A: Tree

Time Limit

9000ms

Memory Limit

256MB

There is a tree with n nodes, at which attach a binary $64 * 64$ matrix $M_i(1 \leq i \leq n)$. There are q queries for matrix multiplication on the path from node a to node b modulo 2. To avoid massive input dataset, $M_i(1 \leq i \leq n)$ is attained by the following algorithm:

Input a random $seed$ (unsigned long long)

```
for(int i = 1; i <= n; ++i) {
    for(int p = 1; p <= 64; ++p) {
        seed ^= seed * seed + 15;
        for(int q = 1; q <= 64; ++q) {
            M[i][p][q] = (seed >> (q - 1)) & 1;
        }
    }
}
```

To avoid massive output, you should output

$$(\sum_{i=1}^{64}\sum_{j=1}^{64} M_{ij} * 19^i * 26^j) \mod 19260817$$

Input Format

There are multi datasets. $(\sum n \leq 3000, \sum q \leq 30000)$.

For each dataset:

In the first $n-1$ lines, there are to integers u, v, indicates there is an edge connects node u and node $v.(1 \leq u, v \leq n)$.

In the next line there is an integer $seed(0 \leq seed < 2^{64})$.

In the next q lines, there is to integers a, b, indicates a query on path from node a to node $b.(1 \leq a, b \leq n)$.

Output Format

For each query, output an integer in one line without any additional space.

Sample Input

```
6 5
4 1
3 4
6 4
5 3
2 3
19260817
4 5
5 6
4 4
2 1
2 2
```

Sample Input

```
4855239
2667906
277543
14478924
1173682
```

B: Coin

Time Limit

1000ms

Memory Limit

32MB

Bob has a not even coin, every time he tosses the coin, the probability that the coin's front face up is $\frac{q}{p}(\frac{q}{p} \leq \frac{1}{2})$.

The question is, when Bob tosses the coin k times, what's the probability that the frequency of the coin facing up is even number.

If the answer is $\frac{X}{Y}$, because the answer could be extremely large, you only need to print $(X * Y^{-1}) \mod (10^9 + 7)$.

Input Format

First line an integer T, indicates the number of test cases ($T \leq 100$).

Then Each line has 3 integer $p, q, k (1 \leq p, q, k \leq 10^7)$ indicates the i-th test case.

Output Format

For each test case, print an integer in a single line indicates the answer.

Sample Input

```
2
2 1 1
3 1 2
```

Sample Output

```
500000004
555555560
```

C: Sum

Time Limit

1000ms

Memory Limit

32MB

Define the function $S(x)$ for x is a positive integer. $S(x)$ equals to the sum of all digit of the decimal expression of x. Please find a positive integer k that $S(k * x)\%233 = 0$.

Input Format

First line an integer T, indicates the number of test cases ($T \leq 100$). Then Each line has a single integer $x (1 \leq x \leq 1000000)$ indicates i-th test case.

Output Format

For each test case, print an integer in a single line indicates the answer. The length of the answer should not exceed 2000. If there are more than one answer, output anyone is ok.

Sample Input

```
1
1
```

Sample Output

```
89999999999999999999999999
```

D: Brain-baffling Game

Time Limit

1000ms

Memory Limit

32MB

Player #0 and Player #1 are playing a game, and they move by turns. Given n strings, all of which only consist of the characters 0 and 1. When in Player #i's move (i = 0, 1), he has to select a character i in any of the n strings, and remove it and all the characters on its right side. Anyone who can't move loses and the other one wins.

They find the game fairly easy after playing a while, so they make a tiny change to the original game. Besides the characters 0 and 1, now each string may contain **at most one pair** of parentheses, denoting that the substring between the parentheses is compressed, which occurs actually infinite times. You need to figure out who will be the winner at last, assuming that both players are rational enough.

Input Format

Multiple test cases(about 100 test cases).

For each test case, a integer in the first line denotes $n (n \leq 10)$.

Then n strings, $S_1, S_2, \cdots, S_n$, follow in the next n lines. $|S_i| \leq 50$.

It is guaranteed that the sum of length of strings in all test cases is no more than 10000.

Output Format

For each test case, if Player #i $(i = 0, 1)$ wins regardless of moving first or not, print `"i wins"`, otherwise print `"tie"`.

Sample Input

```
2
10
0
2
0(1)
100000000000000
2
(0)
(1)
3
0(1)0
1(0)
1(0)
3
0(1)1
0(1)1
1(0)
```

Sample Output

```
0 wins
1 wins
tie
tie
tie
```

Hint

Once a player selects a character between the parentheses and removes it and all the characters on its right side, the actual length of the string will decrease from infinity to a finite number, hence all the games will end in finite turns.

Taking the third sample for instance, without loss of generality, let Player #0 moves first. Let's assume that he selects the (k+1)-th character between the parentheses, and then the string will become $0000...0$, whose length is k. Afterwards Player #1 have to remove infinite 1s and make sure the number of the remaining 1s is greater than or equal to k. Then we will see Player #1 must be the winner. According to symmetry of the initial strings, Player #0 will win if Player #1 moves first. Therefore, the game is a tie.

E: Maximum Flow

Time Limit

1000ms

Memory Limit

32MB

Given a directed graph with n nodes, labeled $0, 1, \cdots, n-1$.

For each $< i, j >$ satisfies $0 \leq i < j < n$, there exists an edge from the i-th node to the j-th node, the capacity of which is i xor j.

Find the maximum flow network from the 0-th node to the (n-1)-th node, modulo 1000000007.

Input Format

Multiple test cases (no more than 10000).

In each test case, one integer in a line denotes $n(2 \leq n \leq 10^{18})$.

Output Format

Output the maximum flow modulo 1000000007 for each test case.

Sample Input

```
2
```

Sample Output

```
1
```

F: Trig Function

Time Limit

1000ms

Memory Limit

128MB

$f(cos(x)) = cos(n * x)$ holds for all x.

Given two integers n and m, you need to calculate the coefficient of x^m in $f(x)$, modulo 998244353.

Input Format

Multiple test cases (no more than 100).

Each test case contains one line consisting of two integers n and m.

$1 \leq n \leq 10^9, 0 \leq m \leq 10^4$.

Output Format

Output the answer in a single line for each test case.

Sample Input

```
2 0
2 1
2 2
```

Sample Output

```
998244352
0
2
```

G: Xor

Time Limit

2000ms

Memory Limit

256MB

There is a tree with n nodes. For each node, there is an integer value a_i, ($1 \leq a_i \leq 1,000,000,000$ for $1 \leq i \leq n$). There is q queries which are described as follow: Assume the value on the path from node a to node b is $t_0, t_1, \cdots t_m$. You are supposed to calculate t_0 xor t_k xor t_{2k} xor ... xor t_{pk} $(pk \leq m)$.

Input Format

There are multi datasets. $(\sum n \leq 50,000, \sum q \leq 50,000)$.

For each dataset: In the first $n-1$ lines, there are two integers u, v, indicates there is an edge connect node u and node v.

In the next n lines, There is an integer a_i ($1 \leq a_i \leq 1,000,000,000$).

In the next q lines, There is three integers a, b and k. ($1 \leq a, b, k \leq n$).

Output Format

For each query, output an integer in one line, without any additional space.

Sample Input

```
5 6
1 5
4 1
2 1
3 2
19
26
0
8
17
5 5 1
1 3 2
3 2 1
5 4 2
3 4 4
1 4 5
```

Sample Output

```
17
19
26
25
0
19
```

H: Music

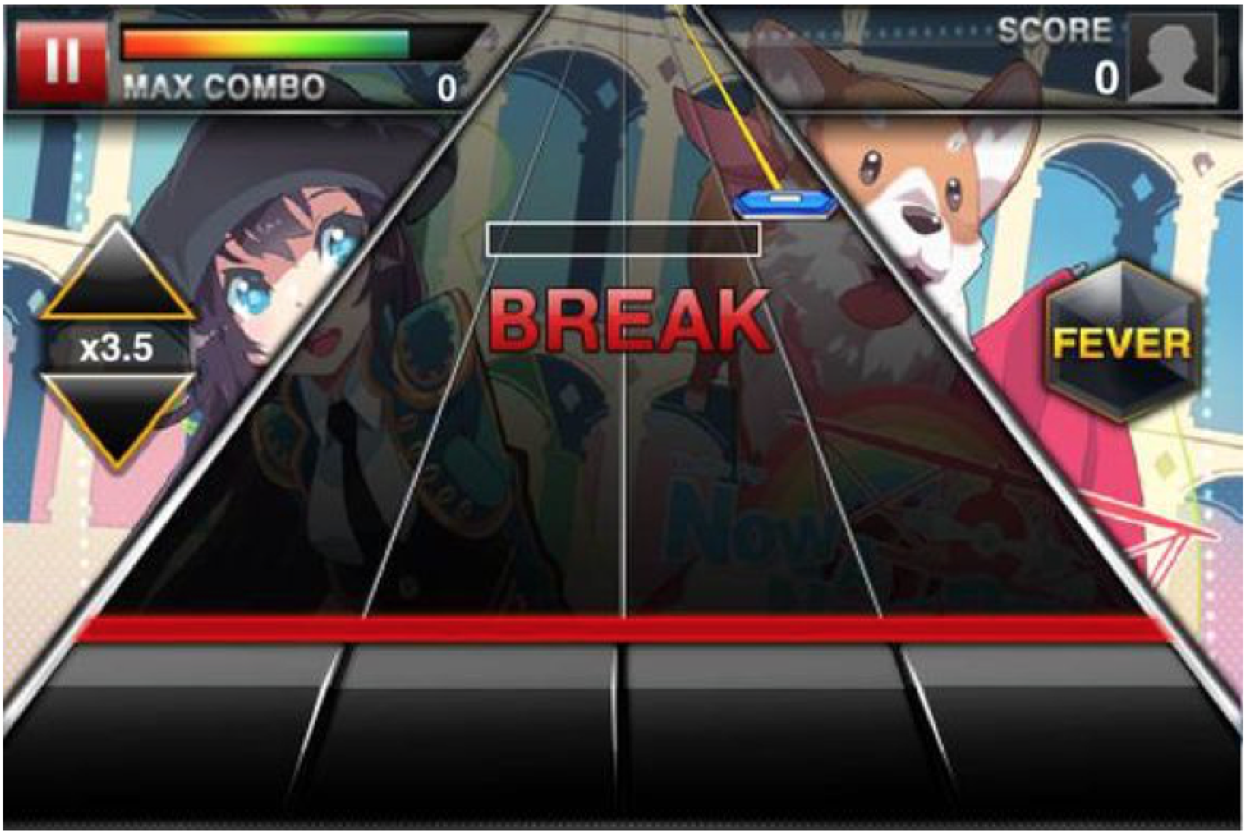
Time Limit

5000ms

Memory Limit

256MB

AI likes playing a game named "Rhythm Master".



He plays this game day after day, it is super noisy. His roommate can not tolerate him anymore, so he decide to write a plug-in to defeat AI.

We can regard this game as in an $1 \times n$ keyboard. Every millisecond, there are some note drop to some position of your keyboard. Notes have two different types.

- `'*'`: The note is a single note, you can press the corresponding button to get 5 points.
- `'#'`: The note is a continuous note, this position will be `'#'` for several milliseconds, only if you press the corresponding button and never get miss between the continuous note, you can get $10 * P$ points, P is the number of `'#'`.

For more, `'.'` means at this time there are no note drop to the position.

The word **miss** means, if you lose any note at millisecond T, you will get miss at T.

For example:

Time 1: `#..*`

Time 2: `#...`

If you press (1001) at the first millisecond(1 means pressing, 0 means no pressing), you will get 5 points, if you press (1000) at the second millisecond, you will get $10 * 2 = 20$ points.

If you press (1000) at the first millisecond, you will get 0 points, if you press (1000) at the second millisecond, you will get 0 points, because you get miss at millisecond 1.

To make this game more interesting, the developer add a new data named combo.

If you don’t get miss at millisecond T, you will get a $combo$, means $combo = combo + 1$;

If you get miss at millisecond T, or the game ends, you will get $\frac{combo^2}{2} + \frac{combo}{2} - 1$ points and then your combo will become to 0.

For example:

Time 1: `#..#`

Time 2: `#..#`

Time 3: `#...`

If you press (1001),(1001),(1001) in first 3 times, you will get $30 + 20 + 3 * 3/2 + 3/2 - 1 = 55$ points.

If you press (1001),(1001),(0000) in first 3 times, you will get $20 + 2 * 2/2 + 2/2 - 1 = 22$ points.

If you press (1001),(0001),(1000) in first 3 times, you will get 0 point.

Now the plug-in is ready for use, but we notice it has some bugs.

If the plug-in press X at millisecond T, It can not press Y at millisecond $T + 1$.

For example:

$X = 1010, Y = 0101.$

If the plug-in press 1010(or 1110,1011,1111) at time T, it can not press 0101 at time $T + 1$, so can’t 1101,0111, and 1111.

Help the plug-in to find the maximum points it can get.

Input Format

Several test cases.

Each case begins with 3 integers $N, M, K (1 \leq N \leq 7, 1 \leq M \leq 5000, 1 \leq K \leq 1000)$. N is the size of keyboard. M is the millisecond of the game. K is the number of bug.

The next K lines, each line contains two `01` string, means X, Y.

Then M lines follow, each line contains a string with length N, the i-th($1 \leq i \leq M$) line means the situation of millisecond i.

Output Format

For each test case, print an integer in a single line.

Sample Input

```
4 4 2
1111 1111
0011 1100
****
#*##
#*##
#..#
```

Sample Output

```
30
```

I: Barty's Computer

Time Limit

4000ms

Memory Limit

512MB

Barty have a computer, it can do these two things.

1. Add a new string to its memory, the length of this string is even.
2. For given 4 strings a, b, c, d, find out how many strings that can be product by $a + s1 + b + c + s2 + d$, and $|a| + |s1| + |b| = |c| + |s2| + |d|$. $|s|$ means the length of string s, $s1$ and $s2$ can be any string, including `""`.

Please help your computer to do these things.

Input Format

Test cases begins with $T(T \leq 5)$.

Then T test cases follows.

Each test case begins with an integer $Q(Q \leq 30000)$.

Then Q lines,

1 s: add a new string s to its memory.

2 a b c d: find how many strings satisfying the requirement above.

$\sum |s| + |a| + |b| + |c| + |d| \leq 2000000.$

Output Format

For type 2 query. Output the answer in one line.

Sample Input

```
1
10
1 abcqaq
1 abcabcqaqqaq
2 ab bc qa aq
2 a c q q
1 abcabcqaqqwq
2 ab bc qa aq
2 a c q q
1 abcq
2 a c q q
2 a b c q
```

Sample Output

```
1
2
1
3
3
1
```

J: Easy Problem

Time Limit

15000ms

Memory Limit

512MB

This problem is very easy. You need to do these Q queries.

Queries have 3 types.

1 n x1 x2 x3 , ... , xn: Add a new sequence to the next line.

2 x l1 r1 y l2 r2: Copy two sub-sequence and expend them to the next line.

3 x k: Output the first k sum of sub-sequence in the x-th line, in increasing order of the sum of each sub-sequence.

Input Format

First line an integer $T(T = 6)$, indicates the number of test cases.

In each case, begin with an integer $Q(Q \leq 100000)$.

Then comes Q lines.

1 n x1 x2 x3, ..., xn: means the first type.

2 x l1 r1 y l2 r2: means the second type. x and y is the line of the two sequences. $[l1, r1]$, $[l2, r2]$ are the two sub-sequences.

3 x k: means we need the first k sum of sub-sequence of the x-th sequence in increasing order.

Sum of n or sum of $k \leq 1000000$.

Each sequence is no longer than 1000000.

We granted that every number you read are non-negative.

Output Format

For every query of type 3, print k number, each number in a single line .

Sample Input

```
1
3
1 5
1 2 3 4 5
3 1 2
3 1 4
```

Sample Output

```
1
2
1
2
3
3
```