CodeForces Problem

May 25, 2023

A. Grasshopper on a Line

Constriants

Time Limit 2 seconds **Memory Limit** 256 MB

Problem Statement

You are given two integers x and k. Grasshopper starts in a point 0 on an OX axis. In one move, it can jump some integer distance, that is not divisible by k, to the left or to the right.

What's the smallest number of moves it takes the grasshopper to reach point x? What are these moves? If there are multiple answers, print any of them.

Input Description

The first line contains a single integer t ($1 \leq t \leq 1000$) — the number of testcases.

The only line of each testcase contains two integers x and k ($1 \leq x \leq 100$; $2 \leq k \leq 100$) — the endpoint and the constraint on the jumps, respectively.

Output Description :

For each testcase, in the first line, print a single integer n — the smallest number of moves it takes the grasshopper to reach point x.

In the second line, print n integers, each of them not divisible by k. A positive integer would mean jumping to the right, a negative integer would mean jumping to the left. The endpoint after the jumps should be exactly x.

Each jump distance should be from -10^9 to 10^9. In can be shown that, for any solution with the smallest number of jumps, there exists a solution with the same number of jumps such that each jump is from -10^9 to 10^9.

It can be shown that the answer always exists under the given constraints. If there are multiple answers, print any of them.

Examples

Input
```
3
10 2
10 3
3 4
```
Output
```
2
7 3
1
10
1
3
```

B. Comparison String

Constriants

Time Limit 2 seconds **Memory Limit** 512 MB

Problem Statement

You are given a string s of length n, where each character is either $<;$ or $>;$.

An array a consisting of $n+1$ elements is compatible with the string s if, for every i from 1 to n, the character s_i represents the result of comparing a_i and a_{i+1}, i. e.: s_i is $<;$ if and only if $a_i <; a_{i+1}$; s_i is $>;$ if and only if $a_i >; a_{i+1}$.

For example, the array $[1, 2, 5, 4, 2]$ is compatible with the string $<;<;>;>;$. There are other arrays with are compatible with that string, for example, $[13, 37, 42, 37, 13]$.

The cost of the array is the number of different elements in it. For example, the cost of $[1, 2, 5, 4, 2]$ if 4; the cost of $[13, 37, 42, 37, 13]$ is 3.

You have to calculate the minimum cost among all arrays which are compatible with the given string s.

Input Description

The first line contains one integer t ($1 \le t \le 500$) — the number of test cases.

Each test case consists of two lines: the first line contains one integer n ($1 \le n \le 100$); the second line contains the string s, consisting of n characters. Each character of s is either $<;$ or $>;$.

Output Description :

For each test case, print one integer — the minimum cost among all arrays which are compatible with the given string s.

Examples

Input

```
4
4
<;<;>;>;
4
>;>;<;<;
5
>;>;>;>;>;
7
<;>;<;>;<;>;<;
```

Output

```
2
2
5
1
```

Note

In the first test case of the example, the array can be $[13, 37, 42, 37, 13]$.
In the second test case of the example, the array can be $[42, 37, 13, 37, 42]$.

C. Best Binary String

Constriants

Time Limit 2 seconds **Memory Limit** 256 MB

Problem Statement

You are given a string s consisting of the characters 0, 1 and/or ?. Let's call it a pattern.

Let's say that the binary string (a string where each character is either 0 or 1) matches the pattern if you can replace each character ? with 0 or 1 (for each character, the choice is independent) so that the strings become equal. For example, 0010 matches ?01?, but 010 doesn't match 1??, ??, or ????.

Let's define the cost of the binary string as the minimum number of operations of the form "reverse an arbitrary contiguous substring of the string" required to sort the string in non-descending order.

You have to find a binary string with the minimum possible cost among those that match the given pattern. If there are multiple answers, print any of them.

Input Description

The first line contains a single integer t ($1 \le t \le 3 \cdot 10^4$) — the number of test cases.

The first and only line of each test case contains the string s ($1 \le |s| \le 3 \cdot 10^5$) consisting of characters 0, 1, and/or ?.

The sum of the string lengths over all test cases does not exceed $3 \cdot 10^5$.

Output Description :

For each test case, print a binary string with the minimum possible cost among those that match the given pattern. If there are multiple answers, print any of them.

Examples

Input
```
4
??01?
10100
1??10?
0?1?10?10
```
Output
```
00011
10100
111101
011110010
```

Note

In the first test case of the example, the cost of the resulting string is 0.

In the second test case, the cost of the resulting string is 2: we can reverse the substring from the 1-st character to the 5-th character, and we obtain the string 00101. Then we reverse the substring from the 3-rd to the 4-th character, and we obtain the string 00011, which is sorted in non-descending order.

D. Bracket Coloring

Constriants

Time Limit 2 seconds **Memory Limit** 512 MB

Problem Statement

A regular bracket sequence is a bracket sequence that can be transformed into a correct arithmetic expression by inserting characters "1" and "+" between the original characters of the sequence. For example: the bracket sequences "()()" and "(())" are regular (the resulting expressions are: "(1)+(1)" and "((1+1)+1)"); the bracket sequences ")(", "(" and ")" are not.

A bracket sequence is called beautiful if one of the following conditions is satisfied: it is a regular bracket sequence; if the order of the characters in this sequence is reversed, it becomes a regular bracket sequence.

For example, the bracket sequences "()()", "(())", ")))(((", "))()((" are beautiful.

You are given a bracket sequence s. You have to color it in such a way that: every bracket is colored into one color; for every color, there is at least one bracket colored into that color; for every color, if you write down the sequence of brackets having that color in the order they appear, you will get a beautiful bracket sequence.

Color the given bracket sequence s into the minimum number of colors according to these constraints, or report that it is impossible.

Input Description

The first line contains one integer t ($1 \le t \le 10^4$) — the number of test cases.

Each test case consists of two lines. The first line contains one integer n ($2 \le n \le 2 \cdot 10^5$) — the number of characters in s. The second line contains s — a string of n characters, where each character is either "(" or ")".

Additional constraint on the input: the sum of n over all test cases does not exceed $2 \cdot 10^5$.

Output Description :

For each test case, print the answer as follows: if it is impossible to color the brackets according to the problem statement, print -1; otherwise, print two lines. In the first line, print one integer k ($1 \le k \le n$) — the minimum number of colors. In the second line, print n integers $c_1, c_2, \dots, c_n$ ($1 \le c_i \le k$), where c_i is the color of the i-th bracket. If there are multiple answers, print any of them.

Examples

Input
4
8
((()))(
4
(())
4
))((
3
(()
Output
2
2 2 2 1 2 2 2 1
1
1 1 1 1
1
1 1 1 1
-1

E. Playoff Fixing

Constriants

Time Limit 2 seconds **Memory Limit** 256 MB

Problem Statement

2^k teams participate in a playoff tournament. The teams are numbered from 1 to 2^k, in order of decreasing strength. So, team 1 is the strongest one, team 2^k is the weakest one. A team with a smaller number always defeats a team with a larger number.

First of all, the teams are arranged in some order during a procedure called seeding. Each team is assigned another unique value from 1 to 2^k, called a seed, that represents its starting position in the playoff.

The tournament consists of $2^k - 1$ games. They are held as follows: the teams are split into pairs: team with seed 1 plays against team with seed 2, team with seed 3 plays against team with seed 4 (exactly in this order), and so on (so, 2^{k-1} games are played in that phase). When a team loses a game, it is eliminated.

After that, only 2^{k-1} teams remain. If only one team remains, it is declared the champion; otherwise, 2^{k-2} games are played: in the first one of them, the winner of the game "seed 1 vs seed 2" plays against the winner of the game "seed 3 vs seed 4", then the winner of the game "seed 5 vs seed 6" plays against the winner of the game "seed 7 vs seed 8", and so on. This process repeats until only one team remains.

After the tournament ends, the teams are assigned places according to the tournament phase when they were eliminated. In particular: the winner of the tournament gets place 1; the team eliminated in the finals gets place 2; both teams eliminated in the semifinals get place 3; all teams eliminated in the quarterfinals get place 5; all teams eliminated in the 1/8 finals get place 9, and so on.

Now that we established the rules, we do a little rigging. In particular, we want: team 1 (not team with seed 1) to take place 1; team 2 to take place 2; teams 3 and 4 to take place 3; teams from 5 to 8 to take place 5, and so on.

For example, this picture describes one of the possible ways the tournament can go with $k = 3$, and the resulting places of the teams:

Some seeds are already reserved for some teams (we are not the only ones rigging the tournament, apparently). We have to fill the rest of the seeds with the remaining teams to achieve the desired placements. How many ways are there to do that? Since that value might be large, print it modulo 998 244 353.

Input Description

The first line contains a single integer k ($0 \le k \le 19$) — there are 2^k teams.

The second line contains 2^k integers $a_1, a_2, \ldots, a_{2^k}$ ($a_i = -1$ or $1 \le a_i \le 2^k$). If $a_i \ne -1$, then team a_i has seed i. Otherwise, the seed i is not reserved for any team.

All values, that are not -1, are distinct.

Output Description :

Print a single integer — the number of ways to fill the non-reserved seeds so that the tournament goes as planned, modulo 998 244 353.

Examples

Input 2 1 2 3 4 **Output** 0
Input 2 1 3 4 2 **Output** 1
Input 1 -1 -1 **Output** 2
Input 2 -1 -1 -1 -1 **Output** 16
Input 3 -1 -1 -1 -1 2 -1 -1 -1 **Output** 768
Input 0 1 **Output** 1

F. Editorial for Two

Constriants

Time Limit 4 seconds **Memory Limit** 256 MB

Problem Statement

Berland Intercollegiate Contest has just finished. Monocarp and Polycarp, as the jury, are going to conduct an editorial. Unfortunately, the time is limited, since they have to finish before the closing ceremony.

There were n problems in the contest. The problems are numbered from 1 to n. The editorial for the i-th problem takes a_i minutes. Monocarp and Polycarp are going to conduct an editorial for exactly k of the problems.

The editorial goes as follows. They have a full problemset of n problems before them, in order. They remove $n - k$ problems without changing the order of the remaining k problems. Then, Monocarp takes some prefix of these k problems (possibly, an empty one or all problems). Polycarp takes the remaining suffix of them. After that, they go to different rooms and conduct editorials for their problems in parallel. So, the editorial takes as much time as the longer of these two does.

Please, help Monocarp and Polycarp to choose the problems and the split in such a way that the editorial finishes as early as possible. Print the duration of the editorial.

Input Description

The first line contains a single integer t ($1 \le t \le 10^4$) — the number of testcases.

The first line of each testcase contains two integers n and k ($1 \le k \le n \le 3 \cdot 10^5$) — the number of problems in the full problemset and the number of problems Monocarp and Polycarp are going to conduct an editorial for.

The second line contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^9$) — the time each editorial takes.

Output Description :

For each testcase, print a single integer — the smallest amount of time the editorial takes, if Monocarp and Polycarp can choose which k of n problems to conduct an editorial for and how to split them among themselves.

Examples

```
Input
6
5 4
1 10 1 1 1
5 3
1 20 5 15 3
5 3
1 20 3 15 5
10 6
10 8 20 14 3 8 6 4 16 11
10 5
9 9 2 13 15 19 4 9 13 12
1 1
1
Output
2
6
5
21
18
1
```