A. GSM

Xiao Ming is traveling around several cities by train. And the time on the train is very boring, so Xiao Ming will use the mobile Internet. We all know that mobile phone receives the signal from base station and it will change the base station when moving on the train. Xiao Ming would like to know how many times the base station will change from city A to city B.

Now, the problem is simplified. We assume the route of train is straight, and the mobile phone will receive the signal from the nearest base station.

Input

Multiple cases. For each case, The first line: N(3<=N<=50) - the number of cities, M(2<=M<=50) - the number of base stations. Then there are N cities with coordinates of (x, y) and M base stations with coordinates of (x, y) - (0<=x<=1000, 0<=y<=1000, both x and y is integer).Then there is a number : K, the next, there are K queries, for each query, each line, there are two numbers: a, b.

Output

For each query, tell Xiao Ming how many times the base station will change from city a to city b.

Sample Input

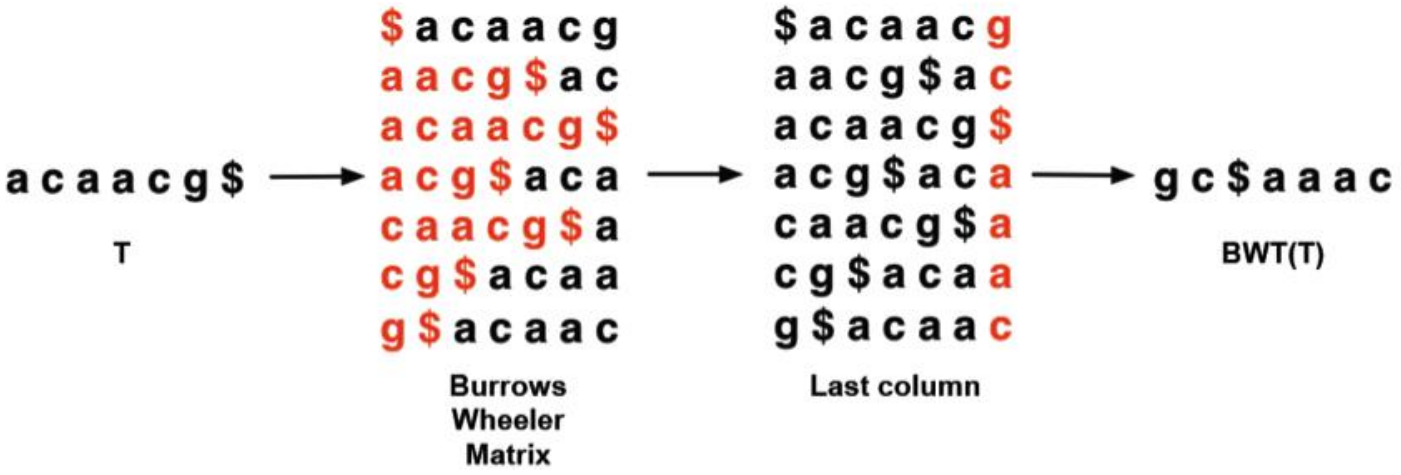
```
4 4
0 2
1 3
1 0
2 0
1 2
1 1
2 2
2 1
4
1 2
1 3
1 4
3 4
```

Sample Output

```
0
1
2
1
```

B. BWT

When the problem to match S string in T string is mentioned, people always put KMP, Aho-Corasick and Suffixarray forward. But Mr Liu tells Canoe that there is an algorithm called Burrows–Wheeler Transform(BWT) which is quite amazing and high-efficiency to solve the problem.
But how does BWT work to solve the matching S-in-T problem? Mr Liu tells Canoe the firstly three steps of it. Firstly, we append the '$' to the end of T and for convenience, we still call the new string T. And then for every suffix of T string which starts from i, we append the prefix of T string which ends at (i – 1) to its end. Secondly, we sort these new strings by the dictionary order. And we call the matrix formed by these sorted strings Burrows Wheeler Matrix. Thirdly, we pick characters of the last column to get a new string. And we call the string of the last column BWT(T). You can get more information from the example below.



Then Mr Liu tells Canoe that we only need to save the BWT(T) to solve the matching problem. But how and can it? Mr Liu smiles and says yes. We can find whether S strings like "aac" are substring of T string like "acaacg" or not only knowing the BWT(T)! What an amazing algorithm BWT is! But Canoe is puzzled by the tricky method of matching S strings in T string. Would you please help Canoe to find the method of it? Given BWT(T) and S string, can you help Canoe to figure out whether S string is a substring of string T or not?

Input

There are multiple test cases.
First Line: the BWT(T) string (1 <= length(BWT(T)) <= 100086).
Second Line: an integer n (1 <=n <= 10086) which is the number of S strings.
Then n lines comes.
There is a S string (n * length(S) will less than 2000000, and all characters of S are lowercase) in every line.

Output

For every S, if S string is substring of T string, then put out "YES" in a line. If S string is not a substring of T string, then put out "NO" in a line.

Sample Input

```
gc$aaac
2
aac
gc
```

Sample Output

```
YES
NO
```

C. Query on a tree VII

You are given a tree (an acyclic undirected connected graph) with n nodes. The tree nodes are numbered from 1 to n. Each node has a color, white or black, and a weight.
We will ask you to perfrom some instructions of the following form:
0 u : ask for the maximum weight among the nodes which are connected to u, two nodes are connected if all the node on the path from u to v (inclusive u and v) have a same color.
1 u : toggle the color of u(that is, from black to white, or from white to black).
2 u w: change the weight of u to w.

Input

Multicase.

The first line contains a number n denoted how many nodes in the tree($1 \le n \le 10^5$). The next n - 1 lines, each line has two numbers (u, v) describe a edge of the tree($1 \le u, v \le n$).

The next 2 lines, each line contains n number, the first line is the initial color of each node(0 or 1), and the second line is the initial weight, let's say Wi, of each node($|W_i| \le 10^9$).

The next line contains a number m denoted how many operations we are going to process($1 \le m \le 10^5$). The next m lines, each line describe a operation (t, u) as we mentioned above($0 \le t \le 2$, $1 \le u \le n$, $|w| \le 10^9$).

Output

For each query operation, output the corresponding result.

Sample Input

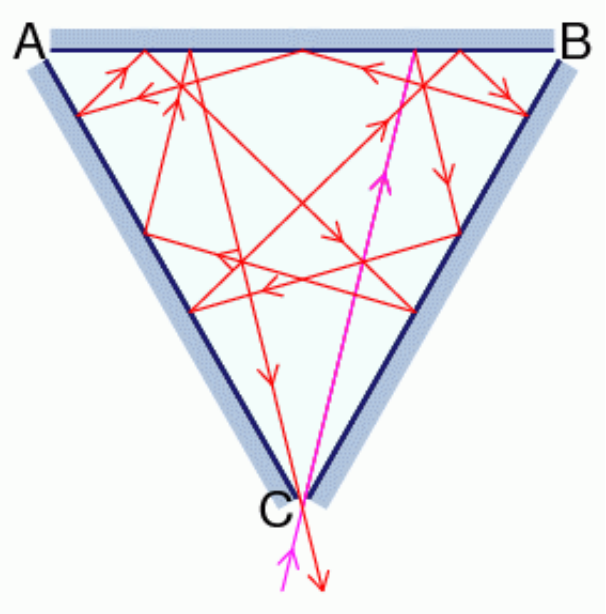
```
5
1 2
1 3
1 4
1 5
0 1 1 1 1
1 2 3 4 5
3
0 1
1 1
0 1
7
1 2
1 3
2 4
2 5
3 6
3 7
0 0 0 0 0 0 0
1 2 3 4 5 6 7
4
0 1
1 1
0 2
0 3
```

Sample Output

```
1
5
7
5
7
```

D. Laser Beam

There is an equilateral triangle consist of 3 mirrors. There is a tiny slit in the corners of the triangle, which can let a laser beam pass through.



We label the 3 slits as A, B and C. There only exists two ways (see the picture above) make a laser beam enter our triangle though C, reflects 11 times in ther triangle, and exit from the triangle though C. The 2 ways are symmetry.

Here is the question for you, our great programmer. How many ways we can make a laser beam enter the triangle though C and exit though C, and the beam reflects n times in the triangle? e.g. there are 80840 ways when n is 1000001.

Input

The first line contains a number T.($1 \le T \le 100$) Then the following T line each line contains a number n.($1 \le n \le 10^7$)

Output

For each n, print the corresponding result.

Sample Input

```
10
2
5
7
11
13
17
19
23
29
31
```

Sample Output

```
0
0
2
2
2
0
4
4
2
6
```

E. Another Graph Game

Alice and Bob are playing a game on an undirected graph with n (n is even) nodes and m edges. Every node i has its own weight W_v, and every edge e has its own weight W_e.

They take turns to do the following operations. During each operation, either Alice or Bob can take one of the nodes from the graph that haven't been taken before. Alice goes first.

The scoring rule is: One person can get the bonus attached to a node if he/she have choosen that node before. One person can get the bonus attached to a edge if he/she have choosen both node that induced by the edge before.

You can assume Alice and Bob are intelligent enough and do operations optimally, both Alice and Bob's target is maximize their score - opponent's.

What is the final result for Alice - Bob.

Input

Muilticases. The first line have two numbers n and m.($1 <= n <= 10^5$, $0<=m<=10^5$) The next line have n numbers from W_1 to W_n which W_i is the weight of node i.($|W_i|<=10^9$)

The next m lines, each line have three numbers u, v, w,($1\le u,v\le n,|w|<=10^9$) the first 2 numbers is the two nodes on the edge, and the last one is the weight on the edge.

Output

One line the final result.

Sample Input

```
4 0
9 8 6 5
```

Sample Output

```
2
```

F. Magic Pen 6

In HIT, many people have a magic pen. Lilu0355 has a magic pen, darkgt has a magic pen, discover has a magic pen. Recently, Timer also got a magic pen from seniors.

At the end of this term, teacher gives Timer a job to deliver the list of N students who fail the course to dean's office. Most of these students are Timer's friends, and Timer doesn't want to see them fail the course. So, Timer decides to use his magic pen to scratch out consecutive names as much as possible. However, teacher has already calculated the sum of all students' scores module M. Then in order not to let the teacher find anything strange, Timer should keep the sum of the rest of students' scores module M the same.

Plans can never keep pace with changes, Timer is too busy to do this job. Therefore, he turns to you. He needs you to program to "save" these students as much as possible.

Input

There are multiple test cases.
The first line of each case contains two integer N and M, (0< N <= 100000, 0 < M < 10000),then followed by a line consists of N integers $a_1,a_2,...a_n$ (-100000000 <= $a_1,a_2,...a_n$ <= 100000000) denoting the score of each student.
(Strange score? Yes, in great HIT, everything is possible)

Output

For each test case, output the largest number of students you can scratch out.

Sample Input

```
2 3
1 6
3 3
2 3 6
2 5
1 3
```

Sample Output

```
1
2
0
```

G. Professor Tian

Timer took the Probability and Mathematical Statistics course in the 2012, But his bad attendance angered Professor Tian who is in charge of the course. Therefore, Professor Tian decided to let Timer face a hard probability problem, and announced that if he fail to slove the problem there would be no way for Timer to pass the final exam.
As a result , Timer passed.
Now, you, the bad guy, also angered the Professor Tian when September Ends. You have to faced the problem too.
The problem comes that there is an expression and you should calculate the excepted value of it. And the operators it may contains are '&' (and),'|'(or) and '^'(xor) which are all bit operators. For example: 7&3=3, 5&2=0, 2|5=7, 4|10=14, 6^5=3, 3^4=7.
Professor Tian declares that each operator O_i with its coming number A_{i+1} may disappear, and the probability that it happens is P_i ($0<i<=n$).

Input

The input contains several test cases. For each test case, there is a integer n ($0<n<=200$) in the first line.In the second line, there are n+1 integers, stand for $\{A_i\}$. The next line contains n operators ,stand for $\{O_i\}$. The forth line contains $\{P_i\}$.

A_i will be less than 2^{20}, $0<=P_i<=1$.

Output

For each text case, you should print the number of text case in the first line.Then output the excepted value of the expression, round to 6 decimal places.

Sample Input

```
2
1 2 3
^ ^
0.1 0.2
2
8 9 10
^ ^
0.5 0.78
1
1 2
&
0.5
```

Sample Output

```
Case 1:
0.720000
Case 2:
4.940000
Case 3:
0.500000
```

H. Minimum Average Weight Path

In mathematics, a graph is a representation of a set of objects where some pairs of the objects are connected by links. The interconnected objects are represented by mathematical abstractions called vertices, and the links that connect some pairs of vertices are called edges. A path in a graph is a sequence of vertices, and for any 2 adjacent u, v, there is a edge from u to v in graph. A path contains at least one edge. In the graph in Sample 2, {3, 3, 2, 2} can form a path from 3 to 2.

One of the common problem is to find the shortest path between two certain vertices, or all of them. They've been well studied as the single source shortest path problem (SSSP) and the all pairs shortest paths problem (APSP).

In this problem, we'll provide you a derivation analogous to APSP. You've been given a directed graph with positive or negative edge weights. We define the average weight of a path, as the sum of the edge weights divide the edges number of path. Now you need to find the minimum average weight between all pairs of vertices (APMAWP).

Input

Muiltcases. The first line contains two integer n, m, ($1 \le n \le 10^2$, $1 \le m \le 10^4$) the number of the vertices and the number of the edges.

The next m lines, each line contains three intergers u, v, w, representing a directed edge from u to v with weight w. ($|w| \le 10^3$)

There is no multi-edge. It can contain self-loops.

Output

A n × n matrix representing the APMAWP. The j's element of the i's row represents the minimum average weight of all the paths from vertex i to vertex j. If no such path exists, you need to output "NO" instead (DO NOT output quote please). For each real number, you need to keep exactly 3 digits after digit point.

Sample Input

```
4 4
2 1 2
1 3 -8
2 4 -6
4 3 1
5 8
3 3 735
2 1 946
4 2 276
2 2 -990
3 2 -162
4 4 -18
3 5 783
5 5 -156
```

Sample Output

```
NO NO -8.000 NO
2.000 NO -3.000 -6.000
NO NO NO NO
NO NO 1.000 NO
NO NO NO NO NO
-990.000 -990.000 NO NO NO
-990.000 -990.000 735.000 NO -156.000
-990.000 -990.000 NO -18.000 NO
NO NO NO NO -156.000
```

I. Partition

How many ways can the numbers 1 to 15 be added together to make 15? The technical term for what you are asking is the "number of partition" which is often called P(n). A partition of n is a collection of positive integers (not necessarily distinct) whose sum equals n.

Now, I will give you a number n, and please tell me P(n) mod 1000000007.

Input

The first line contains a number T($1 \le T \le 100$), which is the number of the case number. The next T lines, each line contains a number n($1 \le n \le 10^5$) you need to consider.

Output

For each n, output P(n) in a single line.

Sample Input

```
4
5
11
15
19
```

Sample Output

```
7
56
176
490
```

J. Dice

You have a dice with m faces, each face contains a distinct number. We assume when we tossing the dice, each face will occur randomly and uniformly. Now you have T query to answer, each query has one of the following form:
0 m n: ask for the expected number of tosses until the last n times results are all same.
1 m n: ask for the expected number of tosses until the last n consecutive results are pairwise different.

Input

The first line contains a number T.($1 \le T \le 100$) The next T line each line contains a query as we mentioned above. ($1 \le m,n \le 10^6$) For second kind query, we guarantee $n \le m$. And in order to avoid potential precision issue, we guarantee the result for our query will not exceeding 10^9 in this problem.

Output

For each query, output the corresponding result. The answer will be considered correct if the absolute or relative error doesn't exceed 10^{-6}.

Sample Input

```
6
0 6 1
0 6 3
0 6 5
1 6 2
1 6 4
1 6 6
10
1 4534 25
1 1232 24
1 3213 15
1 4343 24
1 4343 9
1 65467 123
1 43434 100
1 34344 9
1 10001 15
1 1000000 2000
```

Sample Output

```
1.000000000
43.000000000
1555.000000000
2.200000000
7.600000000
83.200000000
25.586315824
26.015990037
15.176341160
24.541045769
9.027721917
127.908330426
103.975455253
9.003495515
15.056204472
4731.706620396
```

K. k-th point

Consider n data points uniformly distributed in a p-dimensional unit ball centered at the origin. What is the expected distance of the k-th farthest point? (Here k start from 0 to n-1, and 0-th is the farthest point, (n-1)-th is the nearest point.)

Input

The first line contains a number T(T≤100), which is the number of the case number. The next T lines, each line contains a number p, n and k(1≤p, $n \leq 10^4$,0≤k≤n-1) you need to consider.

Output

For each p, n and k, output the expected distance in a single line.The answer will be considered correct if the absolute or relative error doesn't exceed 10-4.

Sample Input

```
5
2 1 0
2 2 0
3 2 0
3 2 1
4 10 5
```

Sample Output

```
0.666666667
0.800000000
0.857142857
0.642857143
0.812559023
```

L. k-edge connected components

Efficiently computing k-edge connected components in a large graph G = (V, E), where V is the vertex set and E is the edge set, is a long standing research problem. It is not only fundamental in graph analysis but also crucial in graph search optimization algorithms. Computing k-edge connected components has many real applications. For example, in social networks, computing k-edge connected components can identify the closely related entities to provide useful information for social behavior mining. In a web-link based graph, a highly connected graph may be a group of web pages with a high commonality, which is useful for identifying the similarities among web pages. In computational biology, a highly connected subgraph is likely to be a functional cluster of genes for biologist to conduct the study of gene microarrays. Computing k-edge connected components also potentially contributes to many other technology developments such as graph visualization, robust detection of communication networks, community detection in a social network.

Clearly, if a graph G is not k-edge connected, there must be a set C of edges, namely a cut, such that the number |C| of edges in C is smaller than k and the removal of the edges in C cuts the graph G into two disconnected subgraphs G_1 and G_2. A connected component is a maximal connected subgraph of G. Note that each vertex belongs to exactly one connected component, as does each edge.

Now, we give you a undirected graph G with n vertices and m edges without self-loop or multiple edge, your task is just find out the number of k-edge connected components in G.

Input

Multicases. 3 integer numbers n, m and k are described in the first line of the testcase.($3 \le n \le 100$, $1 \le m \le n \times (n-1)/2$, $2 \le k \le n$)The following m lines each line has 2 numbers u, v describe the edges of graph G.($1 \le u,v \le n, u \ne v$)

Output

A single line containing the answer to the problem.

Sample Input

```
5 6 3
1 3
2 3
1 4
2 4
1 5
2 5
9 11 2
1 2
1 3
2 3
4 5
4 6
5 6
7 8
7 9
8 9
1 4
1 7
16 30 3
1 2
1 3
1 4
2 3
2 4
3 4
5 6
5 7
5 8
6 7
6 8
7 8
9 10
9 11
9 12
10 11
10 12
11 12
```

13 14
13 15
13 16
14 15
14 16
15 16
1 5
2 6
1 9
2 10
1 13
2 14

Sample Output

5
3
4