# Official Problem Set

## DO NOT OPEN UNTIL CONTEST BEGINS

# 2017 ACM ICPC
# ASIA CHENNAI
# REGIONAL CONTEST

2017

acm

icpc International Collegiate Programming Contest

icpc.foundation

Problem code: **ELHIDARR**

Problem name: **Find an element in hidden array**

There is an array of length **N** consisting of non-negative integers. The array is sorted in non-decreasing order. Each number in the array appears exactly **K** times, except one element, which appears at least once, but less than **K** times. Your task is to identify that element.

This is an interactive problem. You are only given the integer **N** in the input. Both the array and the value of **K** are hidden. You are allowed to ask the judge the following queries: What is the value of the element at index **i** of the array? Identify the value of the element with frequency less than **K** by asking at most 60 such queries.

**Input and Output**

The first line of the input contains a single integer **T** denoting the number of test cases.

For each test case, you should start by reading a single line containing one integer **N** from the input.

You can interact with the judge using the standard input and output. There are two types of operations: to perform one operation, you should print to the standard output a line containing two space-separated integers **type** and **val**.

- If **type** = 1, you are asking the judge a query for the value of the element of the array at index **val**. After printing this line, the judge will print to the standard input a line containing one integer corresponding to the value of the element at index **val**.
- If **type** = 2, you are telling the judge that the element with frequency less than **K** is **val**. For each test case, you should perform this operation exactly once at the end. This is not counted towards the 60 queries.

**Note**

Don't forget to flush the standard output after printing each line. It can be done using fflush(stdout) in C/C++, System.out.flush() in Java and system.out.flush() in Python.

If you ask more than 60 queries, your program will get the verdict Wrong Answer.

## Constraints

- $1 \le \mathbf{T} \le 10^4$
- $3 \le \mathbf{N} \le 10^5$
- $2 \le K \le \mathbf{N - 1}$
- each element of the array lies between 1 and $10^9$ inclusive

## Example

| Input / judge feedback | your output |
|---|---|
| 1 | |
| 3 | |
| | 1 2 |
| 1 | |
| | 1 3 |
| 5 | |
| | 1 1 |
| 1 | |
| | 2 5 |

## Explanation

**Example case 1:** Suppose the array is [1, 1, 5]. Note that the value of **K** is 2, but it is hidden from you.

In the first query, you request the value of the 2nd element and the judge answers 1. Then you request the value of the 3rd element and the judge answers 5, then the value of the first element and the judge answers 1.

Now, you tell the judge that the answer is 5. You made a total of 3 queries.

Problem code: **DEFOREST**
Problem name: **Chef and Deforestation**

The Chef has come to a forest. He sees **N** trees in a line. He wants to tie a rope from the top of one tree to the top of another. To successfully tie this rope, he needs to cut the trees that get in the way of the rope. However, he does not like this kind of deforestation and wants your help to choose the best pair of trees between which to tie the rope.

Formally, you are given **N** trees on the X-axis. Tree $\mathbf{T_i}$ has height $\mathbf{H_i}$ and is planted at $\mathbf{x = x_i}$.

You need to choose two trees $\mathbf{T_i}$ and $\mathbf{T_j}$ (where $\mathbf{x_i} < \mathbf{x_j}$ and $\mathbf{H_i} < \mathbf{H_j}$) and connect the **tops** of these trees with a rope such that the following conditions are satisfied:

- The angle between the rope and the X-axis is equal to **45** degrees.
- The rope **does not pass** through any other trees.

In order to satisfy the second condition, you are allowed to reduce the height of any trees except $\mathbf{T_i}$ and $\mathbf{T_j}$. Formally, you should choose numbers $\mathbf{C_1}, \mathbf{C_2}, \ldots, \mathbf{C_N}$, such that $\mathbf{0 \le C_k \le H_k}$ for each $\mathbf{1 \le k \le N}$ and $\mathbf{C_i = C_j = 0}$. Then decrease $\mathbf{H_k}$ by $\mathbf{C_k}$ for each $\mathbf{1 \le k \le N}$.

Find the maximum value of **(length of the rope between the two tops) - (sum of $\mathbf{C_k}$ for each tree $\mathbf{T_k}$)**.

Note that the rope can touch the top of some intermediate tree. Look at the examples given for such a scenario.

### Input

- The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
- The first line of each test case contains a single integer **N** denoting the number of trees.
- Each of the following **N** lines contains two space-separated integers denoting $\mathbf{x_i}$ and $\mathbf{H_i}$.

## Output

For each test case, print a single line with the maximum value of the given expression. Your answer will be considered correct if the **absolute or relative error ≤ $10^{-6}$**. If it's impossible to find any valid pair of trees to tie the rope between, print **-1** instead.

## Constraints

- $1 \leq \mathbf{T} \leq 100$
- $1 \leq \mathbf{x_i}, \mathbf{H_i} \leq 10^9$
- $1 \leq \mathbf{N} \leq 2.5 \cdot 10^5$
- all $\mathbf{x_i}$ will be distinct
- sum of **N** over all test cases **≤ $5 \cdot 10^5$**

## Example

**Input:**

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

**Output:**

```
2.82842712
-0.17157287
```

## Explanation

**Example case 1:** Tie the rope from tree 1 (1,1) to tree 3 (3,3). You don't need to cut down any trees. Note that this rope would touch the top of tree 2, but that is fine. Answer = length of rope = 2 · sqrt(2) = 2.82842.

**Example case 2:** Tie the rope from tree 1 (1,1) to tree 3 (3,3). You need to cut down tree 2 from height 5 to height 2. Answer = length of rope - (height reduction of tree 2) = 2 · sqrt(2) - 3 = -0.17157287.

Problem code: **BLREDSET**
Problem name: **Black and Red vertices of Tree**

You are given a tree **T** with **N** vertices. The vertices are numbered from 1 to **N**. Some of the vertices are colored black and some are colored red. Also, some vertices can be uncolored. There is at least one black and at least one red vertex.

Compute the number of subsets of vertices **W** such that:

- Each vertex in **W** is uncolored.
- **W** is a connected subgraph of **T**.
- If you remove all the vertices of set **W**, there will be at least one pair of vertices (**u**, **v**) such that:
  - **u** is colored black and **v** red.
  - There is no path from node **u** to node **v**.

Output your answer **modulo $10^9$ + 7**.

### Input

- The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
- The first line of each test case contains a single integer **N** denoting the number of vertices.
- Each of the next **N - 1** lines contains two space-separated integers **u** and **v** denoting an edge between vertices **u** and **v**.
- The next line contains **N** space-separated integers denoting the colors of the vertices. The **i**-th of these integers is 0 if vertex **i** is uncolored, 1 if it's black or 2 if it's red.

### Output

For each test case, print a single line containing one integer denoting the number of ways to select a valid subset **W**, modulo $10^9 + 7$.

## Constraints

- $1 \leq \mathbf{T} \leq 2 \cdot 10^5$
- $2 \leq \mathbf{N} \leq 10^5$
- $1 \leq \mathbf{u}, \mathbf{v} \leq \mathbf{N}$
- Sum of **N** over all test cases doesn't exceed $10^6$

## Example

**Input**

```
2
6
1 2
1 3
1 4
3 5
3 6
0 1 0 1 2 0
6
1 2
1 3
1 4
3 5
3 6
1 0 0 0 2 0
```

**Output**

```
5
2
```

## Explanation

**Example case 1:** The possible subsets **W** are **{1}, {3}, {1, 3}, {3, 6}** and **{1, 3, 6}**.

**Example case 2:** The possible subsets **W** are **{3}** and **{3, 6}**.

Problem code: **OBEXPVAL**
Problem name: **Obtain Desired Expected Value**

You are given $\mathbf{n}$ non-negative integers $\mathbf{x_1, x_2, ..., x_n}$. You are also given a positive integer $\mathbf{E}$. You have to find $\mathbf{n}$ non-negative real numbers $\mathbf{p_1, p_2, ..., p_n}$ such that $\mathbf{p_1 \cdot x_1 + p_2 \cdot x_2 + p_3 \cdot x_3 + .... + p_n \cdot x_n = E}$and $\mathbf{p_1 + p_2 + ... + p_n = 1}$.

If it's not possible to find $\mathbf{n}$ such numbers, output -1 instead.

**Input**

- The first line of the input contains a single integer $\mathbf{T}$ denoting the number of test cases. The description of $\mathbf{T}$ test cases follows.
- The first line of each test case contains two space-separated integers $\mathbf{n}$ and $\mathbf{E}$.
- The second line contains $\mathbf{n}$ space-separated integers $\mathbf{x_1, x_2, ..., x_n}$.

**Output**

For each test case, print a single line containing $\mathbf{n}$ space-separated real numbers denoting the values of $\mathbf{p_1, p_2, ..., p_n}$. If there is more than one possible solution, you may output any one. If there is no solution, print -1 instead.

When a solution exists, your answer will be considered correct if the absolute value of the expression $\mathbf{p_1 \cdot x_1 + p_2 \cdot x_2 + p_3 \cdot x_3 + ... + p_n \cdot x_n - E}$ doesn't exceed $10^{-6}$ and the value of $\mathbf{|(p_1 + p_2 + p_3 + ... + p_n) - 1|}$ doesn't exceed $10^{-6}$.

**Constraints**

- $1 \leq \mathbf{T} \leq 10^5$
- $1 \leq \mathbf{n, E} \leq 10^3$
- $1 \leq \mathbf{x_i} \leq 10^3$
- sum of $\mathbf{n}$ over all test cases doesn't exceed $10^6$

## Example

**Input**

```
3
1 2
1
3 3
1 2 3
4 4
1 2 3 6
```

**Output**

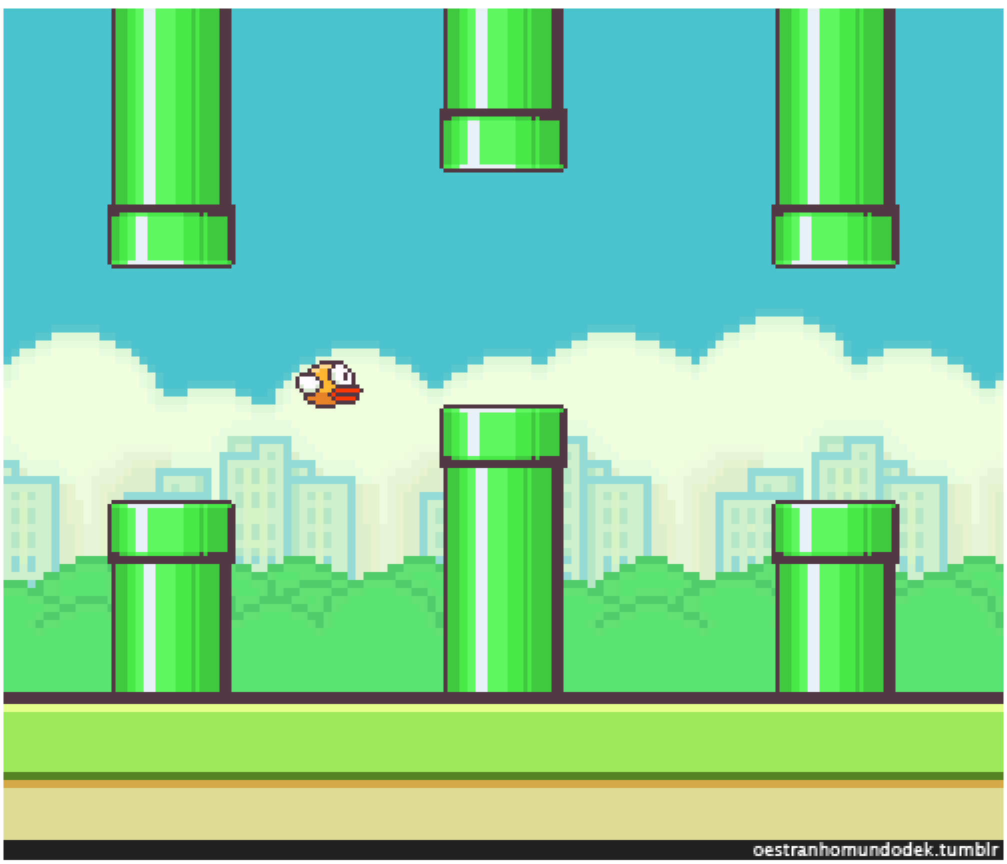
```
-1
0 0 1.00
0 0 0.66666666666 0.33333333333
```

Problem code: **CODIE**
Problem name: **Codie Bird**

Codie is a super intelligent bird stuck inside a flappy bird simulation. The game is simple: if the position of the bird is denoted by **(x, y)**, then the bird can move in the following directions in one step:

- right — to **(x + 1, y)**,
- up-right — to **(x + 1, y + 1)**,
- down-right — to **(x + 1, y - 1)**.

The screen has width **K** and you are **not allowed** to move outside of the screen, i.e. the position of the bird must satisfy 1 ≤ **y** ≤ **K** at each moment.




Codie is moving from **(1, K/2)** to **(N, K/2)**. However, there is exactly one obstacle in the simulation. This obstacle occurs at a random **x**-coordinate between **2** and **N -**

**1** (both inclusive), where each **x** has equal probability. The obstacle is composed of **two** vertical pipes. One of them is at the top and covers all **y**-coordinates between **b** and **K** (both inclusive). The other one is at the bottom and covers all **y**-coordinates between 1 and **a** (both inclusive). The pipes only allow Codie to move through **y**-coordinates that are **≥ a+1** and **≤ b-1** at this particular **x**-coordinate.

Codie, being a super smart and geeky bird, is interested in finding the expected number of paths from **(1, K/2)** to **(N, K/2)**. Help Codie find the answer!

Specifically, let's assume that the expected number of paths is a fraction of the form **P / Q**, where **P** and **Q** are coprime. Let $\mathbf{Q^{-1}}$ be the multiplicative inverse of **Q** modulo $10^9 + 7$. Compute $\mathbf{P \cdot Q^{-1}}$ **modulo** $\mathbf{10^9 + 7}$.

## Input

The only line of the input contains four space-separated integers **N**, **K**, **a** and **b** denoting the final **x**-coordinate, the width of the screen, the highest coordinate blocked by the bottom pipe and the lowest coordinate blocked by the top pipe.

## Output

Print a single line containing one integer — the expected number of paths from **(1, K/2)** to **(N, K/2)** modulo $10^9 + 7$.

## Constraints

- $3 \le \mathbf{N} \le 10^9$
- $2 \le \mathbf{K} \le 50$
- **K** is even
- $1 \le \mathbf{a} < \mathbf{b} \le \mathbf{K}$

## Example

**Input:**

```
5 4 1 3
```

**Output:**

```
666666679
```

## Explanation

Let **X** denote the starting and ending coordinate, and **B** denote the cells blocked by pipes.

| | | | | | |
|---|---|---|---|---|---|
| 4 | | B | | | |
| 3 | | B | | | |
| 2 | X | | | | X |
| 1 | | B | | | |
| 0 | 1 | 2 | 3 | 4 | 5 |

(i) When pipe is located at x = 2, number of ways = 7

| | | | | | |
|---|---|---|---|---|---|
| 4 | | | B | | |
| 3 | | | B | | |
| 2 | X | | | | X |
| 1 | | | B | | |
| 0 | 1 | 2 | 3 | 4 | 5 |

(ii) When pipe is located at x = 3, number of ways = 9

| | | | | | |
|---|---|---|---|---|---|
| 4 | | | | B | |
| 3 | | | | B | |
| 2 | X | | | | X |
| 1 | | | | B | |
| 0 | 1 | 2 | 3 | 4 | 5 |

(iii) When pipe is located at x = 4, number of ways = 7

The expected value is $(7 \cdot 1/3) + (9 \cdot 1/3) + (7 \cdot 1/3) = 23/3 = 666666679$ (modulo $10^9 + 7$).

Problem code: **AREAFIGR**
Problem name: **Area of an amazing figure**

There is an equilateral triangle ABC with side length **2*a**. The coordinates of the triangle's vertices are A = **(-a, 0)**, B = **(a, 0)** and C = **(0, sqrt(3) * a)**. Next, there are three circles with centers at the midpoints of sides AB, BC and AC respectively and radii $\mathbf{r_1}$, $\mathbf{r_2}$ and $\mathbf{r_3}$ respectively. Compute the area of the intersection of the triangle and the three circles.

**Input**

- The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
- The first and only line of each test case contains four space-separated integers $\mathbf{a, r_1, r_2, r_3}$.

**Output**

For each test case, print a single line containing one real number corresponding to the area of the intersection. Your answer will be considered correct if it has an absolute or relative error $\leq 5 * 10^{-2}$.

**Constraints**

- $1 \leq \mathbf{T} \leq 20$
- $1 \leq \mathbf{a} \leq 50$
- $1 \leq \mathbf{r_1, r_2, r_3} \leq 200$

## Example

**Input**

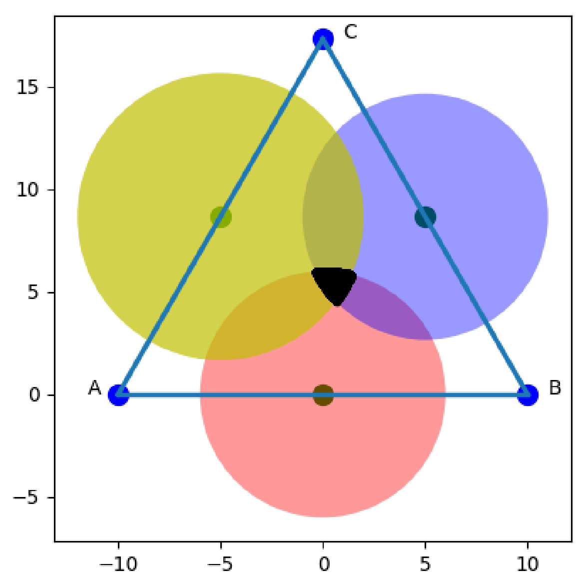
```
5
10 5 5 5
10 6 6 7
10 12 13 14
10 15 15 15
10 19 20 18
```

**Output**

```
0.0000000
1.417303000
138.1789
163.712
173.2064190
```

## Explanation

**Example case 2:** The black colored area shown in the figure is the intersection between the triangle and the three circles.



Problem code: **MRO**
Problem name: **Method Resolution Order**

Zotlin is an object-oriented programming language that provides features such as **Classes** and **Functions**. Zotlin also has the following features:

- A function with the same name and same declarations can have different implementations in different classes.
- Each class can inherit zero or more classes (excluding itself). If **Class A** inherits **Class B**, we say that **Class A** is a *child* of **Class B**.
- A class cannot inherit another class if a cycle is formed by inheritance.

For example, if Class A inherits Class B and Class B inherits Class C, then Class C cannot inherit Class A.

A Class X is called an ancestor of Class Y, if Y inherits directly or indirectly from Class X. ie, say Class Y inherits from Class Z, which inherits from Class X. Then Class X is an ancestor of both Class Y and Class Z.

Mr. Zourist — a very popular programmer — wrote a program in Zotlin with **N** classes numbered from **1** to **N**. Some of these classes have an implementation of a function **F**. The Zotlin compiler needs to determine which definition of function **F** should be called for an object of **Class 1**.

In order to deterministically decide which implementation of function **F** to use, the compiler uses a list called the **Resolution Order** *(RO list)*. This is a list containing **Class 1** and its ancestors. The compiler iterates through this list to search for a class that contains an implementation of function **F**.

The properties of an RO list are as follows:

- The 1st member of the list is **Class 1**.
- The list contains only the ancestors of Class 1. All the Classes in the list are unique.
- It is guaranteed that **no Class occurs after any of its ancestors** in the list.

Let's define the ***Special RO list*** as an RO list of length **N** where the $\mathbf{i^{th}}$ member in the list is **Class i**. That is, the Special RO list is **[1, 2, 3, ... , N]**.

Find out the number of distinct class-inheritance hierarchies Mr. Zourist could have created in his program for which one of the valid RO lists is the *Special RO list*. Print your output modulo $10^9 + 7$.

Two class-inheritance hierarchies are considered different if they have at least one class that inherits a different list of classes.

## Input

The first line of input contains a positive integer **T** - denoting the total number of testcases.

**T** lines follow - the $\mathbf{i^{th}}$ line contains a single positive integer **N**- denoting the number of classes written by Mr. Zourist.

## Output

For each testcase, print the required number of distinct class-inheritance hierarchies modulo $\mathbf{10^9 + 7}$ on a new line.

## Constraints

- $\mathbf{1 \le T \le 10^5}$
- $\mathbf{1 \le N \le 10^5}$

## Example

**Input**

```
2
3
6
```

**Output**

```
3
9765
```

## Explanation

**Explanation for testcase 1:**

Let "class X" denote that X does not inherit any class

Let "class X(Y)" denote that X inherits Y

Let "class X(Y, Z)" denote that X inherits Y, Z

**First possibility:**

```
class 2
class 3
class 1(2, 3)
```

**Second possibility:**

```
class 3
class 1(2)
class 2(3)
```

**Third possibility:**

```
class 3
class 1(2, 3)
class 2(3)
```

These are the three valid class-inheritance hierarchies, and hence the answer is 3.

Note that the following hierarchy is not valid:

```
class 1(2)
class 2
class 3
```

This is because 3 should have been an ancestor of 1, but in the above hierarchy, 3 is a stand-alone class.

Problem code: **VRTXCOVR**
Problem name: **Vertex Cover**

You are given an undirected graph **G = (V, E)** containing **N** nodes and **M** edges. The nodes are numbered from 1 to **N**. A subset **C** of **V** is a *vertex cover* if for every edge (**u**, **v**) ∈ **E**, at least one of **u** and **v**belong to **C**. Note that **C = V** is always a vertex cover.

Consider a partition of **V** into two sets **A** and **B**. It is said to be a *valid* partition, if the following two conditions are satisfied: **A** should be a vertex cover. And for each **i** such that 1 ≤ **i** ≤ **n/2**, nodes **2*i** and **2*i - 1** don't belong to the same set (i.e. one belongs to set **A** and the other to set **B**).

Determine if a *valid* partition exists. If it exists, provide an example of one *valid* partition.

**Input**

- The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
- The first line of each test case contains two space-separated integers **N** and **M** denoting the number of nodes and number of edges in the graph respectively.
- Each of the following **M** lines contains two space-separated integers **u** and **v** denoting an edge between nodes **u** and **v**.

**Output**

- For each test case, print a line containing the string **"possible"** (without quotes) if a solution exists or **"impossible"** otherwise.
- If a solution exists, print a second line containing a binary string. The **i**-th character of this string should be '0' if vertex **i** is in set **B** or '1' if it is in set **A**.

## Constraints

- $1 \leq \mathbf{T} \leq 10^5$
- $1 \leq \mathbf{N} \leq 2 \cdot 10^5$
- $0 \leq \mathbf{M} \leq 2 \cdot 10^5$
- $1 \leq \mathbf{u}, \mathbf{v} \leq \mathbf{N}$
- $1 \leq$ sum of $\mathbf{N}$ over all test cases $\leq 10^6$
- $1 \leq$ sum of $\mathbf{M}$ over all test cases $\leq 10^6$

## Example

**Input:**

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

**Output:**

```
possible
011
impossible
```

## Explanation

**Example case 1:** We can put nodes numbered 2 and 3 in set A and node 1 in set B. Note that this is a *valid* partition because set A is a vertex cover; also, nodes numbered 1 and 2 belong to different sets.

**Example case 2:** There exists no partition which satisfies the conditions.

Problem code: **GOODBAD**
Problem name: **Good and Bad Persons**

Chef is a really nice and respectful person, in sharp contrast to his little brother, who is a very nasty and disrespectful person. Chef always sends messages to his friends in all small letters, whereas the little brother sends messages in all capital letters.

You just received a message given by a string **s**. You don't know whether this message is sent by Chef or his brother. Also, the communication channel through which you received the message is erroneous and hence can flip a letter from uppercase to lowercase or vice versa. However, you know that this channel can make at most **K** such flips.

Determine whether the message could have been sent only by Chef, only by the little brother, by both or by none.

**Input**

- The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
- The first line of each test case contains two space-separated integers **N** and **K** denoting the length of the string **s** and the maximum number of flips that the erroneous channel can make.
- The second line contains a single string **s** denoting the message you received.

**Output**

For each test case, output a single line containing one string — "chef", "brother", "both" or "none".

**Constraints**

- $1 \leq \mathbf{T} \leq 1000$
- $1 \leq \mathbf{N} \leq 100$
- $0 \leq \mathbf{K} \leq \mathbf{N}$
- **s** consists only of (lowercase and uppercase) English letters

## Example

**Input**

```
4
5 1
frauD
5 1
FRAUD
4 4
Life
10 4
sTRAWBerry
```

**Output**

```
chef
brother
both
none
```

## Explanation

**Example case 1:** Only one flip is possible. So it is possible that Chef sent "fraud" and the channel flipped the last character to get "frauD". However, it is not possible for the brother to have sent "FRAUD", because then it would need 4 flips. Hence the answer is "chef".

**Example case 2:** Only one flip is possible. So it is possible that the brother sent "FRAUD" and the channel didn't flip anything. However, it is not possible for Chef to have sent "fraud", because then it would need 5 flips. Hence the answer is "brother".

**Example case 3:** Four flips are allowed. It is possible that Chef sent "life" and the channel flipped the first character to get "Life". It is also possible that the brother sent "LIFE" and the channel flipped the last three characters to get "Life". Hence the answer is "both".

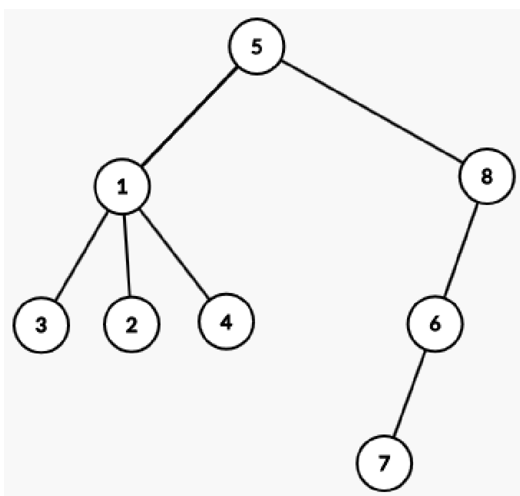
**Example case 4:** Four flips are allowed. It is not possible that Chef sent "strawberry", because it would need five flips to get "sTRAWBerry". It is also not possible that the brother sent "STRAWBERRY", because that would also need five flips. Hence the answer is "none".

Problem code: **MANCBST**
Problem name: **Chef Counts Semi-BSTs**

Everyone has heard of binary search trees or BSTs. A BST is a rooted tree, such that the root has atmost two children. The subtree rooted at each of root's children should be valid BSTs. All the values in the left subtree should be smaller than the root and all the values in the right subtree should be larger than the root.

However, there's a new data structure in town called semi-BST. Formally, a semi-BST is a tree with **N** nodes which satisfies the following conditions:

- Each node contains a distinct value.
- The root has at most two children.
- The right subtree of the root (if it exists) is a valid semi-BST.
- The left subtree of the root (if it exists) is any unordered rooted tree. That is, the children of a vertex are not ordered, and hence no concept of left or right child.
- The BST property for the root is maintained, i.e. all the values in the left subtree of the root are lesser than that of the root and all the values in the right subtree of the root are greater than that of the root.




For example, let us show that tree shown above is a semi-BST. The root of the tree is 5. All the elements in the left subtree are smaller than 5 and all the elements in the right subtree are greater than 5.

Now, all we need to show is that the right subtree is a valid semi-BST:

We are now considering the subtree rooted at 8, and hence the current root is 8. All the elements in the left subtree are smaller than 8 and all the elements in the right subtree (which doesn't exist) are greater than 8. Hence the subtree rooted at 8 is a valid semi-BST. And hence the entire tree, rooted at 5 is a valid semi-BST.

Given **N**, find the number of semi-BSTs having exactly **N** nodes, and such that all the **N** values in the nodes are from {1, 2, ..., **N**}. Output the answer modulo 663224321.

### Input

- The first line of the input contains a single integer **T** denoting the number of test cases. The description of **T** test cases follows.
- The first and only line of each test case contains a single integer **N** denoting the number of nodes.

### Output

For each test case, print a single line containing one integer — the number of semi-BSTs modulo 663224321.

### Constraints

- $1 \leq \mathbf{T} \leq 100000$
- $1 \leq \mathbf{N} \leq 100000$

### Example 1

```
Input:
4
1
2
3
4

Output:
1
2
5
18
```

---

Problem code: **MINSTR**
Problem name: **Minimize the string**

---

You are given **n** strings $\mathbf{s_1, s_2, ..., s_n}$. Each of these strings consists only of letters 'a' and 'b', and the length of each string can be at most 2. In other words, the only allowed strings are "a", "b", "aa", "ab", "ba" and "bb".

Consider a permutation $\mathbf{p} = \{\mathbf{p_1}, \mathbf{p_2}, ..., \mathbf{p_n}\}$ of the integers $\{1, 2, ..., \mathbf{n}\}$. Using this permutation, you can obtain a new string $\mathbf{S = s_{p1} + s_{p2} + ... + s_{pn}}$, where the operator + denotes concatenation of strings.

You can shorten the string **S** by performing the following operation any number of times: choose two consecutive equal characters and remove one of these characters from the string. For example, the string "aabb" can be shortened to "abb" or "aab" in one operation, and then optionally it could still be shortened to "ab".

You are allowed to choose any permutation **p**. Take the string **S** obtained using this permutation, and using any sequence of operations, minimize the string length. Find the minimum possible length of the string obtainable.

**Input**

The first line of the input contains an integer **T** denoting the number of test cases. The description of **T** test cases follows.

The first line of each test case contains an integer **n**.

The second line of each test case contains **n** space-separated strings $\mathbf{s_1, s_2, ..., s_n}$.

**Output**

For each test case, output a single line containing one integer corresponding to the minimum possible length of the shortened string.

**Constraints**

- $1 \leq \mathbf{T} \leq 10^5$
- $1 \leq \mathbf{n} \leq 10^5$
- sum of **n** over all test cases won't exceed $10^6$

## Example

**Input**

```
2
2
ba ab
4
a b a b
```

**Output**

```
3
2
```

## Explanation

### Testcase 1:

You can consider the permutation (2, 1). Using this, you get the string $\mathbf{S} = \mathbf{s_{p1}} + \mathbf{s_{p2}}$ = ab + ba = abba. You can then take the two adjacent b's and remove one of them to get aba, whose length is 3. You cannot do any better, and hence the answer is 3.

### Testcase 2:

You can consider the permutation (1, 3, 2, 4). Using this, you get the string $\mathbf{S} = \mathbf{s_{p1}} + \mathbf{s_{p3}} + \mathbf{s_{p2}} + \mathbf{s_{p4}}$ = a + a + b + b = aabb. You can then take the two adjacent b's and remove one of them to get aab. Then you can take the two adjacent a's and remove one of them to get ab. We end up with a length of 2, and you cannot do any better. Hence the answer is 2.