Codeforces Round #604 (Div. 1)

A. Beautiful Regional Contest

2 seconds, 256 megabytes

So the Beautiful Regional Contest (BeRC) has come to an end! n students took part in the contest. The final standings are already known: the participant in the i-th place solved p_i problems. Since the participants are primarily sorted by the number of solved problems, then $p_1 \ge p_2 \ge \dots \ge p_n$.

Help the jury distribute the gold, silver and bronze medals. Let their numbers be g, s and b, respectively. Here is a list of requirements from the rules, which all must be satisfied:

- for each of the three types of medals, at least one medal must be awarded (that is, $g > 0$, $s > 0$ and $b > 0$);
- the number of gold medals must be strictly less than the number of silver and the number of bronze (that is, $g < s$ and $g < b$, but there are no requirements between s and b);
- each gold medalist must solve strictly more problems than any awarded with a silver medal;
- each silver medalist must solve strictly more problems than any awarded a bronze medal;
- each bronze medalist must solve strictly more problems than any participant not awarded a medal;
- the total number of medalists $g + s + b$ should not exceed half of all participants (for example, if $n = 21$, then you can award a maximum of 10 participants, and if $n = 26$, then you can award a maximum of 13 participants).

The jury wants to reward with medals the total **maximal** number participants (i.e. to maximize $g + s + b$) so that all of the items listed above are fulfilled. Help the jury find such a way to award medals.

Input

The first line of the input contains an integer t ($1 \le t \le 10000$) — the number of test cases in the input. Then t test cases follow.

The first line of a test case contains an integer n ($1 \le n \le 4 \cdot 10^5$) — the number of BeRC participants. The second line of a test case contains integers $p_1, p_2, \dots, p_n$ ($0 \le p_i \le 10^6$), where p_i is equal to the number of problems solved by the i-th participant from the final standings. The values p_i are sorted in non-increasing order, i.e. $p_1 \ge p_2 \ge \dots \ge p_n$.

The sum of n over all test cases in the input does not exceed $4 \cdot 10^5$.

Output

Print t lines, the j-th line should contain the answer to the j-th test case.

The answer consists of three non-negative integers g, s, b.

- Print $g = s = b = 0$ if there is no way to reward participants with medals so that all requirements from the statement are satisfied at the same time.
- Otherwise, print three positive numbers g, s, b — the possible number of gold, silver and bronze medals, respectively. The sum of $g + s + b$ should be the maximum possible. If there are several answers, print any of them.

input
```
5
12
5 4 4 3 2 2 1 1 1 1 1 1
4
4 3 2 1
1
1000000
20
20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
32
64 64 63 58 58 58 58 58 37 37 37 37 34 34 28 28 28 28 28 28
24 24 19 17 17 17 17 16 16 16 16 11
```

output
```
1 2 3
0 0 0
0 0 0
2 5 3
2 6 6
```

In the first test case, it is possible to reward 1 gold, 2 silver and 3 bronze medals. In this case, the participant solved 5 tasks will be rewarded with the gold medal, participants solved 4 tasks will be rewarded with silver medals, participants solved 2 or 3 tasks will be rewarded with bronze medals. Participants solved exactly 1 task won't be rewarded. It's easy to see, that in this case, all conditions are satisfied and it is possible to reward participants in this way. It is impossible to give more than 6 medals because the number of medals should not exceed half of the number of participants. The answer $1, 3, 2$ is also correct in this test case.

In the second and third test cases, it is impossible to reward medals, because at least one medal of each type should be given, but the number of medals should not exceed half of the number of participants.

B. Beautiful Sequence

1 second, 256 megabytes

An integer sequence is called *beautiful* if the difference between any two consecutive numbers is equal to 1. More formally, a sequence $s_1, s_2, \dots, s_n$ is beautiful if $|s_i - s_{i+1}| = 1$ for all $1 \le i \le n - 1$.

Trans has a numbers 0, b numbers 1, c numbers 2 and d numbers 3. He wants to construct a beautiful sequence using all of these $a + b + c + d$ numbers.

However, it turns out to be a non-trivial task, and Trans was not able to do it. Could you please help Trans?

Input

The only input line contains four non-negative integers a, b, c and d ($0 < a + b + c + d \le 10^5$).

Output

If it is impossible to construct a beautiful sequence satisfying the above constraints, print "NO" (without quotes) in one line.

Otherwise, print "YES" (without quotes) in the first line. Then in the second line print $a + b + c + d$ integers, separated by spaces — a beautiful sequence. There should be a numbers equal to 0, b numbers equal to 1, c numbers equal to 2 and d numbers equal to 3.

If there are multiple answers, you can print any of them.

input
```
2 2 2 1
```

output
```
YES
0 1 0 1 2 3 2
```

input
```
1 2 3 4
```

output
```
NO
```

input
```
2 2 2 3
```

output
```
NO
```

In the first test, it is easy to see, that the sequence is beautiful because the difference between any two consecutive numbers is equal to 1. Also, there are exactly two numbers, equal to 0, 1, 2 and exactly one number, equal to 3.

It can be proved, that it is impossible to construct beautiful sequences in the second and third tests.

C. Beautiful Mirrors with queries

2 seconds, 256 megabytes

Creatnx has n mirrors, numbered from 1 to n. Every day, Creatnx asks exactly one mirror "Am I beautiful?". The i-th mirror will tell Creatnx that he is beautiful with probability $\frac{p_i}{100}$ for all $1 \le i \le n$.

Some mirrors are called checkpoints. Initially, only the 1st mirror is a checkpoint. It remains a checkpoint all the time.

Creatnx asks the mirrors one by one, starting from the 1-st mirror. Every day, if he asks i-th mirror, there are two possibilities:

- The i-th mirror tells Creatnx that he is beautiful. In this case, if $i = n$ Creatnx will stop and become happy, otherwise he will continue asking the $i + 1$-th mirror next day;
- In the other case, Creatnx will feel upset. The next day, Creatnx will start asking **from the checkpoint with a maximal number that is less or equal to i**.

There are some changes occur over time: some mirrors become new checkpoints and some mirrors are no longer checkpoints. You are given q queries, each query is represented by an integer u: If the u-th mirror isn't a checkpoint then we set it as a checkpoint. Otherwise, the u-th mirror is no longer a checkpoint.

After each query, you need to calculate the expected number of days until Creatnx becomes happy.

Each of this numbers should be found by modulo 998244353. Formally, let $M = 998244353$. It can be shown that the answer can be expressed as an irreducible fraction $\frac{p}{q}$, where p and q are integers and $q \not\equiv 0 \pmod{M}$. Output the integer equal to $p \cdot q^{-1} \bmod M$. In other words, output such an integer x that $0 \le x < M$ and $x \cdot q \equiv p \pmod{M}$.

Input

The first line contains two integers n, q $(2 \le n, q \le 2 \cdot 10^5)$ — the number of mirrors and queries.

The second line contains n integers: $p_1, p_2, \ldots, p_n$ $(1 \le p_i \le 100)$.

Each of q following lines contains a single integer u $(2 \le u \le n)$ — next query.

Output

Print q numbers – the answers after each query by modulo 998244353.

input
```
2 2
50 50
2
2
```
output
```
4
6
```

input
```
5 5
10 20 30 40 50
2
3
4
5
3
```
output
```
117
665496274
332748143
831870317
499122211
```

In the first test after the first query, the first and the second mirrors are checkpoints. Creatnx will ask the first mirror until it will say that he is beautiful, after that he will ask the second mirror until it will say that he is beautiful because the second mirror is a checkpoint. After that, he will become happy. Probabilities that the mirrors will say, that he is beautiful are equal to $\frac{1}{2}$. So, the expected number of days, until one mirror will say, that he is beautiful is equal to 2 and the answer will be equal to $4 = 2 + 2$.

D1. Beautiful Bracket Sequence (easy version)

2 seconds, 256 megabytes

This is the easy version of this problem. The only difference is the limit of n - the length of the input string. In this version, $1 \le n \le 2000$. The hard version of this challenge is not offered in the round for the second division.

Let's define a correct bracket sequence and its depth as follow:

- An empty string is a correct bracket sequence with depth 0.
- If "`s`" is a correct bracket sequence with depth d then "`(s)`" is a correct bracket sequence with depth $d + 1$.
- If "`s`" and "`t`" are both correct bracket sequences then their concatenation "`st`" is a correct bracket sequence with depth equal to the maximum depth of s and t.

For a (not necessarily correct) bracket sequence s, we define its depth as the maximum depth of any **correct** bracket sequence induced by removing some characters from s (possibly zero). For example: the bracket sequence $s =$"`())(())`" has depth 2, because by removing the third character we obtain a correct bracket sequence "`()(())`" with depth 2.

Given a string a consists of only characters '`(`', '`)`' and '`?`'. Consider all (not necessarily correct) bracket sequences obtained by replacing all characters '`?`' in a by either '`(`' or '`)`'. Calculate the sum of all the depths of all these bracket sequences. As this number can be large, find it modulo 998244353.

Hacks in this problem in the first division can be done only if easy and hard versions of this problem was solved.

Input

The only line contains a non-empty string consist of only '`(`', '`)`' and '`?`'. The length of the string is at most 2000.

Output

Print the answer modulo 998244353 in a single line.

input
```
??
```
output
```
1
```

input
```
(?(?))
```
output
```
9
```

In the first test case, we can obtain 4 bracket sequences by replacing all characters '`?`' with either '`(`' or '`)`':

- "`((`". Its depth is 0;
- "`))`". Its depth is 0;
- "`)(`". Its depth is 0;
- "`()`". Its depth is 1.

So, the answer is $1 = 0 + 0 + 0 + 1$.

In the second test case, we can obtain 4 bracket sequences by replacing all characters '`?`' with either '`(`' or '`)`':

- "`((((())`". Its depth is 2;

- "`()()))`". Its depth is 2;
- "`((()))`". Its depth is 3;
- "`()(())`". Its depth is 2.

So, the answer is $9 = 2 + 2 + 3 + 2$.

D2. Beautiful Bracket Sequence (hard version)

2 seconds, 256 megabytes

This is the hard version of this problem. The only difference is the limit of n - the length of the input string. In this version, $1 \leq n \leq 10^6$.

Let's define a correct bracket sequence and its depth as follow:

- An empty string is a correct bracket sequence with depth 0.
- If "`s`" is a correct bracket sequence with depth d then "`(s)`" is a correct bracket sequence with depth $d + 1$.
- If "`s`" and "`t`" are both correct bracket sequences then their concatenation "`st`" is a correct bracket sequence with depth equal to the maximum depth of s and t.

For a (not necessarily correct) bracket sequence s, we define its depth as the maximum depth of any **correct** bracket sequence induced by removing some characters from s (possibly zero). For example: the bracket sequence $s =$"`())(())`" has depth 2, because by removing the third character we obtain a correct bracket sequence "`()(())`" with depth 2.

Given a string a consists of only characters '`(`', '`)`' and '`?`'. Consider all (not necessarily correct) bracket sequences obtained by replacing all characters '`?`' in a by either '`(`' or '`)`'. Calculate the sum of all the depths of all these bracket sequences. As this number can be large, find it modulo 998244353.

Hacks in this problem can be done only if easy and hard versions of this problem was solved.

Input

The only line contains a non-empty string consist of only '`(`', '`)`' and '`?`'. The length of the string is at most 10^6.

Output

Print the answer modulo 998244353 in a single line.

input
```
??
```
output
```
1
```

input
```
(?(?))
```
output
```
9
```

In the first test case, we can obtain 4 bracket sequences by replacing all characters '`?`' with either '`(`' or '`)`':

- "`((`". Its depth is 0;
- "`))`". Its depth is 0;
- "`)(`". Its depth is 0;
- "`()`". Its depth is 1.

So, the answer is $1 = 0 + 0 + 0 + 1$.

In the second test case, we can obtain 4 bracket sequences by replacing all characters '`?`' with either '`(`' or '`)`':

- "`(((())`". Its depth is 2;
- "`()()))`". Its depth is 2;
- "`((()))`". Its depth is 3;
- "`()(())`". Its depth is 2.

So, the answer is $9 = 2 + 2 + 3 + 2$.

E. Beautiful League

1 second, 256 megabytes

A football league has recently begun in Beautiful land. There are n teams participating in the league. Let's enumerate them with integers from 1 to n.

There will be played exactly $\frac{n(n-1)}{2}$ matches: each team will play against all other teams exactly once. In each match, there is always a winner and loser and there is no draw.

After all matches are played, the organizers will count the number of *beautiful triples*. Let's call a triple of three teams (A, B, C) beautiful if a team A win against a team B, a team B win against a team C and a team C win against a team A. We look only to a triples of different teams and the order of teams in the triple is important.

The beauty of the league is the number of beautiful triples.

At the moment, m matches were played and their results are known.

What is the maximum beauty of the league that can be, after playing all remaining matches? Also find a possible results for all remaining $\frac{n(n-1)}{2} - m$ matches, so that the league has this maximum beauty.

Input

The first line contains two integers n, m ($3 \leq n \leq 50, 0 \leq m \leq \frac{n(n-1)}{2}$) — the number of teams in the football league and the number of matches that were played.

Each of m following lines contains two integers u and v ($1 \leq u, v \leq n$, $u \neq v$) denoting that the u-th team won against the v-th team. It is guaranteed that each unordered pair of teams appears at most once.

Output

Print n lines, each line having a string of exactly n characters. Each character must be either 0 or 1.

Let a_{ij} be the j-th number in the i-th line. For all $1 \leq i \leq n$ it should be true, that $a_{ii} = 0$. For all pairs of teams $i \neq j$ the number a_{ij} indicates the result of the match between the i-th team and the j-th team:

- If a_{ij} is 1, the i-th team wins against the j-th team;
- Otherwise the j-th team wins against the i-th team;
- Also, it should be true, that $a_{ij} + a_{ji} = 1$.

Also note that the results of the m matches that were already played cannot be changed in your league.

The beauty of the league in the output should be maximum possible. If there are multiple possible answers with maximum beauty, you can print any of them.

input
```
3 1
1 2
```
output
```
010
001
100
```

input
```
4 2
1 2
1 3
```
output
```
0110
0001
0100
1010
```

The beauty of league in the first test case is equal to 3 because there exists three beautiful triples: $(1, 2, 3)$, $(2, 3, 1)$, $(3, 1, 2)$.

The beauty of league in the second test is equal to 6 because there exists six beautiful triples: $(1, 2, 4)$, $(2, 4, 1)$, $(4, 1, 2)$, $(2, 4, 3)$, $(4, 3, 2)$, $(3, 2, 4)$.

F. Beautiful Fibonacci Problem

1 second, 256 megabytes

The well-known Fibonacci sequence $F_0, F_1, F_2, \ldots$ is defined as follows:

- $F_0 = 0, F_1 = 1$.
- For each $i \geq 2$: $F_i = F_{i-1} + F_{i-2}$.

Given an increasing arithmetic sequence of positive integers with n elements: $(a, a + d, a + 2 \cdot d, \ldots, a + (n - 1) \cdot d)$.

You need to find another increasing arithmetic sequence of positive integers with n elements $(b, b + e, b + 2 \cdot e, \ldots, b + (n - 1) \cdot e)$ such that:

- $0 < b, e < 2^{64}$,
- for all $0 \leq i < n$, the decimal representation of $a + i \cdot d$ appears as substring in the last 18 digits of the decimal representation of $F_{b+i \cdot e}$ (if this number has less than 18 digits, then we consider all its digits).

Input

The first line contains three positive integers n, a, d ($1 \leq n, a, d, a + (n - 1) \cdot d < 10^6$).

Output

If no such arithmetic sequence exists, print -1.

Otherwise, print two integers b and e, separated by space in a single line ($0 < b, e < 2^{64}$).

If there are many answers, you can output any of them.

input
```
3 1 1
```
output
```
2 1
```

input
```
5 1 2
```
output
```
19 5
```

In the first test case, we can choose $(b, e) = (2, 1)$, because $F_2 = 1, F_3 = 2, F_4 = 3$.

In the second test case, we can choose $(b, e) = (19, 5)$ because:

- $F_{19} = 4181$ contains 1;
- $F_{24} = 46368$ contains 3;
- $F_{29} = 514229$ contains 5;
- $F_{34} = 5702887$ contains 7;
- $F_{39} = 63245986$ contains 9.