Educational Codeforces Round 104 (Rated for Div. 2)

A. Arena

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

n heroes fight against each other in the Arena. Initially, the i-th hero has level a_i.

Each minute, a fight between two different heroes occurs. These heroes can be chosen arbitrarily (**it's even possible that it is the same two heroes that were fighting during the last minute**).

When two heroes of equal levels fight, nobody wins the fight. When two heroes of different levels fight, the one with the higher level wins, and his level increases by 1.

The winner of the tournament is the first hero that wins in at least 100^{500} fights **(note that it's possible that the tournament lasts forever if no hero wins this number of fights, then there is no winner)**. A *possible winner* is a hero such that there exists a sequence of fights that this hero becomes the winner of the tournament.

Calculate the number of *possible winners* among n heroes.

Input
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 ($2 \le n \le 100$) — the number of heroes. The second line contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_i \le 100$), where a_i is the initial level of the i-th hero.

Output
For each test case, print one integer — the number of *possible winners* among the given n heroes.

Example

input
3 3 3 2 2 2 5 5 4 1 3 3 7
output
1 0 3

Note
In the first test case of the example, the only *possible winner* is the first hero.

In the second test case of the example, each fight between the heroes results in nobody winning it, so the tournament lasts forever and there is no winner.

B. Cat Cycle

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

Suppose you are living with two cats: A and B. There are n napping spots where both cats usually sleep.

Your cats like to sleep and also like all these spots, so they change napping spot each hour cyclically:

- Cat A changes its napping place in order: $n, n-1, n-2, \ldots, 3, 2, 1, n, n-1, \ldots$ In other words, at the first hour it's on the spot n and then goes in decreasing order cyclically;
- Cat B changes its napping place in order: $1, 2, 3, \ldots, n-1, n, 1, 2, \ldots$ In other words, at the first hour it's on the spot 1 and then goes in increasing order cyclically.

The cat B is much younger, so they have a strict hierarchy: A and B don't lie together. In other words, if both cats'd like to go in spot x then the A takes this place and B moves to the next place in its order (if $x < n$ then to $x+1$, but if $x = n$ then to 1). Cat B follows his order, so **it won't return to the skipped spot x after A frees it, but will move to the spot $x+2$ and so on**.

Calculate, where cat B will be at hour k?

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

The first and only line of each test case contains two integers n and k ($2 \le n \le 10^9$; $1 \le k \le 10^9$) — the number of spots and hour k.

Output
For each test case, print one integer — the index of the spot where cat B will sleep at hour k.

Example

input
7 2 1 2 2 3 1 3 2 3 3 5 5 69 1337
output
1 2 1 3 2 2 65

Note
In the first and second test cases $n = 2$, so:

- at the 1-st hour, A is on spot 2 and B is on 1;
- at the 2-nd hour, A moves to spot 1 and B — to 2.

If $n = 3$ then:

- at the 1-st hour, A is on spot 3 and B is on 1;
- at the 2-nd hour, A moves to spot 2; B'd like to move from 1 to 2, but this spot is occupied, so it moves to 3;
- at the 3-rd hour, A moves to spot 1; B also would like to move from 3 to 1, but this spot is occupied, so it moves to 2.

In the sixth test case:

- A's spots at each hour are $[5, 4, 3, 2, 1]$;
- B's spots at each hour are $[1, 2, 4, 5, 2]$.

C. Minimum Ties

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

A big football championship will occur soon! n teams will compete in it, and each pair of teams will play exactly one game against each other.

There are two possible outcomes of a game:

- the game may result in a tie, then both teams get 1 point;
- one team might win in a game, then the winning team gets 3 points and the losing team gets 0 points.

The score of a team is the number of points it gained during all games that it played.

You are interested in a hypothetical situation when all teams get the same score at the end of the championship. A simple example of that situation is when all games result in ties, but you want to minimize the number of ties as well.

Your task is to describe a situation (choose the result of each game) so that all teams get the same score, and the number of ties is the minimum possible.

Input
The first line contains one integer t ($1 \le t \le 100$) — the number of test cases.

Then the test cases follow. Each test case is described by one line containing one integer n ($2 \le n \le 100$) — the number of teams.

Output
For each test case, print $\frac{n(n-1)}{2}$ integers describing the results of the games in the following order: the first integer should correspond to the match between team 1 and team 2, the second — between team 1 and team 3, then 1 and 4, ..., 1 and n, 2 and 3

, 2 and 4, ..., 2 and n, and so on, until the game between the team $n-1$ and the team n.

The integer corresponding to the game between the team x and the team y should be 1 if x wins, -1 if y wins, or 0 if the game results in a tie.

All teams should get the same score, and the number of ties should be the minimum possible. If there are multiple optimal answers, print any of them. It can be shown that there always exists a way to make all teams have the same score.

Example

input
2 2 3
output
0 1 -1 1

Note

In the first test case of the example, both teams get 1 point since the game between them is a tie.

In the second test case of the example, team 1 defeats team 2 (team 1 gets 3 points), team 1 loses to team 3 (team 3 gets 3 points), and team 2 wins against team 3 (team 2 gets 3 points).

D. Pythagorean Triples

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

A Pythagorean triple is a triple of integer numbers (a, b, c) such that it is possible to form a right triangle with the lengths of the first cathetus, the second cathetus and the hypotenuse equal to a, b and c, respectively. An example of the Pythagorean triple is $(3, 4, 5)$.

Vasya studies the properties of right triangles, and he uses a formula that determines if some triple of integers is Pythagorean. Unfortunately, he has forgotten the exact formula; he remembers only that the formula was some equation with squares. So, he came up with the following formula: $c = a^2 - b$.

Obviously, this is not the right formula to check if a triple of numbers is Pythagorean. But, to Vasya's surprise, it actually worked on the triple $(3, 4, 5)$: $5 = 3^2 - 4$, so, according to Vasya's formula, it is a Pythagorean triple.

When Vasya found the right formula (and understood that his formula is wrong), he wondered: how many are there triples of integers (a, b, c) with $1 \le a \le b \le c \le n$ such that they are Pythagorean both according to his formula and the real definition? He asked you to count these triples.

Input

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

Each test case consists of one line containing one integer n ($1 \le n \le 10^9$).

Output

For each test case, print one integer — the number of triples of integers (a, b, c) with $1 \le a \le b \le c \le n$ such that they are Pythagorean according both to the real definition and to the formula Vasya came up with.

Example

input
3 3 6 9
output
0 1 1

Note

The only Pythagorean triple satisfying $c = a^2 - b$ with $1 \le a \le b \le c \le 9$ is $(3, 4, 5)$; that's why the answer for $n = 3$ is 0, and the answer for $n = 6$ (and for $n = 9$) is 1.

E. Cheap Dinner

time limit per test: 4 seconds
memory limit per test: 512 megabytes
input: standard input

output: standard output

Ivan wants to have a good dinner. A good dinner should consist of a first course, a second course, a drink, and a dessert.

There are n_1 different types of first courses Ivan can buy (the i-th of them costs a_i coins), n_2 different types of second courses (the i-th of them costs b_i coins), n_3 different types of drinks (the i-th of them costs c_i coins) and n_4 different types of desserts (the i-th of them costs d_i coins).

Some dishes don't go well with each other. There are m_1 pairs of first courses and second courses that don't go well with each other, m_2 pairs of second courses and drinks, and m_3 pairs of drinks and desserts that don't go well with each other.

Ivan wants to buy exactly one first course, one second course, one drink, and one dessert so that they go well with each other, and the total cost of the dinner is the minimum possible. Help him to find the cheapest dinner option!

Input

The first line contains four integers n_1, n_2, n_3 and n_4 ($1 \le n_i \le 150000$) — the number of types of first courses, second courses, drinks and desserts, respectively.

Then four lines follow. The first line contains n_1 integers $a_1, a_2, \ldots, a_{n_1}$ ($1 \le a_i \le 10^8$), where a_i is the cost of the i-th type of first course. Three next lines denote the costs of second courses, drinks, and desserts in the same way ($1 \le b_i, c_i, d_i \le 10^8$).

The next line contains one integer m_1 ($0 \le m_1 \le 200000$) — the number of pairs of first and second courses that don't go well with each other. Each of the next m_1 lines contains two integers x_i and y_i ($1 \le x_i \le n_1$; $1 \le y_i \le n_2$) denoting that the first course number x_i doesn't go well with the second course number y_i. All these pairs are different.

The block of pairs of second dishes and drinks that don't go well with each other is given in the same format. The same for pairs of drinks and desserts that don't go well with each other ($0 \le m_2, m_3 \le 200000$).

Output

If it's impossible to choose a first course, a second course, a drink, and a dessert so that they go well with each other, print -1. Otherwise, print one integer — the minimum total cost of the dinner.

Examples

input
4 3 2 1 1 2 3 4 5 6 7 8 9 10 2 1 2 1 1 2 3 1 3 2 1 1 1
output
26

input
1 1 1 1 1 1 1 1 1 1 1 0 0
output
-1

Note

The best option in the first example is to take the first course 2, the second course 1, the drink 2 and the dessert 1.

In the second example, the only pair of the first course and the second course is bad, so it's impossible to have dinner.

F. Ones

time limit per test: 3 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

You are given a positive (greater than zero) integer n.

You have to represent n as the sum of integers (possibly negative) consisting only of ones (digits '1'). For example,

$24 = 11 + 11 + 1 + 1$ and $102 = 111 - 11 + 1 + 1$.

Among all possible representations, you have to find the one that uses the minimum number of ones in total.

Input
The single line contains one integer n ($1 \le n < 10^{50}$).

Output
Print one integer x — the minimum number of ones, such that there exist a representation of n as the sum of integers (possibly negative) that uses x ones in total.

Examples

input
24
output
6

input
102
output
7

G. String Counting

time limit per test: 10 seconds
memory limit per test: 1024 megabytes
input: standard input
output: standard output

You have c_1 letters 'a', c_2 letters 'b', ..., c_{26} letters 'z'. You want to build a *beautiful* string of length n from them (obviously, you cannot use the i-th letter more than c_i times). **Each c_i is greater than $\frac{n}{3}$.**

A string is called *beautiful* if there are no palindromic contiguous substrings of odd length greater than 1 in it. For example, the string "abacaba" is not beautiful, it has several palindromic substrings of odd length greater than 1 (for example, "aca"). Another example: the string "abcaa" is *beautiful*.

Calculate the number of different strings you can build, and print the answer modulo 998244353.

Input
The first line contains one integer n ($3 \le n \le 400$).

The second line contains 26 integers c_1, c_2, ..., c_{26} ($\frac{n}{3} < c_i \le n$).

Output
Print one integer — the number of strings you can build, taken modulo 998244353.

Examples

input
4 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
output
422500

input
3 2 2 2 2 2 2 3 3 3 2 2 2 2 2 2 3 3 3 2 2 3 2 2 3 2 2
output
16900

input
400 348 322 247 158 209 134 151 267 268 176 214 379 372 291 388 135 147 304 169 149 193 351 380 368 181 340
output
287489790