Algospot.com 4th Anniversary Contest

September 23, 2011

Problems A to J, Total 19 pages

참가자를 위한 도움말

주의 사항

- 모든 입력은 **표준 입력**으로 주어지며, 모든 출력은 **표준 출력**으로 합니다. 표준 입/출력의 사용 방법에 대해서는 아래의 각 언어별 도움말을 참조해 주십시오.
- 모든 테스트 케이스에 대한 출력을 모아서 하실 필요 없이, 각 테스트 케이스를 처리할 때마다 출력해도 괜찮습니다. 역시 아래의 각 언어별 도움말을 참조해 주십시오.
- **리턴 코드** 에 주의하십시오. 프로세스가 0 이 아닌 다른 리턴 코드를 되돌리는 경우 **No – Runtime Error** 를 받게 됩니다.
- 파일 이름을 영문자와 숫자로만 구성하십시오. 특히 **공백**, **특수 문자**, **한글**의 경우는 정상적인 채점을 보장할 수 없으며, 채점이 정상적으로 이루어지지 않은 경우 **"No – Compilation Error"** 를 받게 됩니다.
- 대회 도중에 ACM-ICPC 에서 정하는 규정에 따라 **인터넷 검색**, bison이나 yacc와 같은 **자동 코드 생성 도구 사용** 등의 행위를 삼가해 주시기 바랍니다.
- 문제에 대한 질의 사항은 채점 시스템인 PC^2의 **Clarification** 기능을 사용해 주시기 바랍니다. 이 때 대답해 드리기 어려운 질문에 대해서는 **"No response, read problem statement"** 로 대답될 수 있으므로 유의하십시오. 대답해 드리기 어려운 질문의 예는 아래와 같습니다:
  - 테스트 케이스의 수는 몇 개나 되나요
  - 시간제한은 얼마나 되나요
  - 제 솔루션이 왜 WA인가요
  - 문제에서 이러이러한 사항을 임의로 가정해도 되나요
  - 제 시스템에서는 컴파일이 잘 되는데 왜 **"No – Compilation Error"** 를 받나요: 위에서 언급한 파일 이름 규칙을 위반하였거나, 혹은 대회에서 지정한 컴파일러 버전과의 불일치일 수 있습니다. **특히 Visual C++ 6.0을 사용하시는 경우 빈번히 문제가 됩니다.**

채점 결과에 대하여

Yes 제출하신 답안이 모든 테스트 데이터를 정해진 시간 안에 통과하여 정답으로 인정되었음을 의미합니다.

No – Compilation Error 제출하신 답안 프로그램을 컴파일하는 도중 오류가 발생하였음을 의미합니다.

No – Run-time Error 제출하신 답안 프로그램을 실행하는 도중 프로세스가 비정상적으로 종료되었음을 의미합니다.

No – Time-limit Exceeded 제출하신 답안 프로그램이 정해진 시간 안에 종료되지 않았음을 의미합니다. 시간 제한은 공개되지 않습니다.

No – Wrong Answer 제출하신 답안 프로그램이 테스트 데이터에 대해 생성한 출력이 출제자의 정답과 일치하지 않음을 의미합니다.

No – Excessive Output 제출하신 답안 프로그램이 지나치게 많은 양의 출력물을 생성하여 강제로 종료되었음을 의미합니다.

No – Output Format Error 제출하신 답안 프로그램이 정해진 출력 형식을 따르지 않았음을 의미합니다.

No – Other – Contact Staff 위에서 나열한 이유 이외의 문제로 채점이 거부되었음을 의미하며, Clarification 을 통해 대회 본부에 문의해 주십시오.

만약 여러 가지의 원인으로 인해 "**Yes**" 가 아닌 다른 결과를 얻는 경우, 그 중 어떤 것도 결과가 될 수 있습니다. 예를 들어 답도 잘못되었고 출력 형식도 잘못된 코드를 제출하신 경우 대부분 "**No – Output Format Error**" 를 받으시게 되지만, 경우에 따라서 "**No – Wrong Answer**" 를 받을 수도 있습니다.

언어별 도움말

C/C++

다음 두 예시 코드는 먼저 테스트 케이스의 개수를 입력받고, 각 테스트 케이스에 대해 두 정수를 입력받아 더한 값을 출력합니다.

```c
// Written in C
#include <stdio.h>

int main()
{
        int t;
        scanf("%d", &t);
        for (int i = 0;i < t;i++)
        {
                int a, b;
                scanf("%d␣%d", &a, &b);
                printf("%d\n", a + b);
        }
        return 0;
}
```

```cpp
// Written in C++
#include <iostream>

using namespace std;

int main()
{
        int t;
        cin >> t;
        for (int i = 0;i < t;i++)
        {
                int a, b;
                cin >> a >> b;
```

```
                cout << a + b << endl;
        }
        return 0;
}
```

Java

다음 예시 코드는 먼저 테스트 케이스의 개수를 입력받고, 각 테스트 케이스에 대해 두 정수를 입력받아 더한 값을 출력합니다.

```
import java.util.*;

public class Adder
{
        public static void main(String[] args)
        {
                Scanner sc = new Scanner(System.in);
                int t = sc.nextInt();
                for (int i = 0;i < t;i++)
                {
                        int a = sc.nextInt();
                        int b = sc.nextInt();
                        System.out.println(a + b);
                }
        }
}
```

언어의 규칙 상, 소스 파일의 이름과 클래스 이름이 **일치해야** 함에 유의하십시오.

This page is intentionally left blank.

Problem A. Anagram

Algospot Inc. is planning to launch a new product iWook on September 2011. iWook, is a diskless desktop computer and will be used by geek CS major students around the world. To classify each device, a unique serial number is assigned.

Wookayin, the tech lead of iWook, decided that the password for a iWook device must use all letters from the device's serial number exactly once in a case-sensitive way, and not in the same order (*Obviously, in the real world, it's a faily bad security practice to enforce such restrictions to passwords.*)

Algospot Inc. just hired you to write a program to verify passwords entered by users. For a successful launch, develop the perfect verifier ASAP!

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The following T lines will each contain two strings, the first of which being the serial number of the device, and the second being a password entered by a user for the device.

Both strings will only contain alphanumeric characters, and will not exceed 100 characters in length.

Output

For each test case, print a single line. If the given password is valid for the device, print "`Yes`" (quotes are for clarity). Otherwise, print "`No.`".

Sample input and output

Standard Input	Standard Output
2 weird wired apple angle	Yes No.

Problem B. 너의 공격 패턴을 파악했다

서기 2222년, cdinside 알고스팟 갤러리에서 WonhaJin은 무패 전설의 키보드 워리어로 유명하다. WonhaJin은 항상 만년떡밥을 던짐으로써 낚여든 상대방을 무참히 공격하여 굴복시키곤 했다. 이러한 패턴으로 그는 한 번도 키보드 배틀에서 패배한 적이 없다.

한편, 그동안 그와 대등하게 승부했지만 한 번도 이겨 본 적이 없는 숙명의 라이벌 WookMan은 WonhaJin에게 패배를 안겨주고자 22개월간 WonhaJin의 키보드 배틀 패턴을 파악하고자 노력하였다.

그러던 도중 2222년 2월 22일 22시 22분 22초에 그가 뻘글을 쓰는 것을 확인하였다. 이는 키보드 워리어에게 있어 치명적인 약점이다. WookMan은 이로부터 그가 성시마다 약점을 보인다는 가설을 세웠다. 성시란 다음과 같다:

1. 어떤 시각이 24시간제 기준으로 X년 Y월 Z일 A시 B분 C초이면, 이를 $XYZABC$의 형태로 이어 붙인 것과 그것을 뒤집은 문자열이 서로 같을 때 그 시각은 성시이다.
2. 시각은 그레고리력(양력)을 따라 4의 배수 해는 2월이 29일까지 있는 윤년이다. 단, 100의 배수 해는 윤년이 아니지만 400의 배수 해는 윤년이다.
3. X는 네 자리에 맞추어 표기하며, 나머지는 두 자리에 맞추어 표기한다. 모자라는 자리는 "0"으로 채운다.

예를 들어 2011년 9월 20일 23시 57분 11초의 경우, 20110920235711인데 이는 이를 뒤집은 11753202901102와 다르기 때문에 성시가 아니다. 한편 2011년 12월 11일 21시 11분 2초의 경우 20111211211102이므로 성시이다.

이러한 패턴을 이용하여 WonhaJin을 공격하고자 WookMan은 1970년 1월 1일 0시 0분 0초 (이는 프로그래머들에게 있어 현존하는 모든 역사가 시작된 시간이다) 부터 N번째로 등장하는 성시를 찾는 프로그램을 작성하려고 하였는데, 최근 취업과 육아를 동시에 시작하여 너무 바쁜 나머지 추종자들에게 도움을 요청하였다.

WookMan의 꼼꼼함에 반한 추종자인 당신은 이 프로그램을 작성해주어야 한다.

Input

입력은 여러 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 테스트 케이스의 수 T가 주어진다.

각 테스트 케이스는 하나의 행으로 이루어져 있는데, 그 행에는 양의 정수 N $(1 \leq N \leq 10,000)$이 주어진다.

Output

각 테스트 케이스에 대해 N번째 성시를 한 행에 하나씩 "`XXXX-YY-ZZ AA:BB:CC`"의 형식으로 출력한다.

Sample input and output

Standard Input	Standard Output
4 7 2 20 99	2000-11-11 11:00:02 2000-01-22 10:00:02 2001-12-22 21:10:02 2013-12-11 21:31:02

Problem C. Hamming Code

Jaeha is writing an operating system for his toys, so they will be able to communicate with each other. However, the wireless chips in his toys are very cheap, so they are prone to transmission errors. Quite frequently, Jaeha is seeing some noises in the transmitted data: some bits get flipped during the transmission process. Jaeha wants to implement Hamming Code to remedy this situation.

The following is a brief description of how Hamming Code works.

> Hamming(7,4) code encodes 4 bits of data into a 7-bit code, by adding 3 bits of parity data. The parity data ensures that the receiver will be able to decode the correct data even when one of the 7 bits get flipped during transmission. The encoding process is as follows.
>
> 1. First, we number the bits in the encoded message from 1 to 7.
> 2. Next, we assign the data to be transmitted to the 3rd, 5th, 6th, 7th bit of the encoded data. For example, if we were to transmit data `1011`, the encoded message will look like `_ _ 1 _ 0 1 1`, leaving 3 bits to be filled in.
> 3. Next, the remaining bits are filled by parity data, from the first bit, as follows.
>   - The 1st bit is filled so that the XOR of 1st, 3rd, 5th and 7th bit is `0`. In the above example, the XOR of 3rd, 5th and 7th bit is already `0`, so the 1st bit must be `0`.
>   - The 2nd bit is filled so that the XOR of 2nd, 3rd, 6th and 7th bit is `0`. In the above example, the XOR of 3rd, 6th and 7th bit is `1`, so the 2nd bit must be `1`.
>   - The 4th bit is filled so that the XOR of 4th, 5th, 6th and 7th bit is `0`. In the above example, the XOR of 5th, 6th and 7th bit is already `0`, so the 4th bit must be `0`.
> 4. After the process, we get the fully encoded message `0 1 1 0 0 1 1`.

Position	1	2	3	4	5	6	7
Raw Message			1		0	1	1
Encoded Message	0	1	1	0	0	1	1

> Now, let's talk about the decoding process of Hamming(7,4) code. To see how the code corrects an isolated error, let's suppose that the 3rd bit gets flipped incorrectly when the message `0 1 1 0 0 1 1` is transmitted. Therefore, the receiver will receive the following corrupted message instead: `0 1 0 0 0 1 1`.

Position	1	2	3	4	5	6	7
Encoded Message	0	1	1	0	0	1	1
Error			x				
Corrupted Message	0	1	0	0	0	1	1

> The receiver then calculates a 3-bit integer, called **syndrome**, as follows.
>
> - The 1st (lowest) bit is determined as the XOR of 1st, 3rd, 5th and 7th bit. In the above corrupted message, this bit will be `1`.

- The 2nd bit is determined as the XOR of 2nd, 3rd, 6th and 7th bit. In the above corrupted message, this bit will be `1`.
- The 3rd bit is determined as the XOR of 4th, 5th, 6th and 7th bit. In the above corrupted message, this bit will be `0`.

Concatenating these three digits, the syndrome will be $011_2 = 3_{10}$: the location of the error bit.

Position	1	2	3	4	5	6	7	
Check 1	0		0		0		1	1
Check 2		1	0			1	1	1
Check 3				0	0	1	1	0

Syndrome $= 011_2 = 3_{10}$

Thus the receiver can flip the 3rd bit in the received message. The resulting sequence of `0`'s and `1`'s will be now correct and we can take the 3rd, 5th, 6th and 7th bit to get the original message: `1 0 1 1`.

Position	1	2	3	4	5	6	7
Corrupted Message	0	1	0	0	0	1	1
Indicated Error			x				
Corrected Message	0	1	1	0	0	1	1
Decoded Message			1		0	1	1

Note when all bits are transmitted correctly, the syndrome will be `0` and no bit needs to be flipped.

As a smart baby, Jaeha had no problem writing the encoding procedure. However, his parents don't allow him to use the computer more than 10 minutes a day, because he is only 3 months old. Let's help Jaeha by implementing the decoding procedure.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input. The next T line each contains a transmitted message in 7 binary digits, starting from the 1st bit. The message will contain at most a single incorrect bit.

Output

For each test case, you must output a single line of four binary digits denoting the decoded message.

Sample input and output

Standard Input	Standard Output
2 0100011 1111111	1011 1111

Problem D. 그리프시드 모으기

M 명의 마법소녀가 있습니다. 그들에게는 마스코트인 큐베 (◕ ‿ ◕) 라는 생물이 있습니다. 큐베는 마법소녀들에게 한 가지 제안을 합니다.

> "너희는 열심히 하니까 그리프시드라는 보물이 잔뜩 모여 있는 곳을 알려줄게."

큐베는 지도를 보여 줍니다. 지도는 R 행 C 열의 행렬로 되어 있었고, 각 칸에는 그 지점에 있는 그리프시드의 개수가 나와 있습니다. 그리고 큐베는 한가지 조건을 더 붙입니다.

> "하지만 너희 마법소녀들이 보물을 규칙 없이 가져가면 내가 지도를 갱신하는 게 피곤하니까 규칙을 정하자. 축과 평행한 직사각형 영역을 미리 정해두고, 나와 계약하면 거기 있는 걸 전부 가져가는 것을 허락할게."

마법소녀들은 수군댑니다. 그 중 가장 쿨한 호무라라는 마법소녀는 이런 제안을 합니다.

> "우리끼리 싸우면 곤란하잖아? 그러니까 그리프시드의 개수가 M 의 배수가 되도록 큐베와 계약하자."

그러자 욕심이 많은 마법소녀 쿄코가 말합니다.

> "좋아. 대신 최대한 많은 그리프시드를 모을 수 있도록 직사각형 영역을 정해보자."

그런데 마법소녀들은 어느 영역을 골라야 큐베의 조건과 호무라와 쿄코의 제안을 만족하게 할 수 있을지 지도가 너무 방대해서 알 수 없게 되어버렸습니다. 여러분이 이 마법소녀들이 적절한 영역을 고를 수 있도록 도와주세요.

Input

입력은 여러 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 테스트 케이스의 수 T 가 주어진다.

각 테스트 케이스의 첫 행에는 세 정수 R, C $(1 \leq R, C \leq 300)$ 와 M $(1 \leq M \leq 100,000)$ 이 공백으로 구분되어 주어진다. 다음 R 행에 걸쳐 한 행에 음이 아닌 정수가 C 개씩 주어지며, 이는 행렬로써 지도에 나온 그리프시드의 개수를 의미한다. 모든 그리프시드의 개수의 합은 32비트 부호형 정수로 표현 가능하다.

Output

각 테스트 케이스에 대해 한 행에 하나씩 조건을 만족시켰을 때 얻을 수 있는 그리프시드의 개수를 출력한다. 만약 조건을 만족하는 영역이 존재하지 않는다면 0을 출력한다.

Sample input and output

Standard Input	Standard Output
2 1 1 3 1000 3 3 6 1 2 3 4 5 6 7 8 9	0 24

Problem E. 보노보노와 동굴 아저씨

> 동굴 아저씨는 어디선가 나타난다.
> 동굴 아저씨는 존재한다.
> 곤란해하기만 하면 동굴 아저씨가 가둬버릴지도 몰라.
> 하지만 곤란한 건 어쩔 수 없는데.
> 동굴 아저씨가 나타나면 난 더 곤란한데.
>
> — 보노보노(해달, 불명)

아기 해달 보노보노는 동굴 아저씨를 한 번도 만나본 적이 없지만 무척이나 무서워한다. 동굴 아저씨는 언제건 **"나쁜 아이는 동굴 안에 가둬버린다"**며 보노보노를 아래 그림과 같이 동굴 안에 가두어버리기 때문이다. 너부리에게 동굴 아저씨 따위는 없다고 핀잔을 듣곤 하지만, 그런 것 따위 아무래도 상관없다.

보노보노는 동굴 아저씨에게 갇히는 상상을 하다가 용기를 내어 인권… 아니 해달권을 주장하기로 했다. 아무리 잘못했기로서니 빛이 하나도 들어오지 않는 어두컴컴한 곳에 가두는 것은 너무나 잔혹한 일이다. 그래서, 보노보노는 동굴 안을 환하게 밝혀 달라고 동굴 아저씨에게 사정했다.

동굴 아저씨는 사실 그렇게 나쁜 사람… 아니 동물이 아니라서, 그 부탁을 들어주기로 했다. 동굴 아저씨의 동굴은 직선 형태로 뻗어 있고, 그 안에 보노보노와 다른 나쁜 아이들이 갇혀 있는 돌무더기 N 개가 일정한 간격으로 존재한다. 동굴 아저씨는 나쁜 아이들 중 몇몇에게 횃불을 맡길 생각이다. 횃불을 맡은 나쁜 아이는 어두컴컴한 곳에 갇혀 있는 대신 횃불을 들고 서있어야 한다.

그런데 횃불은 크고 무겁고, 동물들은 각자 체격 조건이 다르기 때문에 감당할 수 있는 횃불의 크기가 다를 수 있다. 횃불의 크기는 양의 정수로 정의되는데, 크기가 k 인 횃불은 자기 자신을 포함하여 좌우로 $k-1$ 개의 돌무더기를 더 밝힐 수 있다. 예를 들어 크기가 2 인 횃불은 총 세 개의 돌무더기를 밝힐 수 있다.

동굴 아저씨는 최대한 많은 나쁜 아이들에게 벌을 주어야 하기 때문에 되도록 적은 동물들에게 횃불을 들고 서 있게 하고 싶다. 동굴을 모두 밝히기 위해서는 최소 몇 마리의 동물이 횃불을 들고 서 있어야 하는가?

Input

입력은 여러 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 테스트 케이스의 수 T 가 주어진다.

각 테스트 케이스의 첫 행에는 돌무더기의 갯수 N $(1 \leq N \leq 200,000)$이 주어진다. 다음 행에는 동굴의 맨 왼쪽의 돌무더기에 갇힌 동물부터 차례로 감당 가능한 횃불의 크기를 나타내는 정수 N 개가 주어진다. 횃불의 크기는 1 이상 $200,000$ 이하이다.

Output

각 테스트 케이스에 대해 한 행에 하나씩, 횃불을 들어야 하는 최소 동물의 수를 출력한다.

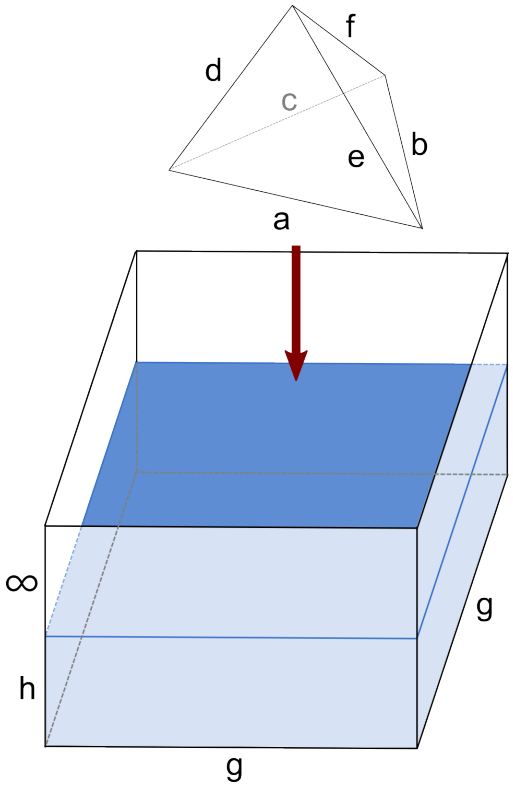
Sample input and output

Standard Input	Standard Output
1 9 1 2 2 1 3 1 1 2 4	3

Problem F. Water World

Ainu7, who is fond of kids, invented a simple game named "Water World" to train children's mathematical skills.

The game is quite simple to play. As you can see in the picture to the right, the game involves two items. The first item is the water tank, which is a large rectangular box filled with water. The bottom face of the tank is a square of size $g \times g$. The current water level is given as h. The second item is a small triangular pyramid block (a tetrahedron).



The game is played by two children. At the beginning of a game, one player(called thrower) throws the pyramid block softly into the tank. Now the opponent(called guesser) should guess how much the water level has increased. If the guess is correct, the guesser wins, otherwise the thrower wins.

You can assume that the water tank is large enough so the block will have one of its faces completely touching the bottom of the tank.

Before the game begins, ainu7 wants to find the correct answer of this game. Because most kids' guesses are too large, ainu7 will just memorize the largest answer possible. That will enable him to quickly find out if an answer is too large.

Help ainu7 to release his new game to play with his beloved children!

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain two integers g and h: g ($1 \leq g \leq 100$) represents the length of bottom side of the water tank, h ($1 \leq h \leq 1000$) represents the current level of water in the tank. The second line will contain six integers a, b, c, d, e, and f ($1 \leq a, b, c, d, e, f \leq \frac{g}{2}$). Each integer represents the length of a side of the pyramid block, with respect to the figure above.

Output

Print exactly one line for each test case. The line should contain the biggest change of water level. The number should be rounded to three digits after the decimal point.

Sample input and output

Standard Input	Standard Output
1 20 10 6 6 6 6 6 6	0.064

Problem G. Memilization V

— The enemy at the hill.

You are playing a game, "Memilization V". In this game, each player has his own country and cicada(매미) in the capital of the country. A player must conquer other players' capital to kill their cicada. When a player's cicada is killed, he will fall in the "Memi-less" state and lose the game.

"Memilization V" is a turn-based game; in each turn, a player can command each of his unit to move or to attack (or, do nothing). You can only issue a single command for each unit in each turn — you can not command a unit to move AND attack both in one turn.

The map of the game consists of hexagon tiles, where each tile of the map has its own "height" (which will be important for the upcoming task). Moreover, no two units will share a single tile; that is, at most one unit can be on a single tile at any time.

Now, you and your friend started a game, and your friend gathered a numerous army and moved to invade your country! Fortunately, one of your scout unit found the army, so you planned to ambush the enemy far from your capital. As the battle will be placed far from your capital, let us assume your capital will be safe — don't worry about your cicadas.

Your operation code name is "Shock and Awe" — you wish to kill as many units as possible in a single turn. For this, you planned to use portable missile turret units (abbreviated as PMTs). A PMT is a very powerful weapon with ability to attack in range. It can even kill any unit with a single attack. For a PMT unit located at the tile X to attack an enemy unit at another tile Y, the following two conditions have to be satisfied:

1. The length of the shortest path from X to Y must be equal to or less than D.
2. There must be at least one shortest path from X to Y where each tile's height on the path is less than or equal to the height of X.

Note that a path from X to Y is defined as a sequence of tiles beginning at X and ending at Y, where two adjacent tiles share a border. The length of a path is defined as the number of tiles in the path minus 1. For example, the length of a path containing only two adjacent tiles will be 1.

You are given the map of the battlefield, which contains the information about the heights of each tile, the positions of your PMT units and your friend's units. Now this is the first turn; how many units of your firend you can kill at most in the first turn?

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain two integers R, D ($1 \leq R \leq 50, 1 \leq D \leq 100$). The local map has a hexagonal shape with each side having R tiles. The next $2R+1$ lines will contain the height of each tile, which will be integers between 1 to 100, inclusive. The next $2R+1$ lines will tell you whether each tile contains a unit or not. Each character in these lines will be one of "`M`", "`E`", or "`.`":

- The character "`M`" means one of your PMT units is on the tile.

- The character "E" means an enemy unit is on the tile.
- The character "." means there are no units on the tile.

The total number of PMT units, and the total number of enemy units will not exceed 100. Please consult the sample input for detailed specification of the input.

Output

Print exactly one line for each test case. The line should contain the maximum number of units you can kill in the first turn.

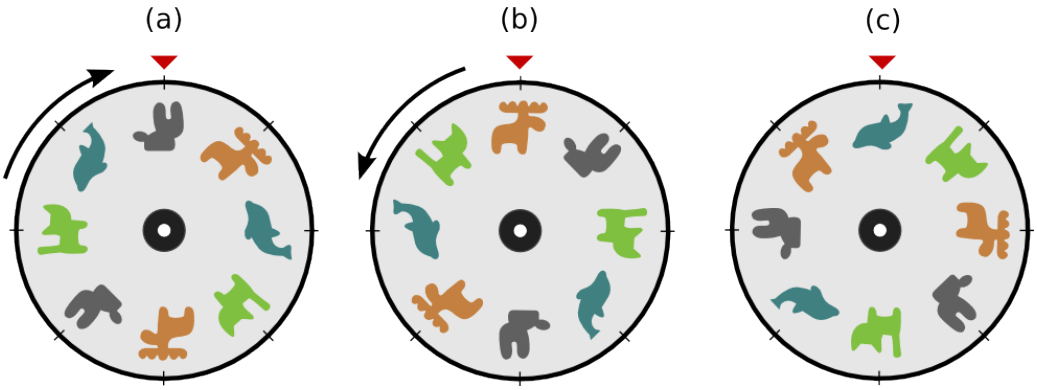
Sample input and output

Standard Input	Standard Output

2
3 2
  1 2 1
 1 2 2 1
1 1 2 3 1
 1 1 3 3
  1 1 3
  . E .
 . . . .
. . M M .
 . . . .
  . . E
3 2
  1 2 1
 1 3 3 1
1 1 2 3 1
 1 1 3 3
  1 1 3
  . E .
 . . . .
. . M M .
 . . . .
  . . E
	2
	1

Problem H. Jaeha's Safe

Jaeha just bought a new children's safe to store his valuables. The safe uses a dial to lock the door, and on the rim of dial are drawn pictures of cute animals instead of numbers. To open the safe, Jaeha must rotate the dial to reach certain positions, alternating direction each time.



The above pictures show an example of how the safe can be opened. Picture (a) shows the dial's current position. Jaeha needs to rotate the dial clockwise until position (b) is reached. Then Jaeha needs to rotate the dial counterclockwise until position (c) appears, and then the safe will open.

Jaeha wants to be careful, so he only rotates the dial one *tick* at a time. A tick means a picture drawn on the rim of the dial. Therefore, to open the safe in the above example, Jaeha needs to rotate 4 ticks to reach position (b) and 6 ticks to reach position (c), totalling 10 ticks.

Like his father, Jaeha is an impatient child. Let's help him out by writing a program that, given a set of dial positions, calculate how many ticks Jaeha has to rotate the dial to open the safe.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input.

The first line of each test case will contain an integer $N (1 \leq N \leq 50)$, the number of positions Jaeha needs to reach. The next $N + 1$ lines will each contain a dial configuration. A configuration is given by listing the pictures in clockwise order, starting from the topmost picture. Each type of picture is denoted by an alphabet character, therefore each configuration is given as a string. The first configuration shows the current dial. Jaeha will rotate clockwise to reach the second configuration, rotate counterclockwise to reach the third, and so on.

The number of pictures on a dial will not exceed 10,000.

Two adjacent configurations given in the input will always be different. It is always possible to open the safe.

Output

Print exactly one line for each test case. The line should contain the minimum number of ticks required to open the safe.

Sample input and output

Standard Input	Standard Output
2 3 abbab babab ababb bbaba 2 RMDCMRCD MRCDRMDC DCMRCDRM	6 10

Problem I. 호무라의 군수창고 습격

"호무라"는 무기 수집가이다. 호무라는 탁월한 능력을 갖추고 있는 마법소녀이기도 하다. 따라서 무기를 모으거나 위기를 모면하는 것은 이미 호무라에게는 익숙한 일상일 뿐이었다.

하지만 그런 그녀도 실수를 하는 법이라, 얼마 전에 자신의 능력을 잘못 사용하여 그동안 모은 무기를 모두 잃어버리고 말았다.

> "호무, 호무……. 나의 무기로 가득했던 진열장이 사라지다니 이렇게 슬픈 일이!"

슬퍼하던 호무라는 지금 다시 무기를 모으기 위해 군수창고를 습격할 계획을 세우고 있다. 그녀는 이미 습격하고자 하는 군수창고가 무기 보관용 캐비넷 N 개가 일렬로 배치된 형태이며, 각 캐비넷에는 정확히 한 종류씩의 무기가 들어 있음을 확인해 두었다. 심지어는, 각 캐비넷에 어떤 무기가 들어 있는지조차 완벽하게 파악해 두었다.

이러한 탈취 작전은 신속하고 정확할 필요가 있다. 그래서 호무라는 신속성을 위해 N 개의 캐비넷 중 연속된 일부분만을 탈취하려 하며, 탈취한 무기들의 종류가 정확히 K 종류가 되도록 하자고 스스로에게 다짐했다. 호무라가 무기를 가져올 수 있는 방법은 몇 가지나 있을까?

Input

입력은 여러 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 테스트 케이스의 수 T가 주어진다.

각 테스트 케이스는 두 행으로 구성된다. 각 테스트 케이스의 첫 번째 행에는 캐비넷의 개수 N $(1 \leq N \leq 10^6)$과 탈취해야 하는 무기의 가짓수 K $(1 \leq K \leq 10^6)$가 공백을 사이에 두고 차례로 주어진다. 그 다음 행에는 무기의 종류를 나타내는 번호 A_i $(1 \leq A_i \leq 10^9)$ 가 N 개 캐비넷이 배치된 순서대로 주어진다.

Output

각 테스트 케이스에 대해 한 행에 하나씩 목적을 달성할 수 있는 구간의 경우의 수를 출력한다.

Sample input and output

Standard Input	Standard Output
2 3 4 1 2 3 7 2 1 2 2 4 2 3 1	0 9

Notes

두 번째 테스트 케이스는 $\langle 1,2\rangle$, $\langle 1,2,2\rangle$, $\langle 2,2,4\rangle$, $\langle 2,2,4,2\rangle$, $\langle 2,4\rangle$, $\langle 2,4,2\rangle$, $\langle 4,2\rangle$, $\langle 2,3\rangle$, $\langle 3,1\rangle$ 으로 아홉 가지가 있다.

Problem J. 매미 원리

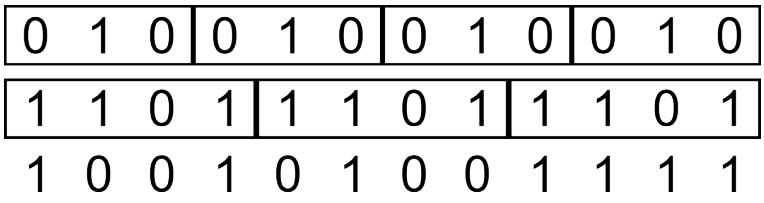
여름이면 성장을 마치고 지상으로 나와 구애를 하는 매미들의 경우, 대부분의 종은 매년 여름 지상에서 발견할 수 있다. 이와는 달리 일부 종은 다른 해에는 발견되지 않다가 특정 해에만 대량 발견되곤 한다.

예를 들어 북미에는 이러한 매미가 일곱 종이 있으며, 이러한 매미들을 매미목 매미과의 주기매미속(*Magicicada*)으로 분류한다. 이들 중 네 종은 주기가 13년이고, 세 종은 17년 간격으로 지상에 대규모로 등장한다. 이들이 12년째, 16년째 등에 등장하는 경우는 거의 발견되지 않는다. 이와 같이 소수 간격의 동기화된 생애 주기를 설명할 수 있는 가장 설득력있는 가설은 천적의 개체 수 증감 사이클과의 생애 주기가 동기화되는 일을 최소화하고 지상으로 나갈 때에는 최대한 많은 개체가 나감으로써 생존을 보장하고자 한다는 것이다. 실제로, 주된 천적인 새, 거미, 뱀의 경우 2-6년 간격으로 개체 수의 증감을 보인다고 한다.

이와 같은 효율적인 원리를 웹 디자인에 응용할 수 있다. 웹 페이지의 배경 이미지로 작은 소수 길이의 패턴 여러 가지를 포개어 놓으면 사람의 눈으로는 규칙성을 파악할 수 없다는 것이다. 예를 들어 29픽셀, 37픽셀, 53픽셀 간격으로 반복되는 세 패턴을 준비해 포개어 두게 되면 $29 \times 37 \times 53 = 56869$픽셀의 간격으로 같은 패턴이 반복되게 된다.

특급 긱인 Being은 이를 이용해 웹 페이지의 배경 이미지를 줄무늬로 디자인하려고 한다. 줄무늬이므로 이를 1차원의 색상 배열로 생각할 수 있다. 그러나 이 웹 페이지는 흑백 터미널 시절에 대한 오마쥬를 컨셉으로 하기 때문에, 사용할 수 있는 색은 검은 색과 흰 색 뿐이다.

Being은 시험 삼아 패턴 여러 개를 겹쳐 보았는데, 흰 색이 지나치게 많이 등장하는 문제가 있었다. 이를 해결하기 위해 최종적으로 표현되는 색을 해당 칸의 여러 색들 중 하나가 흰 색일 때 흰 색으로 두는 것이 아니라, 검은 색과 흰 색을 각각 0과 1로 표현하여 XOR 연산을 한 결과값으로 생각하기로 하였다. 이와 같이 하니 무늬가 마치 난수열과도 같이 적절하게 표시되는 것을 확인할 수 있었다.



이렇게 하고 나니 자연스럽게 이 결과물이 얼마나 규칙성을 파악하기 어려운 지 알아내고 싶어진 Being은 이 문제를 해결하는 프로그램을 작성하기로 했다. 패턴의 개수 N과 각 패턴의 길이 L_i, 그리고 이 패턴들의 시작점을 일치시켜 여럿 겹쳤을 때의 결과의 맨 앞 일부분이 주어졌을 때 가능한 패턴들의 방법의 수를 결정하는 프로그램을 작성하자.

Input

입력은 여러 개의 테스트 케이스로 구성된다. 입력의 첫 행에는 테스트 케이스의 수 T가 주어진다.

각 테스트 케이스의 첫 행에는 패턴의 개수 $N(1 \leq N \leq 100)$과 각 패턴의 길이를 나타내는 N개의 정수 $L_i(1 \leq L_i \leq 100)$가 공백으로 구분되어 주어진다. 패턴의 길이의 총 합은 63을 넘지 않는다. 다음 행에는 패턴을 표시한 결과의 맨 앞 일부분을 나타내는 "`0`"과 "`1`"로 구성된 문자열 $S(1 \leq |S| \leq 10000)$가 주어진다.

Output

각 테스트 케이스에 대해 한 행에 하나씩, 가능한 패턴의 수를 출력한다.

Sample input and output

Standard Input	Standard Output
1 2 3 4 100101001111	2

Notes

첫 번째 테스트 케이스의 경우 "010"과 "1101", 또는 "101"과 "0010" 두 가지 조합이 가능하다.

이 문제는 아래의 웹 페이지를 참조했다.

1. Periodical Cicada Page,
   http://insects.ummz.lsa.umich.edu/fauna/michigan_cicadas/Periodical/
2. The Cicada Principle and Why It Matters to Web Designers,
   http://designfestival.com/the-cicada-principle-and-why-it-matters-to-web-designers/