A. Treasure Hunt

```
Input file:     standard input
Output file:    standard output
Time limit:     1 second
Memory limit:   256 megabytes
```

Little B and his friend Little K found a treasure map, and now they just need to dig up the treasure, which is buried at a depth of $a.5$ meters.

They take turns digging. On the first day, Little B digs; on the second day, Little K. After each day, they switch. Little B digs exactly x meters of soil each day, while Little K digs y meters. They became curious about who will dig up the treasure first, meaning during whose day the total dug depth will exceed $a.5$.

But they are too busy digging, so help them and tell who will dig up the treasure!

Input

The first line contains an integer t ($1 \leq t \leq 1000$) — the number of test cases. The description of the test cases follows.

In a single line of each test case, three integers x, y, a are given ($1 \leq x, y, a \leq 10^9$).

Output

For each test case, output "NO" if Little B digs it up first; otherwise, output "YES". You can output the answer in any case.

Standard Input	Standard Output
3 1 2 4 2 1 4 2 2 1	YES NO NO

Note

In the first test case, on the first day they will dig 1 meter; on the second day $1 + 2 = 3$ meters in total; on the third day $1 + 2 + 1 = 4$ meters; and on the fourth day they will dig 6 meters. Thus, the treasure will be dug up first by Little K.

In the second test case, on the first day they will dig 2 meters; on the second day $2 + 1 = 3$ meters in total; on the third day $2 + 1 + 2 = 5$ meters, meaning it was dug up first by Little B.

,

B. Pushing Balls

Input file: standard input
Output file: standard output
Time limit: 1 second
Memory limit: 512 megabytes

Ecrade has an $n \times m$ grid, originally empty, and he has pushed several (possibly, zero) balls in it.

Each time, he can push one ball into the grid either from the leftmost edge of a particular row or the topmost edge of a particular column of the grid.

When a ball moves towards a position:

- If there is no ball originally at that position, the incoming ball will stop and occupy the position.
- If there is already a ball at that position, the incoming ball will stop and occupy the position, while the original ball will continue moving to the next position in the same direction.

Note that if a row or column is full (i.e., all positions in that row or column have balls), he cannot push a ball into that row or column.

Given the final state of whether there is a ball at each position of the grid, you need to determine whether it is possible for Ecrade to push the balls to reach the final state.

Input

The first line contains an integer t ($1 \leq t \leq 10\,000$) — the number of test cases. The description of the test cases follows.

The first line of each test case contains two integers n and m ($1 \leq n, m \leq 50$).

This is followed by n lines, each containing exactly m characters and consisting only of 0 and 1, describing the final state of the grid. There is a ball at one position of the grid if and only if the corresponding position of the given input is 1.

It is guaranteed that the sum of $n \cdot m$ over all test cases does not exceed $10\,000$.

Output

For each test case, output "Yes" (without quotes) if it is possible for Ecrade to push the balls to reach the final state, and "No" (without quotes) otherwise.

You can output "Yes" and "No" in any case (for example, strings "YES", "yEs" and "yes" will be recognized as a positive response).

Standard Input	Standard Output
5 3 3 001 001 110 3 3 010 111 010 3 3 111 111 111 3 3 000 000 000 3 3 000 000 001	YES YES YES YES NO

Note

For simplicity, if Ecrade pushes a ball from the leftmost edge of the i-th row, we call the operation $\mathrm{ROW}\ i$; if he pushes a ball from the topmost edge of the i-th column, we call the operation $\mathrm{COL}\ i$.

For intuitive understanding, a non-zero number x in the matrix given below represents the x-th ball that is pushed in.

In the first test case, a possible series of operations:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \xrightarrow{\mathrm{ROW}\ 3}\xrightarrow{\mathrm{ROW}\ 3} \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 2 & 1 & 0 \end{pmatrix} \xrightarrow{\mathrm{COL}\ 3}\xrightarrow{\mathrm{COL}\ 3} \begin{pmatrix} 0 & 0 & 4 \\ 0 & 0 & 3 \\ 2 & 1 & 0 \end{pmatrix}$$

In the second test case, a possible series of operations:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \xrightarrow{\mathrm{ROW}\ 2}\xrightarrow{\mathrm{ROW}\ 2}\xrightarrow{\mathrm{ROW}\ 2} \begin{pmatrix} 0 & 0 & 0 \\ 3 & 2 & 1 \\ 0 & 0 & 0 \end{pmatrix} \xrightarrow{\mathrm{COL}\ 2}\xrightarrow{\mathrm{COL}\ 2} \begin{pmatrix} 0 & 5 & 0 \\ 3 & 4 & 1 \\ 0 & 2 & 0 \end{pmatrix}$$

In the third test case, a possible series of operations:

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \xrightarrow{\mathrm{ROW}\ 1}\xrightarrow{\mathrm{ROW}\ 2}\xrightarrow{\mathrm{ROW}\ 3} \begin{pmatrix} 1 & 0 & 0 \\ 2 & 0 & 0 \\ 3 & 0 & 0 \end{pmatrix} \xrightarrow{\mathrm{COL}\ 3}\xrightarrow{\mathrm{COL}\ 3}\xrightarrow{\mathrm{COL}\ 3} \begin{pmatrix} 1 & 0 & 6 \\ 2 & 0 & 5 \\ 3 & 0 & 4 \end{pmatrix} \xrightarrow{\mathrm{ROW}\ 1}\xrightarrow{\mathrm{ROW}\ 2}\xrightarrow{\mathrm{ROW}\ 3} \begin{pmatrix} 7 & 1 & 6 \\ 8 & 2 & 5 \\ 9 & 3 & 4 \end{pmatrix}$$

,

C. Dining Hall

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

Inside the large kingdom, there is an infinite dining hall. It can be represented as a set of cells (x, y), where x and y are non-negative integers. There are an infinite number of tables in the hall. Each table occupies four cells $(3x + 1, 3y + 1)$, $(3x + 1, 3y + 2)$, $(3x + 2, 3y + 1)$, $(3x + 2, 3y + 2)$, where x and y are arbitrary non-negative integers. All cells that do not belong to any of the tables are corridors.

There are n guests that come to the dining hall one by one. Each guest appears in the cell $(0, 0)$ and wants to reach a table cell. In one step, they can move to any neighboring by side **corridor** cell, and in their **last** step, they must move to a neighboring by side a free **table** cell. They occupy the chosen table cell, and no other guest can move there.

Each guest has a characteristic t_i, which can either be 0 or 1. They enter the hall in order, starting to walk from the cell $(0, 0)$. If $t_i = 1$, the i-th guest walks to the nearest vacant table cell. If $t_i = 0$, they walk to the nearest table cell that belongs to a completely unoccupied table. Note that other guests may choose the same table later.

The distance is defined as the smallest number of steps needed to reach the table cell. If there are multiple table cells at the same distance, the guests choose the cell with the smallest x, and if there are still ties, they choose among those the cell with the smallest y.

For each guest, find the table cell which they choose.

Input

The first line contains a single integer q ($1 \le q \le 5000$) — the number of test cases. Then follows their description.

The first line of each test case contains a single integer n ($1 \le n \le 50\,000$) — the number of guests.

The second line of each test case contains n integers $t_1, t_2, \ldots, t_n$ ($t_i \in \{0, 1\}$) — the characteristics of the guests.

It is guaranteed that the sum of n for all test cases does not exceed $50\,000$.

Output

For each test case, output n lines — for each guest, the cell where they sit.

Standard Input	Standard Output
2 6 0 1 1 0 0 1 5 1 0 0 1 1	1 1 1 2 2 1 1 4 4 1 1 5 1 1 1 4 4 1 1 2 2 1

Note

Consider the first test case:

The distance from the first guest to the cell $(1, 1)$ is 2, so he sits there.

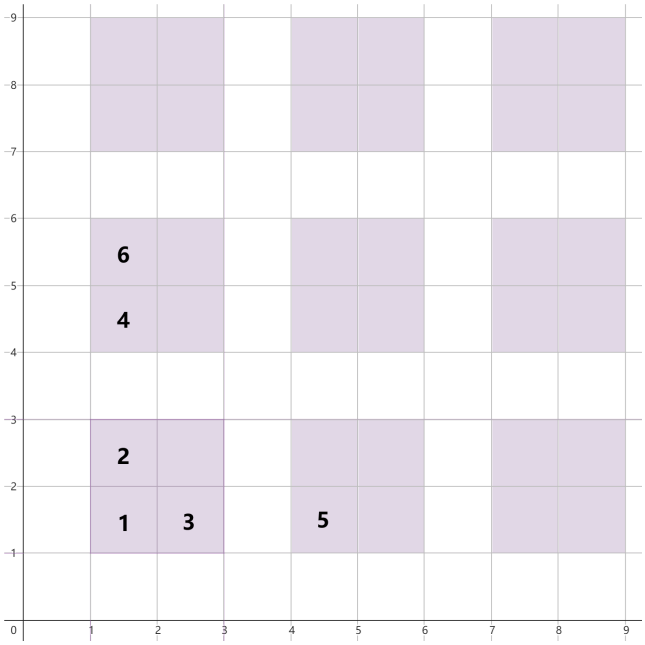
The distance from the second guest to the cell $(1, 2)$ is 3, as is the distance to the cell $(2, 1)$, but since the first coordinate is smaller for the first option, he will choose it.

The distance from the third guest to the cell $(2, 1)$ is 3, so he will choose it.

The distance from the fourth guest to the cell $(1, 4)$ is 5, and he will choose it.

The distance from the fifth guest to the cell $(4, 1)$ is 5.

The distance from the sixth guest to the cell $(1, 5)$ is 6, as is the distance to the cell $(2, 2)$, but since the first coordinate is smaller, he will choose the first option.



,

D. Simple Permutation

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 256 megabytes

Given an integer n. Construct a permutation $p_1, p_2, \ldots, p_n$ of length n that satisfies the following property:

For $1 \leq i \leq n$, define $c_i = \lceil \frac{p_1+p_2+\ldots+p_i}{i} \rceil$, then among $c_1, c_2, \ldots, c_n$ there must be at least $\lfloor \frac{n}{3} \rfloor - 1$ prime numbers.

Input

The first line contains an integer t ($1 \leq t \leq 10$) — the number of test cases. The description of the test cases follows.

In a single line of each test case, there is a single integer n ($2 \leq n \leq 10^5$) — the size of the permutation.

Output

For each test case, output the permutation $p_1, p_2, \ldots, p_n$ of length n that satisfies the condition. It is guaranteed that such a permutation always exists.

Standard Input	Standard Output
3 2 3 5	2 1 2 1 3 2 1 3 4 5

Note

In the first test case, $c_1 = \lceil \frac{2}{1} \rceil = 2$, $c_2 = \lceil \frac{2+1}{2} \rceil = 2$. Both are prime numbers.

In the third test case, $c_1 = \lceil \frac{2}{1} \rceil = 2$, $c_2 = \lceil \frac{3}{2} \rceil = 2$, $c_3 = \lceil \frac{6}{3} \rceil = 2$, $c_4 = \lceil \frac{10}{4} \rceil = 3$, $c_5 = \lceil \frac{15}{5} \rceil = 3$. All these numbers are prime.

,

E1. Canteen (Easy Version)

Input file: standard input
Output file: standard output
Time limit: 2 seconds
Memory limit: 512 megabytes

This is the easy version of the problem. The difference between the versions is that in this version, $k = 0$. You can hack only if you solved all versions of this problem.

Ecrade has two sequences $a_0, a_1, \ldots, a_{n-1}$ and $b_0, b_1, \ldots, b_{n-1}$ consisting of integers. It is guaranteed that the sum of all elements in a does not exceed the sum of all elements in b.

Initially, Ecrade can make exactly k changes to the sequence a. It is guaranteed that k does not exceed the sum of a. In each change:

- Choose an integer i ($0 \leq i < n$) such that $a_i > 0$, and perform $a_i := a_i - 1$.

Then Ecrade will perform the following three operations sequentially on a and b, which constitutes one round of operations:

1. For each $0 \leq i < n$: $t := \min(a_i, b_i), a_i := a_i - t, b_i := b_i - t$;
2. For each $0 \leq i < n$: $c_i := a_{(i-1) \bmod n}$;
3. For each $0 \leq i < n$: $a_i := c_i$;

Ecrade wants to know the minimum number of rounds required for all elements in a to become equal to 0 after exactly k changes to a.

However, this seems a bit complicated, so please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 2 \cdot 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n, k ($1 \leq n \leq 2 \cdot 10^5$, $k = 0$).

The second line of each test case contains n integers $a_0, a_1, \ldots, a_{n-1}$ ($1 \leq a_i \leq 10^9$).

The third line of each test case contains n integers $b_0, b_1, \ldots, b_{n-1}$ ($1 \leq b_i \leq 10^9$).

It is guaranteed that the sum of n across all test cases does not exceed $2 \cdot 10^5$. It is also guaranteed that in each test case the sum of a does not exceed the sum of b, and that k does not exceed the sum of a.

Output

For each test case, output the minimum number of rounds required for all elements in a to become equal to 0 after exactly k changes to a.

Standard Input	Standard Output
4 3 0 1 1 4 5 1 4 4 0 1 2 3 4 4 3 2 1 4 0 2 1 1 2 1 2 2 1 8 0 1 2 3 4 5 6 7 8 8 7 6 5 4 3 2 1	1 4 4 8

Note

In this version, Ecrade cannot make changes to a.

In the first test case:

- After the first round, $a = [0, 0, 0], b = [4, 0, 0]$.

In the second test case:

- After the first round, $a = [3, 0, 0, 1], b = [3, 1, 0, 0]$;
- After the second round, $a = [1, 0, 0, 0], b = [0, 1, 0, 0]$;
- After the third round, $a = [0, 1, 0, 0], b = [0, 1, 0, 0]$;
- After the fourth round, $a = [0, 0, 0, 0], b = [0, 0, 0, 0]$.

,

E2. Canteen (Hard Version)

```
Input file:     standard input
Output file:    standard output
Time limit:     2 seconds
Memory limit:   512 megabytes
```

This is the hard version of the problem. The difference between the versions is that in this version, there are no additional limits on k. You can hack only if you solved all versions of this problem.

Ecrade has two sequences $a_0, a_1, \ldots, a_{n-1}$ and $b_0, b_1, \ldots, b_{n-1}$ consisting of integers. It is guaranteed that the sum of all elements in a does not exceed the sum of all elements in b.

Initially, Ecrade can make exactly k changes to the sequence a. It is guaranteed that k does not exceed the sum of a. In each change:

- Choose an integer i ($0 \leq i < n$) such that $a_i > 0$, and perform $a_i := a_i - 1$.

Then Ecrade will perform the following three operations sequentially on a and b, which constitutes one round of operations:

1. For each $0 \leq i < n$: $t := \min(a_i, b_i), a_i := a_i - t, b_i := b_i - t$;
2. For each $0 \leq i < n$: $c_i := a_{(i-1) \bmod n}$;
3. For each $0 \leq i < n$: $a_i := c_i$;

Ecrade wants to know the minimum number of rounds required for all elements in a to become equal to 0 after exactly k changes to a.

However, this seems a bit complicated, so please help him!

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \leq t \leq 2 \cdot 10^4$). The description of the test cases follows.

The first line of each test case contains two integers n, k ($1 \leq n \leq 2 \cdot 10^5$, $0 \leq k \leq 2 \cdot 10^{14}$).

The second line of each test case contains n integers $a_0, a_1, \ldots, a_{n-1}$ ($1 \leq a_i \leq 10^9$).

The third line of each test case contains n integers $b_0, b_1, \ldots, b_{n-1}$ ($1 \leq b_i \leq 10^9$).

It is guaranteed that the sum of n across all test cases does not exceed $2 \cdot 10^5$. It is also guaranteed that in each test case the sum of a does not exceed the sum of b, and that k does not exceed the sum of a.

Output

For each test case, output the minimum number of rounds required for all elements in a to become equal to 0 after exactly k changes to a.

Standard Input	Standard Output
8 3 0 1 1 4 5 1 4 4 0 1 2 3 4 4 3 2 1 4 0 2 1 1 2 1 2 2 1 8 0 1 2 3 4 5 6 7 8 8 7 6 5 4 3 2 1 3 6 1 1 4	1 4 4 8 0 2 2 1

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

Note

In the fifth test case, all elements in a become 0 after exactly 6 changes.

In the sixth test case, Ecrade can do exactly one change to a_3, then a will become $[1, 2, 2, 4]$.

- After the first round, $a = [3, 0, 0, 0], b = [3, 1, 0, 0]$;
- After the second round, $a = [0, 0, 0, 0], b = [0, 1, 0, 0]$.

In the seventh test case, Ecrade can do exactly one change to a_4, then a will become $[2, 1, 1, 1]$.

- After the first round, $a = [0, 1, 0, 0], b = [0, 1, 1, 0]$;
- After the second round, $a = [0, 0, 0, 0], b = [0, 0, 1, 0]$.

,

F1. Key of Like (Easy Version)

Input file: standard input
Output file: standard output
Time limit: 3 seconds
Memory limit: 512 megabytes

This is the easy version of the problem. The difference between the versions is that in this version, it is guaranteed that $k = 0$. You can hack only if you solved all versions of this problem.

A toy box is a refrigerator filled with childhood delight. Like weakness, struggle, hope ... When such a sleeper is reawakened, what kind of surprises will be waiting?

M received her toy box as a birthday present from her mother. A jewellery designer would definitely spare no effort in decorating yet another priceless masterpiece as a starry firmament with exquisitely shaped gemstones. In addition, l distinct locks secure the tiny universe of her lovely daughter: a hair clip featuring a flower design, a weathered feather pen, a balloon shaped like the letter M ... each piece obscures a precious moment.

A few days ago, M rediscovered her toy box when she was reorganizing her bedroom, along with a ring of keys uniquely designed for the toy box. Attached to the key ring are $(l + k)$ keys, of which l keys are able to open one of the l locks correspondingly, while the other k keys are nothing but counterfeits to discourage brute-force attack. To remind the correspondence, M's mother adorned each key with a gemstone of a different type. However, passing days have faded M's memory away.

"... So I have to turn to you all," M said while laying that ring of keys on the table.

K picked up the keys and examined them carefully. "The appearance of these keys unveils nothing fruitful. Thus, I am afraid that we shall inspect them sequentially."

Although everyone is willing to help M, nobody has a plan. Observing others' reactions, T suggested, "Let's play a game. Everyone tries a key in turn, and who opens the most locks is *amazing*."

n members, including M herself, take turns to unlock the toy box recursively in the same order until all the l locks are unlocked. At each turn, the current member only selects a single key and tests it on exactly one of the locks. To open the toy box as soon as possible, every member chooses the key and the lock that maximize the probability of being a successful match. If there are multiple such pairs, a member will randomly choose one of such pairs with equal probability. Apparently, if a lock has been matched with a key, then neither the lock nor the key will be chosen again in following attempts.

Assume that at the very beginning, the probability that a lock can be opened by any key is equal. If everyone always tries the optimal pairs of keys and locks based on all the historical trials, what will the expected number of successful matches be for each member?

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). The description of the test cases follows.

The only line of the input contains three integers, n, l, k ($1 \le n \le 100, 1 \le l \le 5000, k = 0$) — the number of members participating in the game, the number of locks, and the number of counterfeit keys.

It is guaranteed that the sum of l across all test cases does not exceed 5000.

Output

For each test case, output a single line with n integers $e_1, \ldots, e_n$, where e_i represents the expected number of successful matches, modulo $10^9 + 7$.

Formally, let $M = 10^9 + 7$. It can be shown that the exact 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 e_i that $0 \le x < M$ and $e_i \cdot q \equiv p \pmod{M}$.

Standard Input	Standard Output
4 3 1 0 3 2 0 2 5 0	1 0 0 500000004 1 500000004 200000004 800000008

9 104 0	869203933 991076635 39374313 496894434 9358446 51822059 979588764 523836809 38844739

Note

For the first test case, there is only 1 lock, so the first member opens the only lock with the only key undoubtedly.

For the second test case, there are exactly 2 locks and 2 keys, with each key corresponding to one of the locks. Without extra information, the first member randomly chooses a key and a lock with equal probabilities, for which the probability of success is $1/2$.

- If the first member succeeds, the second member will open the other lock with the other key.
- If the first member fails, then the key she selected can open the other lock, and the other key must correspond to the lock she chose. This information allows both the second and the third member to open a lock.

In conclusion, the expected numbers of successful matches will be:

$$\begin{aligned}
e_1 &= \frac{1}{2} \times 1 + \frac{1}{2} \times 0 = \frac{1}{2} \equiv 500,000,004 \pmod{10^9 + 7}, \\
e_2 &= \frac{1}{2} \times 1 + \frac{1}{2} \times 1 = 1, \\
e_3 &= \frac{1}{2} \times 0 + \frac{1}{2} \times 1 = \frac{1}{2} \equiv 500,000,004 \pmod{10^9 + 7}.
\end{aligned}$$

,

F2. Key of Like (Hard Version)

```
Input file:     standard input
Output file:    standard output
Time limit:     3 seconds
Memory limit:   512 megabytes
```

This is the hard version of the problem. The difference between the versions is that in this version, k can be non-zero. You can hack only if you solved all versions of this problem.

A toy box is a refrigerator filled with childhood delight. Like weakness, struggle, hope ... When such a sleeper is reawakened, what kind of surprises will be waiting?

M received her toy box as a birthday present from her mother. A jewellery designer would definitely spare no effort in decorating yet another priceless masterpiece as a starry firmament with exquisitely shaped gemstones. In addition, l distinct locks secure the tiny universe of her lovely daughter: a hair clip featuring a flower design, a weathered feather pen, a balloon shaped like the letter M ... each piece obscures a precious moment.

A few days ago, M rediscovered her toy box when she was reorganizing her bedroom, along with a ring of keys uniquely designed for the toy box. Attached to the key ring are $(l + k)$ keys, of which l keys are able to open one of the l locks correspondingly, while the other k keys are nothing but counterfeits to discourage brute-force attack. To remind the correspondence, M's mother adorned each key with a gemstone of a different type. However, passing days have faded M's memory away.

"... So I have to turn to you all," M said while laying that ring of keys on the table.

K picked up the keys and examined them carefully. "The appearance of these keys unveils nothing fruitful. Thus, I am afraid that we shall inspect them sequentially."

Although everyone is willing to help M, nobody has a plan. Observing others' reactions, T suggested, "Let's play a game. Everyone tries a key in turn, and who opens the most locks is *amazing*."

n members, including M herself, take turns to unlock the toy box recursively in the same order until all the l locks are unlocked. At each turn, the current member only selects a single key and tests it on exactly one of the locks. To open the toy box as soon as possible, every member chooses the key and the lock that maximize the probability of being a successful match. If there are multiple such pairs, a member will randomly choose one of such pairs with equal probability. Apparently, if a lock has been matched with a key, then neither the lock nor the key will be chosen again in following attempts.

Assume that at the very beginning, the probability that a lock can be opened by any key is equal. If everyone always tries the optimal pairs of keys and locks based on all the historical trials, what will the expected number of successful matches be for each member?

Input

Each test contains multiple test cases. The first line contains the number of test cases t ($1 \le t \le 100$). The description of the test cases follows.

The only line of the input contains three integers, n, l, k ($1 \le n \le 100, 1 \le l \le 5000, 0 \le k \le 25$) — the number of members participating in the game, the number of locks, and the number of counterfeit keys.

It is guaranteed that the sum of l across all test cases does not exceed 5000.

Output

For each test case, output a single line with n integers $e_1, \ldots, e_n$, where e_i represents the expected number of successful matches, modulo $10^9 + 7$.

Formally, let $M = 10^9 + 7$. It can be shown that the exact 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 e_i that $0 \le x < M$ and $e_i \cdot q \equiv p \pmod{M}$.

Standard Input	Standard Output
4 3 1 4 3 2 0 25 2 5	800000006 800000006 400000003 500000004 1 500000004 142857144 166666668 615646263 639455787 234126986 257936510 195918369 502040820 478316330 81264173

4 102 9	190523433 471438023 23809524 0 0 0 0 0 0 0 0 0 0 0 0 568832210 85779764 969938175 375449967

Note

For the first test case, there is only 1 lock, so the strategy will always be choosing any key that no one has ever tried. Since there are $1 + 4 = 5$ keys in total, the probability that each member successfully opens the lock will be $2/5, 2/5, 1/5$ respectively, which are also the expected numbers of successful matches.

For the second test case, there are exactly 2 locks and 2 keys, with each key corresponding to one of the locks. Without extra information, the first member randomly chooses a key and a lock with equal probabilities, for which the probability of success is $1/2$.

- If the first member succeeds, the second member will open the other lock with the other key.
- If the first member fails, then the key she selected can open the other lock, and the other key must correspond to the lock she chose. This information allows both the second and the third member to open a lock.

In conclusion, the expected numbers of successful matches will be:

$$\begin{aligned}
e_1 &= \frac{1}{2} \times 1 + \frac{1}{2} \times 0 = \frac{1}{2} \equiv 500,000,004 \pmod{10^9 + 7}, \\
e_2 &= \frac{1}{2} \times 1 + \frac{1}{2} \times 1 = 1, \\
e_3 &= \frac{1}{2} \times 0 + \frac{1}{2} \times 1 = \frac{1}{2} \equiv 500,000,004 \pmod{10^9 + 7}.
\end{aligned}$$