A

Palentine's Day

Input: Standard Input
Output: Standard Output

Palentine was the great mathematician in Rome city. He loved to use mathematical formulas in his everyday life. One day he met a girl and decided to propose her. Now it was time to impress the girl. Suddenly a thought came into his mind. He decided to give her some flowers, each day when he would meet the girl. But it was hard to decide for Palatine that how many flowers he would buy. So he made an equation to evaluate the number of flowers.

Palentine decided to give **1+2+…+N** flowers on N-th Palentine day (Since Palentine does not believe in year and month, he started his own day counting system starting from 1 to infinity).

So according to the equation he gave 1 flower to the girl on the 1st Palentine's day. As such he gave 1+2 = 3 flowers to her on the 2nd Palantine's day, 1+2+3+ … + 9 + 10 = 55 flowers on the 10-th day and so on.

The days were passing, and Palantine was busy with calculating his number and buying flowers. Now Palentine is too tired and bored to calculate the number of flower to buy and today is the 365-th Palantine's day. He needs you to calculate the number of flowers he is going to buy on 365-th day.

Input

There is no input in this problem.

Output

Output a single integer number, the number of flowers he should buy on 365-th day.

Sample Input	Output for Sample Input
	5151

Note: The sample output might give wrong answer as this is a sample to show the output format.

Hint: Code snippet in C/C++ to print the above sample output

```
#include<stdio.h>
int main()
{
    int N = 365, flower;
    //flower = something;
    printf("%d\n",flower);
    return 0;
}
```

B	Special Days **Input:** Standard Input **Output:** Standard Output

Special Days

Input: Standard Input
Output: Standard Output

People loves to celebrate special days with or without any reason and number of these special days are increasing day by day. For example, Valentine’s Day, Rose Day, Propose Day, Dance Day etc are some of those. Today your will be given a list of special days and your job is to find out name of the next special day from the given list.

Pohela Falgun	February 13
Valentine's Day	February 14
International Mother Language Day	February 21
International Women's Day	March 8
International Color Day	March 21
Bangladesh Independence Day	March 26
April Fools Day	April 1
Pohela Boishakh	April 14
May Day	May 1
Father's Day	June 19
Mandela Day	July 18
World Mosquito Day	August 20
Halloween	October 31
World Toilet Day	November 19
Human Rights Day	December 10
Victory Day of Bangladesh	December 16
Christmas Day	December 25

For example, if today is “February 16”, then next special day is “International Mother Language Day” on February 21.

Input

First line of the input will contain a single integer T (<= 100), denoting the number of test cases to follow.

Each test case consists of a string M and D, where M is the current month name (January, February, March, April, May, June, July, August, September, October, November, December) and D is the day of the month. The date given will always be a valid date (considering there is no leap year).

Output

Print the test case number and name of the next special day.

Sample Input	Output for Sample Input
3 February 16 December 10 December 31	Case 1: International Mother Language Day Case 2: Victory Day of Bangladesh Case 3: Pohela Falgun

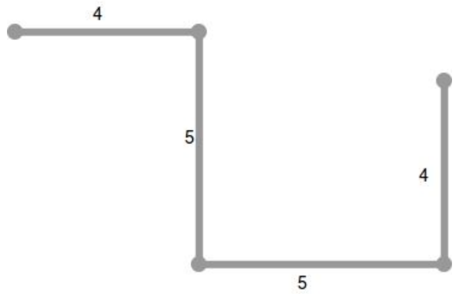
C

Bulb and Pipes

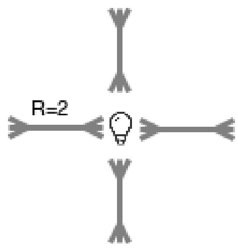
Input: Standard Input
Output: Standard Output

HackerLand is building a new tunnel system. The tunnel can be viewed as a combination of one or more connected horizontal and vertical pipes. Each pipe has the same width but may have different lengths.

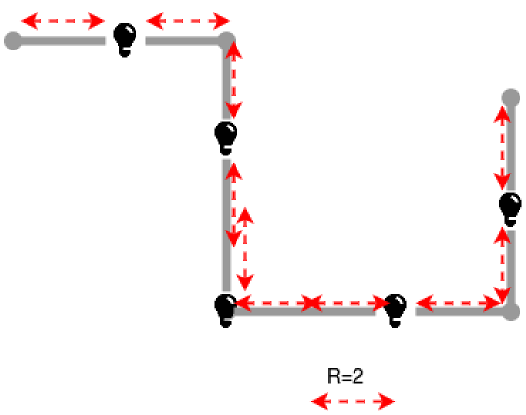
To make the problem simpler, we can consider each tunnel as a straight line. See the image below:



Mayor of HackerLand wants to place light bulbs in the tunnel. Scientists of HackerLand made a special light bulb that has a range **R**. It can illuminate the tunnel up to **R** meters in 4 directions (up, down, left and right).



Now the mayor wants to enlighten the whole tunnel using the minimum number of bulbs. So if R=2 and you need at least 5 bulbs for example above. One possible placement is shown in the image below:



You can assume that the pipes won't intersect. Given the configuration of the pipes and the range of the bulb, can you find the minimum number of bulbs needed?

Input

The first line contains an integer **T (1<=T<=2000)** denoting the number of test cases. Each test case contains three lines. The first line contains two space-separated integers **N (1<=N<=2000)** and **R (1<=R<=50)** where N denotes the number of pipes in the tunnel. The next line contains N space-separated integers **L_i(1<=L_i<=1000)**, the i^{th} integer denotes the length of the i^{th} tunnel. The next line contains N space-separated characters. Each character is either 'R', 'U' or 'D' denoting the direction (right, up, down) of the i^{th} tunnel.

Note that, two adjacent tunnels can have same directions. But two pipes will never overlap (there won't be an adjacent U and D in the input).

Output

For each test case, print the case number and the minimum number of bulbs needed.

Sample Input	Output for Sample Input
5 4 2 4 5 5 4 R D R U 2 2 5 5 R R 5 3 6 4 5 7 10 R D R U R 5 1 6 4 5 7 10 R D R U R 3 4 10 20 15 R D R	Case 1: 5 Case 2: 3 Case 3: 7 Case 4: 16 Case 5: 6

D

Sum of Two Sequences

Input: Standard Input
Output: Standard Output

Meera wants to become a Data Scientist. In order to learn the way of data analysis, she sought help from the great master Data Scientist, Zinnah the Analyzer. Under the supervision of Zinnah, Meera solves various analytical problem to improve her skills.

The problem Meera is solving today is called "Tale of Two Sequence : Sum of All".

In this problem, Meera is given two arithmetic sequence by Zinnah. For each sequence, Meera is given the first term **A**, the difference between two consecutive values **D** and number of terms **K**.

Meera now must find the sum of all terms **S,** when both arithmetic sequences are considered together. That is, Meera needs to find $\mathbf{S = s_1 + s_2}$, where s_1 is the sum of all terms in first sequence and s_2 is the sum of all terms in second sequence.

For example, suppose we are given $A_1 = 1$, $D_1 = 1$, $K_1 = 10$ and $A_2 = 15$, $D_2 = -3$, $K_2 = 4$, then $S = 97$.

How? $A_1 = 1$, $D_1 = 1$, $K_1 = 10$ represents the sequence [1, 2, 3, 4, 5, 6, 7, 8, 9, 10] and $A_2 = 15$, $D_2 = -3$, $K_2 = 4$ represents [15, 12, 9, 6]. If we consider both sequence together then we find that the sum is ((1+2+3+4+5+6+7+8+9+10) + (15+12+9+6)) = 97.

Zinnah believes that Meera will be able to solve the problem eventually since she is a smart girl. Meanwhile, he decided to send the same problem to today's programming contest, for all of you to solve. Have fun :).

Input

First line of the input will be a single integer T, indicating the number of test cases. Next T lines will follow, containing 6 integers, representing A_1, D_1, K_1, A_2, D_2 and K_2.

Constraints

$1 <= T, K_1, K_2 <= 10^5$

$-10^5 <= A_1, D_1, A_2, D_2 <= 10^5$

Output

For each test case, output the test case number and a single integer value which is the sum of all terms that occur in the two sequences. See sample input/output for details.

Sample Input	Output for Sample Input
2 1 1 10 15 -3 4 1 2 10 1000 500 3	Case 1: 97 Case 2: 4600

E	Coin and Bulbs
	Input: Standard Input **Output:** Standard Output

Coin and Bulbs

This is yet another boring problem about a coin and some bulbs, if you feel bored you should skip this problem!

There are n bulbs. Initially, each bulb are turned off. In each move you can select one random bulb (Probability of selecting each bulb is same). If the bulb is already turned on, you do nothing. If the bulb is turned off, you must toss a coin. If it's head, you can turn the bulb on, but if it's tail, the bulb will remain off.

To make the problem even more boring, the coin is not a fair coin, the chance of landing a tail is p%.

What is the expected number of moves needed to turn on all the bulbs?

Input

The first line will contain T (1<=T<=5000) denoting the number of test cases. For each case, there will be two integers n (1<=n<=40) and p(0<=p<=99).

Output

For each case, print the case number and the expected number of moves as a reduced fraction p/q;

Sample Input	Output for Sample Input
4 1 50 2 50 2 0 5 20	Case 1: 2/1 Case 2: 6/1 Case 3: 3/1 Case 4: 685/48

F	**Grundy and K-Nim** **Input:** Standard Input **Output:** Standard Output	CUET NCPC 2017

Little Grundy loves the game of Nim. He is always challenging other boys and girls of his age to play the game. He is very good at it and always wins (well not always, because there are some games, which you can't win unless you are playing with an idiot).

Those who doesn't know the game of Nim, it's played between two players. The game can have several piles of stones. The players take turns to give their moves. In each move, a player can choose a non-empty pile and remove any positive amount of stones from the pile. The game ends when there are no stones left to remove. The player who can't make a move at her turn loses the game. For more details about the game you can visit: https://en.wikipedia.org/wiki/Nim

Now Grundy knows everything about traditional Nim. So to keep things interesting he is making variations of the game. One such variation is that instead of removing any positive number of stones, at each turn you can remove at most K stones from the chosen pile. Lets call this game K-Nim. Grundy always plays first. Given the pile sizes and the value of K for a K-Nim game, you need to determine if the given game can be won by Grundy. Each player plays optimally, meaning they will not make a move that causes them to lose the game if some better, winning move exists.

Input

Input starts with an integer, T (T <= 10), number of test cases. Each case starts with N (0 < N < 1001), the number of piles. Next line contains N space separated integers Pi (0 < Pi <= 10^9), which denotes the size of the ith pile. Next line will have an integer Q (0 < Q < 1001), the number of queries. Q space separated integers will follow denoting the queries. Each query will contain an integer K (0 < K <= 10^9). For each query, you should assume that the K-Nim game consists of the piles given in the case and the value of K is given in that query. You need to find out the winner of the game for each query.

Output

Output case no for each case. Then add query no for each query. For each query mention "Win" or "Lose" based on the outcome of the game for little Grundy. See sample for clarification.

Sample Input	Output for Sample Input
1 3 3 5 5 2 2 4	Case 1: Query 1: Lose Query 2: Win

G	Merge the Strings **Input:** Standard Input **Output:** Standard Output

Merge the Strings

Input: Standard Input
Output: Standard Output

You are given two strings **A** and **B** which you are required to merge into a single string **C**. To merge **A** and **B**, we need to find some of their common non-overlapping substrings. After that, we need to group those common substrings and the remaining uncommon substrings separately and merge these groups to form **C**. The merging needs to be done in such a way that the order of the characters of **A** and **B** will remain the same in **C**. This means that both **A** and **B** will be subsequences of **C** and it will not contain any extra characters which aren't present in either **A** or **B**. We can indicate the common substrings, the substrings which only originate from **A** and the substrings which only originate from **B** by surrounding them with **()**, **{}** and **[]** respectively.

For example, if **A = BABBA** and **B = BBBAC** one possible way to get common non-overlapping substrings would be **(B)** and **(A)**, so in their grouped form **A = (B)(A){BBA}** and **B = (B)[BB](A)[C]**. A possible merge can be **C = BBBACBBA** with its grouped form being **(B)[BB](A)[C]{BBA}**.

Another possible way to get common non-overlapping substrings would be **(B)** and **(BBA)**, so in their grouped form **A = (B){A}(BBA)** and **B = (B)(BBA)[C]**. The only possible merge for this is **C = BABBAC** with its grouped form being **(B){A}(BBA)[C]**.

Note that even though **A** and **B** were the same, two different **C**s were obtained with the latter one having smaller length. Your task is to find such a **C** which has the smallest length and group the characters so that you can tell which character originates from which string. You need to surround a substring of **C** with **()**, **{}** or **[]** if it originates from both **A** and **B**, only **A** or only **B** respectively.

If you obtain multiple **C** of the smallest size, find the one which is lexicographically smallest after grouping the characters. For this problem, assume that **'A' < … < 'Z' < '(' < ')' < '{' < '}' < '[' < ']'**.

For example, if **A = BABBA** and **B = BBBAC**, you will obtain **C** = **BABBAC** which has the minimal length **6** and will have the grouped form of **(B){A}(BBA)[C]**.

Note:

A subsequence **R** of a string **S** is a sequence which is obtained by removing zero or more characters from **S**.

A substring **R** of a string **S** is a continuous block of characters of **S** in the inclusive range from index **l** to **r**, where $\mathbf{R = S_l S_{(l+1)} S_{(l+2)} \ldots S_r}$ and **1 <= l <= r <= |S|**.

Input

The first line of the input contains a single integer **T**, which denotes the number of test cases. This is followed by the test cases.

Each case consists of two lines which contain **A** and **B** respectively.

Constraints

- **1 <= T <= 30**
- **1 <= |A|,|B| <= 50**
- **All characters are uppercase letters.**

Output

For each test case, output the case number, followed by the length of **C** and it's grouped form, each in it's own separate line.

Again, if you obtain multiple **C** of the smallest size, output the grouped form which is lexicographically smallest. For this problem, assume that **'A' < … < 'Z' < '(' < ')' < '{' < '}' < '[' < ']'**.

See the sample input/output for more clarification.

Sample Input	Output for Sample Input
3	Case 1:
BABBA	6
BBBAC	(B){A}(BBA)[C]
ABC	Case 2:
ABC	3
ABC	(ABC)
XYZ	Case 3:
	6
	{ABC}[XYZ]

H	**Count Clog** **Input:** Standard Input **Output:** Standard Output	

Given **N** distinct integers from 1 to **N**, you have to find the number of ways the **N** integers can be rearranged in **M** empty slots such that, no integer matches with its slot index. Note that, slots are indexed from 1 to **M**.

For example, if N = 3 and M = 5, then here is a possible arrangement:

2		**1**	**3**	

Here **2** is placed in slot 1, **1** is placed in slot 3 and **3** is placed in slot 4. Slot 2 and 5 are kept empty.

Input

An integer T <= 200, the number of test cases. Next T lines will contain two space separated integers N and M.

Constraints:
0 < N <= M <= 100,000

Output

Print the number of ways modulo **23377788**.

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

Explanation:

Let us consider 0 as blank space and check the value for sample input.

1 2 3 (m=3 positions)
1 2 0 (invalid)
1 0 2 (invalid)
2 1 0 (valid)
2 0 1 (valid)
0 1 2 (valid)
0 2 1 (invalid)

I	Pom Gana **Input:** Standard Input **Output:** Standard Output	

This year the ACM ICPC World Final will be hosted by the country of Gana, which is popular for its weird road system. Gana consists of N city and M roads. All the roads in the country are unidirectional. That's why people often get stuck in some dead end cities and had to use a helicopter to return from there. In Gana, those cities are called Pom city. A city X is a Pom city if a person starts his/her journey from X, he/she cannot return to X again using one or more roads.
So the ACM ICPC headquarter wanted to know how many Pom city are there in Gana.

Input

First line of the input contains a single integer T (<=100), the number of test cases to follow.
First line of each test case contains space- separated integers N and M (1<=N<=100, 0<=M<=N*N), the number of cities and roads respectively. Next M lines will contain two space-separated integers U and V (1<=U,V<=N, U!=V), such that there is a unidirectional road from city U to city V.

Output

For each test case, print the test case number and number of Pom city in Gana.

Sample Input	Output for Sample Input
2 4 4 1 2 2 3 3 1 1 4 4 5 1 2 2 3 3 1 1 4 4 1	Case 1: 1 Case 2: 0