Problem A

Are the Nodes Reachable?

You are given a directed acyclic graph consisting of N nodes, numbered from 1 to N, and M directed edges. Your task is to answer Q queries about reachability between nodes.

For each query, you will be given two nodes, U and V:

- If node V is reachable from node U through existing roads (directed edges), you should output 0 as no additional road is required.
- If V is not reachable from U, you are allowed to build one temporary edge between any two nodes in the graph to make V reachable from U. After the query is answered, this road will be destroyed.

You must determine the minimum cost of constructing this road, if needed. The cost of building a road between node X and node Y is defined as $|X - Y|$, i.e., the absolute difference between their node labels.

Input

The first line will contain a single integer $T (1 \le T \le 100)$. First line of each test case will contain two integers N $(1 \le N \le 10^4)$ and M $(1 \le M \le 10^6)$ where N denotes the number of nodes in the graph and M denotes the number of directed edges.

Each of the next M lines will contain two integers U and V $(1 \le U, V \le N, U \ne V)$ denoting a directed edge from U to V**.** Next line of the input will contain a single integer $Q(1 \le Q \le 10^6)$ denoting the number of queries. Next Q lines will contain two integers U and V $(1 \le U, V \le N)$ for which the answer of the query needs to be processed.

It is guaranteed that, $\Sigma N \le 2 \times 10^5$, $\Sigma M \le 10^6$, $\Sigma Q \le 10^6$ across all test cases.

Output

For each test case, you need to print the case number on the first line in the format "**Case T:**" where **T** is the case number. Each of the next Q lines should contain one integer, the answer to the query.

Sample Input	Output for Sample Input
1 4 4 1 2 1 3 1 4 4 3 2 2 3 2 4	Case 1: 1 1

It is recommended to use fast input/output methods due to the large input size.

Problem B

Yet Another Crossover Episode

Pebae has an integer array $a_1, a_2, \cdots, a_n$. Please help her find the maximum of $gcd(a_i \oplus a_j, a_i \& a_j)$ over all $1 \le i, j \le n$. Additionally, you need to count the total number of pairs (i, j) that achieve this maximum value.

Here, $gcd(x, y)$ is the greatest common divisor of x and y, $gcd(x, 0) = x$ for $x \ge 0$, $\oplus$ is the bitwise XOR operator, and & is the bitwise AND operator.

Input

The first line of the input contains an integer t $(1 \le t \le 10^5)$ — the number of test cases.
The first line of each test case contains an integer n $(1 \le n \le 2^{19})$.
The second line contains n space-separated integers $a_1, a_2, \cdots, a_n (0 \le a_i < 16n)$.
It is guaranteed that the sum of n over all test cases does not exceed 2^{20}.

Output

For each test case, output two space-separated integers — the maximum value and the number of pairs that achieve this maximum value.

Sample Input	Output for Sample Input
4 4 1 9 1 9 1 0 3 0 0 1 7 12 2 3 0 110 1 69	9 4 0 1 1 5 111 2

It is recommended to use fast input/output methods due to the large input size.

Note

In the first test case, the maximum value that can be obtained is 9 for the pairs $(2, 2)$, $(2, 4)$, $(4, 2)$ and $(4, 4)$.

Problem C

Cut the Stick, Share You Must

Yoda recently noticed his stick is worn out after years of use. Thinking it's time for an upgrade, he decides to get a new stick. But the old one holds so many memories that he cannot part with it entirely. Instead, he plans to cut the stick into pieces, share them with his friends, and keep one piece for himself.

However, he doesn't want to make the cuts randomly. Being fond of primes and integers, Yoda wants the stick to be cut into integer lengths such that the GCD of the resulting lengths is a prime number. The stick's total length is also an integer.

At first, the task seems simple. But as Yoda starts counting the possibilities, he realizes there are too many ways to cut the stick. Frustrated, he turns to you and says: *"Overwhelmed, I am, with the possibilities. Help me figure out the answer, will you? Only then, satisfied, I shall be."*

More formally, Yoda will give you two integers n and k, where:

- n denotes the length of his stick
- k is the number of cuts you need to make on the stick.

Your task is to count the number of ways to make exactly k cuts such that the GCD (Greatest Common Divisor) of the lengths of the resultant $k + 1$ stick pieces is a prime number.

A prime number is an integer greater than 1 and only divisible by 1 and itself.

The cuts must be made at integer points along the stick, and each resulting stick piece must have a positive integer length.

For $n = 10$ & $k = 2$, there are 6 valid ways to cut the stick.
Please, refer to the table*. *We can also see that the order in which you perform the cuts does not matter.

Cut Positions	Lengths of The Stick Pieces	GCD
$\{2, 4\}$	$\{2, 2, 6\}$	2
$\{2, 6\}$	$\{2, 4, 4\}$	2
$\{2, 8\}$	$\{2, 6, 2\}$	2
$\{4, 6\}$	$\{4, 2, 4\}$	2
$\{4, 8\}$	$\{4, 4, 2\}$	2
$\{6, 8\}$	$\{6, 2, 2\}$	2

Input

The first line contains an integer $T\ (1 \leq T \leq 10^5)$, the number of test cases. Each of the next T lines contains two integers n and $k\ (1 \leq k < n \leq 10^6)$.

Output

For each test case, you need to print the number of valid ways to cut the stick in a single line. Since the result can be quite large, output the value of the result modulo $1000000007 (10^9 + 7)$.

Sample Input	Output for Sample Input
4 4 1 8 2 10 2 12 3	1 3 6 11

It is recommended to use fast input/output methods due to the large input size.

Problem D

CatGPT

A group of cats is called a clowder. There are N cats living in a street, each of them belongs to exactly one clowder. The cats are numbered from 1 to N, and the i-th cat belongs to clowder c_i.

Sometimes two or more cats with consecutive IDs convince their respective clowders to hang out together and form one big clowder, where every cat from those old clowders would be a part of this new clowder even if they were not part of the group of cats who came up with this idea. These types of events can be described by an interval $e = [L, R]$, where cats $L,\ L + 1,\ \ldots\ R$ convinces their clowders to merge. There are M such *possible* events – the ***i'th*** can be described by $e_i = \left[L_i, R_i\right]$.

For example, let's assume that there are 5 cats and they belong to clowders *A, B, C, A, D* respectively. If cats 2, 3 and 4 convince their respective clowders *B, C, A* to form a new clowder *X*, the new clowder *X* will consist of cats 1, 2, 3 and 4. Notice that although cat 1 was not one of the cats who convinced its clowder, it does belong to the new clowder because cat 4 convinced clowder *A*. This example event can be described as $[2, 4]$.

A query is defined as an interval of events $[x, y]$ where $1 \leq x \leq y \leq M$. For each such query, report the **maximum possible size** of a clowder if **any number of events** in $e_x, e_{x+1}, \cdots,\ e_y$ were to take place in **any order**. The size of a clowder is the number of cats belonging to it.

Please note that each query is independent i.e. one query neither impacts other queries, nor the original clowders.

Input

The first line contains a single integer $T (1 \leq T \leq 10)$ - the number of test cases. T test cases follow.

The first line of each test case contains a positive integer $N (1 \leq N \leq 10^5)$ - the total number of cats. The following line contains N **uppercase English letters** without any spaces between them. The i'th letter represents the clowder c_i of the i'th cat.

The next line contains a positive integer $M\ (1 \leq M \leq 10^5)$ - the total number of events. M lines follow, i'th line contains two integers L_i *and* R_i representing event $e_i\ (1 \leq L_i \leq R_i \leq N)$.

The next line contains a positive integer $Q\ (1 \leq Q \leq 10^5)$ - the total number of queries. Q lines follow, i'th line contains two integers x_i *and* y_i representing a query.

It is guaranteed that, $\Sigma N \leq 10^5, \Sigma M \leq 10^5, \Sigma Q \leq 10^5$ across all test cases.

Output

For each test case, print the case number in one line in the format "**Case T:**", where T is the test case number. Then for each query, output the result (maximum possible size of a clowder) in a single line.

Sample Input	Output for Sample Input

`1`
`12`
`AABCBBDCEEEE`
`4`
`1 12`
`1 3`
`3 4`
`7 8`
`4`
`1 1`
`2 2`
`2 3`
`4 4`
	`Case 1:`
	`12`
	`5`
	`7`
	`4`

It is recommended to use fast input/output methods due to the large input size.

Explanation

In the sample I/O above –

- for query 1 ($[1, 1]$), it is optimal for event 1 to take place. As a result, all of the cats belong to a single clowder.
- for query 2 ($[2, 2]$), it is optimal for event 2 to take place. Clowders *A* and *B* form a new clowder of 5 cats.
- for query 3 ($[2, 3]$), it is optimal for all events to take place sequentially. In event 2, clowders *A* and *B* form a new clowder *X* of 5 cats. In event 3, clowders *X* and *C* form a new clowder *Y* of a total 7 cats.
- for query 4 ($[4, 4]$*),* we can choose not to have any events at all. If event 4 was to take place, clowders *C* and D would form a new clowder of 3 cats whereas clowder *E* already has 4 cats.

Problem E

Quasi-binary Representations

Every positive integer has a unique binary representation using only the digits 0 and 1. However, if we allow digits other than 0 and 1, a number may have multiple valid representations.

For example, the number 4 can be represented in the standard binary system as **100**. However, if the digit **2** is allowed, it can also be represented as **20** or **12**. Here:

- **20** represents $2 \cdot 2^1 + 0 \cdot 2^0 = 4 + 0 = 4$
- **12** represents $1 \cdot 2^1 + 2 \cdot 2^0 = 2 + 2 = 4$

These are called **quasi-binary representations** of a number. Note that, as usual, we do not allow leading zeros in the representation, so, **020** is not a valid representation.

You are given n and k. Your task is to find the number of distinct quasi-binary representations of n using digits from $\{0, 1, 2, \cdots, k\}$. Since this number can be large, you need to output its value modulo 998244353.

Input

Input starts with an integer T — the number of test cases.
Each test case is described by a single line containing two integers, n and k.

Constraints

- $1 \le T \le 1000$
- $1 \le k \le 1000$
- $1 \le n \le 10^{18}$

Output

For each test case, print one line containing a single integer, the number of quasi-binary representations modulo 998244353.

Sample Input	Output for Sample Input
3 4 2 3 3 10 10	3 2 14

Problem F

Flowers

In this problem, You will be given a forest. This forest consists of one or multiple directed binary trees. Nodes of the tree can be black or white. You will reconstruct the forest by executing some queries. Note that, each edge directs the relation from parent to child and each node will always have at most one parent. When asked a query, determine the number of **unique flower shapes** in a tree.

Definitions

- **Tree Root:** A node with no parent.
- **Flower Root:** A white node with no parent or a black parent.
- **Flower:** A flower consists of a **Flower Root** and all the white nodes that can be reached from the flower root via directed edges, with no black nodes appearing along the path.
- **Unique Flower Shapes**: Two **Flowers** are considered to have the same unique shape if they are structurally identical, meaning:
  - Both flowers have the same number of nodes.
  - The arrangement of left and right children at every node is the same.

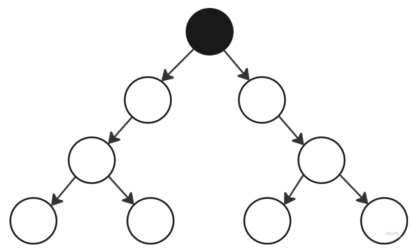

Figure: The flowers are in different shapes.

There will be N nodes in the forest. For each node in the forest, you will be given three pieces of information (Left child, Right Child, and Color of that node). After that, there will be Q instructions. Each of the instructions can be any of the two types below. After each query, the vertices will not break the directed binary tree property.

- $1\ X\ Y$
  Let, R_X = The root of the tree containing the node X, and L_Y = The **leftmost leftless** node of the tree containing the node Y. Add R_X as the left child of L_Y. It is guaranteed that X and Y are not in the same tree.
  The leftmost leftless node in a directed binary tree is the first node encountered during a pre-order traversal (starting at the root and prioritizing the left child first, then the right child) that does not have a left child. If the tree has only one node, that node is the leftmost leftless node.
- $2\ X$
  Determine the number of unique flower shapes in the tree containing the node X.

Input

The first line will contain a single integer $T(1 \le T \le 5\times10^5)$ denoting the number of test cases.
Each test case will start with two integers $N(1 \le N \le 5\times10^5)$ and $Q(1 \le Q \le 5\times10^5)$.

The next N lines will contain three space-separated integers L_i, R_i & $C_i(C_i \in \{0, 1\})$. It is guaranteed that with this information, a valid forest with valid directed binary trees can be formed. L_i or R_i is -1 if the i^{th} node doesn't contain the respective child node. Otherwise, they follow the constraints $(1 \le L_i, R_i \le N,\ L_i \ne R_i)$. $C_i = 0$ means the i^{th} node is colored black and $C_i = 1$ denotes the node is colored white.

After building the trees, there will be Q instructions. Each of the instructions can be any of the two types below.

- $1\ X\ Y$
  Constraints: X and Y are integers. $(1 \le X, Y \le N, X \ne Y)$
- $2\ X$
  Constraints: X is an integer. $(1 \le X \le N)$

Note that, the sum of N across all test cases is at most 5×10^5, and the sum of Q across all test cases is at most 5×10^5. It is guaranteed that there will be at least one query of type 2 in each test case.

Output

For each test case, first print "**Case T**:" on a separate line, where T is the test case number. Then, for each query of type 2, print the corresponding answer on a new line.

Sample Input	Output for Sample Input
2 9 7 2 3 0 -1 -1 1 4 -1 1 -1 -1 1 6 7 0 -1 -1 1 8 -1 1 -1 -1 1 -1 -1 1 2 1 2 5 2 9 1 9 1 2 1 1 5 1 2 5 1 1 -1 -1 0 2 1	Case 1: 2 2 1 1 2 Case 2: 0

It is recommended to use fast input/output methods due to the large input size.

Problem G

Library Function vs Keyword

In C there is a keyword $sizeof$ which can be used to find how much memory space (In bytes) a string variable occupies. There is another function $strlen()$ which finds the length of a string. But there is a difference between how these two work. $sizeof()$ adds a NULL character at the end of **any** string and calculates memory requirement using that, on the other hand $strlen()$ calculates the length up to the first null character but excludes null characters from string length. The output of the program below will make the difference clear.

Code:

```
#include<stdio.h>
#include<string.h>
char line[]="abracadabra\0abraka";
int main(void)
{
    printf("%d %d\n",sizeof(line),strlen(line));
    return 0;
}
```

Output Console:

```
19 11

Process returned 0 (0x0)   execution time : 0.094 s
Press any key to continue.
```

The assigned string here is **abracadabra**\0abraka, $strlen()$ shows the length of the boldfaced part and $sizeof()$ shows the memory required by the whole string and an additional $null$ character.

Given the value of the assigned string to variable $line[]$, you will have to print the value of $sizeof(line)$ and $strlen(line)$.

Input

The first line of the input file contains a positive integer $N (N < 1000)$ which denotes the total number of input lines. Each of the next lines contains a non-empty string S which is the value of the assigned variable $line[]$. This string only contains alphabets ('A'- 'Z', 'a'- 'z') and $null$ (Denoted with '\0') characters. There are a maximum of 100 characters in this string assuming '\' and '0' are counted separately.

Output

For each line of input produce one line of output which denotes the values of $sizeof(S)$ and $strlen(S)$ respectively.

Sample Input	Output for Sample Input
1 **abracadabra\0abraka**	**19 11**

Problem H

Hand Cricket

Alice and Bob play a game on an integer array A of length N. Alice chooses a secret index i, and Bob chooses a secret index j. If $i \neq j$ then Alice gains A_i points, else she gains 0 points. Then the game ends.

Alice wants to maximize the point. Bob wants to minimize it. A *strategy* for a player is a probability distribution $P_1, P_2, \ldots, P_N$ ($P_1 + P_2 + \ldots + P_N = 1,\ 0 \leq P_i \leq 1$) which implies that the player will choose index i with probability P_i. Both of them will choose their own *strategy* in such a way that their *strategy* cannot be exploited, even if it is leaked to the other person.

Answer Q queries, each with a given tuple $[L, R, K]$:
For each query:

- The game will be played within the subarray $A_L, A_{L+1}, \ldots, A_R$.
- Before the game starts, Alice can apply at most K increment moves. An increment move is to do $A_i = A_i + 1$ for some $i \in [L, R]$. The move can be applied multiple times on the same i. The increment moves only affect the current query and will not persist to subsequent queries.

Print the expected number of points gained by Alice. The answer can be expressed as an irreducible fraction $\frac{X}{Y}, Y \neq 0$. You need to print the value of $(X \cdot Y^{-1})\ modulo\ 998244353$. It is guaranteed that for each of the queries an answer can be calculated (i.e Y^{-1} exists modulo 998244353).

Input

The first line contains an integer $N (1 \leq N \leq 2 \times 10^5)$. The second line has N space-separated integers $A_1, A_2, \ldots, A_N$ $(1 \leq A_i \leq 10^8)$. The third line contains an integer Q $(1 \leq Q \leq 1 \times 10^5)$, the number of queries.

The following Q lines contain 3 integers each, L_i, R_i, K_i $(1 \leq L_i \leq R_i \leq N,\ 0 \leq K \leq 10^8)$ representing the i^{th} query.

Output

Print the answer of each query in a separate line.

Sample Input	Output for Sample Input
3 1 2 3 3 1 3 3 1 1 6 1 2 2	2 0 598946613

Problem I

In Search of a Kind Person

There was a long queue of people waiting to enter into the Victory Day Street Fair. The main show had already begun, and everyone in the queue was trying to take a peek inside. While some were anxious to get in quick; there were a few, considerate enough to think about others even in this rush. They found perfect positions for themselves, without blocking the view to anyone standing behind them. We want to meet such a person and say, ‘Thank you!’

So your job is to find the person who is strictly taller than everyone standing in front of him, and strictly shorter than everyone behind.

Input

The first line tells the number of test cases with a single integer $T (1 \leq T \leq 10^5)$.
Each test starts with an integer $N (1 \leq N \leq 10^5)$, in a line, denoting the length of the queue. The following line contains a list of N integers $A_1, A_2, \ldots, A_N$ $(1 \leq A_i \leq 10^9, i = 1..N)$, describing the height of each person in the queue. A_1 is the height of the person standing at the front of the queue, and A_N describes the person at the very back.

It is guaranteed that $\Sigma N \leq 10^6$ over all test cases.

Output

For each case, print a single line containing the case number, and an integer **i**, the position of the person we are looking for. If there are multiple, find the frontmost one. If there is no such person, print “**Humanity is doomed!**” without quotes.

Check the format below.

Sample Input	Output for Sample Input
4 5 2 1 3 5 4 8 2 1 2 5 5 7 8 9 2 2 1 5 1 2 3 4 5	Case 1: 3 Case 2: 6 Case 3: Humanity is doomed! Case 4: 1

It is recommended to use fast input/output methods due to the large input size.

Problem J

The Taxman

You are the Royal Taxman of the prosperous kingdom of Luxandra, tasked by King Alexander IV to design a progressive tax system for its citizens. The kingdom relies on your mathematical precision to ensure a just and fair taxation process that supports the royal treasury without burdening its hard working citizens.

King Alexander IV has decreed that the tax system will be divided into several bands, with wealthier citizens contributing more to the kingdom's prosperity. The people of Luxandra earn their income in Luxans (LX), the official currency of the kingdom.

The Kingdom's Tax System: The tax system for the year 1424 is outlined as follows:

Tax Band	Income Range (LX)	Tax Rate (%)
Personal allowance	0 to 12,500	0%
Basic rate	Over 12,500 up to 50,000	20%
Higher rate	Over 50,000 up to 150,000	40%
Additional rate	Over 150,000	45%

Royal Decrees:

1. Personal allowance: Every citizen enjoys a tax-free personal allowance of first 12,500 Luxans (LX). No tax is paid on this portion of their income.
2. Basic rate (20%): The next portion of the income over 12,500 up to 50,000 LX is taxed at 20%.
3. Higher rate (40%): The next portion of the income over 50,000 up to 150,000 LX is taxed at 40%.
4. Additional rate (45%): The next portion of the income above 150,000 LX is taxed at 45%.
5. Reduction of personal allowance: For any amount earned above 100,000 LX, the personal allowance is reduced by half of that amount, but the personal allowance can't go below zero (e.g., earning 100,001 LX reduces the allowance by 0.5 LX).
6. Taxing the reduced personal allowance at the higher rate: Once a citizen's salary exceeds 100,000 LX, and their personal allowance is reduced, the reduced portion of the allowance will be taxed at 40% (the higher rate). For example, earning 100,002 LX reduces the allowance by 1 LX, and this 1 LX is taxed at 40%.

After designing and implementing the tax system, you calculated the total tax owed by each citizen. Unfortunately, the kingdom has since lost all records of citizen incomes for the year 1424. However, the tax amounts collected from each citizen were securely stored.

The king has now given you the task of reconstructing the income records for each citizen based on their tax amounts. You must write a program that calculates the yearly income of a citizen from the tax amount provided.

Input

The first line contains an integer T, the number of test cases. Each of the next T lines contains a floating-point number X, denoting the tax amount collected from a citizen in Luxans (LX), formatted to two decimal places.

Constraints

- $1 \le T \le 10^4$
- $0 \le X \le 10^9$

Output

For each test case, output a **non-negative** real number representing the calculated yearly income of the citizen. If there are multiple solutions, you can output any of them.

Note that this problem uses a special judge. Your output will be considered correct if the absolute or relative error between the actual tax and the tax calculated from your output income does not exceed 10^{-6}. Formally, if the tax calculated from your output income is A and the actual tax provided in the input is B, your answer will be accepted if and only if $\frac{|A-B|}{max(1, B)} \le 10^{-6}$.

Sample Input	Output for Sample Input
4 500.50 8028.40 33504.20 75000.00	15002.50 51321 110007.00 200000.0000

Explanation

Example 1: The yearly income is 15,002.50 LX. Here the first 12,500 LX is tax-free personal allowance. And the rest of the income, 2,502.5 LX is taxed at basic rate (20%).

Example 2: The tax calculation for income amount 51,321 LX is explained below -

- The first 12,500 is tax free personal allowance.
- For income between above 12,500 and 50,000, it falls in the basic rate and is taxed at 20%. So the citizen paid 7,500 LX in tax for this range.
- The remaining 1,321 LX falls in the additional rate and is taxed at 40%. So this citizen paid 528.4 LX for this range. So the total tax is 7,500 + 528.4 = 8,028.40 LX.

Example 3: Since this citizen earned 10,007 LX over 100,000, their personal allowance is reduced by half of 10,007 and taxed at 40%. So the extra tax here is 2,001.40 LX.

Problem K

Packet Transmission

You are given a connected network **tree** representing routers and channels. Each node represents a router, and each edge represents a channel connecting two routers. These channels are used to transmit packets from one router to one of its adjacent routers, and the transmission follows the TCP/IP protocol. The edges are labeled with t_i indicating the time (in arbitrary units) it takes to transmit a packet from u_i to v_i or vice versa.

Additionally, each router has enough storage capacity to store any number of packets, they can store packets there indefinitely and they can transmit multiple packets simultaneously but via different channels. This means that you can't send a packet via a channel that is currently occupied by another transmission.

Your task is to send two packets concurrently, one from s_1 router to d_1 router and another from s_2 router to d_2 router. Determine the optimal transmission strategy to minimize the maximum time for delivering both of these packets from source to destination.

Input

Input starts with an integer $T(1 \le T \le 1000)$ denoting the number of test cases. Each of the test cases starts with two integers N and Q $(1 \le N, Q \le 100000)$. Then there will be $N - 1$ lines of input consisting of three integers $u,\ v,\ t\ (1 \le u, v \le n, u \ne v,\ 1 \le t \le 10^9)$ representing edges of the **tree**.

Then, There will be Q lines each with four integers $s_1, d_1, s_2, d_2 (1 \le s_1, d_1, s_2, d_2 \le N)$ representing the source and destination routers for two packets. $\Sigma N \le 500000,\ \Sigma Q \le 500000$ holds across all test cases.

Output

For each test case print the test case number in the format of “**Case T:**” in one line where T is the test case number. Then for each query use the optimal transmission strategy to minimize the maximum time to deliver both packets and print the minimum time in a separate line.

Sample Input	Output for Sample Input
1 8 3 1 2 3 2 3 5 2 6 7 3 4 4 4 5 3 6 7 2 1 8 6 1 8 2 6 1 5 2 3 5 8 8 5	Case 1: 7 15 24

It is recommended to use fast input/output methods due to the large input size.

Problem L

Unhappy Team

You are managing a team of N people numbered from 1 to N. In the team everyone perceives everyone else as stronger or weaker than them in work performance. For the year end bonus you need to stack rank the team in some way. A stack rank is a permutation of the numbers from 1 to N. After the stack ranking is done, it is visible to everyone in the team. Everyone has an unhappiness score which is equal to the number of people ahead of them in the stack rank that they deemed weaker than them. If we find the unhappiness score of everyone and sum up the largest or worst K unhappiness score then we find the score of that stack ranking. For example if the ascendingly sorted unhappiness score of a stack ranking is $[0, 1, 2, 2, 3]$ then for $K = 2$ the score is $5(3 + 2)$ and for $K = 3$ the score is $7(3 + 2 + 2)$.

Given N, K and everyone's perception of who is stronger or weaker than them, find the expected unhappiness score of a randomly chosen stack ranking. The answer can be expressed as $\frac{P}{Q}, Q \neq 0$, then answer the value of $(P * Q^{-1})\ modulo\ 998244353$.

Input

The first line of the input contains an integer $T(1 \leq T \leq 50)$, denoting the number of test cases. For each test scenario, the first line contains two integers N and K $(1 \leq K \leq N \leq 16)$. The next N lines each contain N uppercase characters. The character j in line i represents what person i thinks of person j. If the character is 'S' then i thinks j is stronger. If it is 'W' then i thinks j is weaker. It will be 'X' if $i = j$. It is guaranteed that the number of test cases where $N > 12$ is at most 3.

Output

For each test scenario print one line, the expected score of a randomly chosen stack ranking modulo 998244353.

Sample Input	Output for Sample Input
3 3 2 XSW SXW WSX 4 3 XSWS WXSW SSXS SWWX 1 1 X	499122178 499122179 0