. But John doesn't like to idly sit around while his code is compiling, so he immediately found himself an interesting distraction. The point of his distraction was to maintain a water level in the water cooler used by other zebras.



Originally the cooler contained exactly k liters of water. John decided that the amount of water must always be at least l liters of water but no more than r liters. John will stay at the office for exactly t days. He knows that each day exactly x liters of water will be used by his colleagues. At the beginning of each day he can add exactly y liters of water to the cooler, but at any point in time the amount of water in the cooler must be in the range $[l, r]$.

Now John wants to find out whether he will be able to maintain the water level at the necessary level for t days. Help him answer this question!

Input

The first line of the input contains six integers k, l, r, t, x and y ($1 \le l \le k \le r \le 10^{18}; 1 \le t \le 10^{18}; 1 \le x \le 10^6; 1 \le y \le 10^{18}$) — initial water level, the required range, the number of days, daily water usage and the exact amount of water that can be added, respectively.

Output

Print "Yes" if John can maintain the water level for t days and "No" otherwise.

Examples

input
```
8 1 10 2 6 4
```
output
```
No
```

input
```
8 1 10 2 6 5
```
output
```
Yes
```

input
```
9 1 10 9 2 9
```
output
```
No
```

input
```
20 15 25 3 5 7
```
output
```
Yes
```

Note

In the first example, John can't increase the amount of water at the beginning of the first day, since it would exceed the limit r. That is why after the first day the cooler will contain 2 liters of water. The next day John adds 4 liters to the cooler but loses 6 liters, leaving John with 0 liters, which is outside the range $[1, 10]$.

In the second example, after the first day John is left with 2 liters of water. At the beginning of the next day he adds 5 liters, then 6 liters get used, leaving John with 1 liter of water which is in range $[1, 10]$.

In the third example, after the first day John is left with 7 liters, after the second day — 5 liters, after the fourth — 1 liter. At the beginning of the fifth day John will add 9 liters and lose 2 liters. Meaning, after the fifth day he will have 8 liters left. Then each day the water level will decrease by 2 liters and after the eighth day John will have 2 liters and after the ninth day — 0 liters. 0 is outside range $[1, 10]$, so the answer is "No".

In the fourth example, after the first day John is left with 15 liters of water. At the beginning of the second day he adds 7 liters and loses 5, so after the second day he is left with 17 liters. At the beginning of the third day he adds 7 more liters of water and loses 5, so after the third day he is left with 19 liters. 19 is in range $[15, 25]$ so the answer is "Yes".

F. Mathematical Expression

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Barbara was late for her math class so as a punishment the teacher made her solve the task on a sheet of paper. Barbara looked at the sheet of paper and only saw n numbers $a_1, a_2, \ldots, a_n$ without any mathematical symbols. The teacher explained to Barbara that she has to place the available symbols between the numbers in a way that would make the resulting expression's value as large as possible. To find out which symbols were available the teacher has given Barbara a string s which contained that information.

It's easy to notice that Barbara has to place $n - 1$ symbols between numbers in total. The expression must start with a number and all symbols must be allowed (i.e. included in s). Note that multiplication takes precedence over addition or subtraction, addition and subtraction have the same priority and performed from left to right. Help Barbara and create the required expression!

Input

The first line of the input contains a single integer n ($1 \le n \le 10^5$) — the amount of numbers on the paper.

The second line of the input contains n integers $a_1, a_2, \ldots, a_n$ ($0 \le a_i \le 9$), where a_i is the i-th element of a.

The third line of the input contains the string s ($1 \le |s| \le 3$) — symbols allowed in the expression. It is guaranteed that the string may only consist of symbols "-", "+" and "*". It is also guaranteed that all symbols in the string are distinct.

Output

Print n numbers separated by $n - 1$ symbols — a mathematical expression with the greatest result. If there are multiple equally valid results — output any one of them.

Examples

input
```
3
2 2 0
+-*
```
output
```
2*2-0
```

input
```
4
2 1 1 2
+*
```
output
```
2+1+1+2
```

Note

The following answers also fit the first example: "2+2+0", "2+2-0", "2*2+0".