netsoc

UCD Internet and Computer Science Society

Programming Competition

Questions Sheet

April 22, 2013

Quick guide

Welcome to the second netsoc Programming Competition!

Prizes

One **Google Nexus 7** per winner. There are two tablets to be won.

NOTE: The winner is selected by the jury, and might not reflect the scores on the DOMJudge scoreboard! *This is due to different question sheets per year!*

Rules

- You are required to bring your own computer for the competition
- The submissions open at **6PM** on the 24th of April and close at **8:30PM**
- You are allowed to use **any** text editor of your choice (This includes IDEs, such as eclipse)
- You are **NOT** allowed to use the **Internet**! Exceptions:
  1. Submit solutions to DOMJudge
  2. Consult the documentation of the language of your choice
- The allowed languages and the installed compilers/interpreters are:
  - **C** *gcc v4.6.3*
  - **C++** *g++ v4.6.3*
  - **Java** *OpenJDK v1.7.0_15*
  - **Ruby** *ruby v1.8.7*
  - **Python** *python v2.7*
  - **JavaScript** *node v0.10.4*
  - **LOLCODE** *lci v0.10.4*
  - **Malbolge** *malbolge v0.1.1*

Example I/O code for Netsoc programming contest.

Each of the simple examples below shows how to read and write from stdin and stdout in the majority of the languages used for the competition. With the exception of some of the more esoteric languages such as LOLCODE & Malbolge.

Example C code

```
#include <stdio.h>
int main(void)
{
    char buffer[14];
    int c;
    int i = 0;
    while ((c = getchar()) != EOF) {
        buffer[i] = c;
        i++;
    }
    printf("Hello %s",buffer);
}
```

Example C++ code

```
#include <iostream>
using namespace std;

int main ()
{
    string i;
    cin >> i;
    cout << "Hello " << i << "\n";
    return 0;
}
```

Example Java Code

```
import java.io.BufferedReader;
import java.io.IOException;
```

```
import java.io.InputStreamReader;


public class HelloWorld {
    public static void main(String[] args) throws IOException {
        BufferedReader in = new BufferedReader  (new InputStreamReader(System.in))
        String s;
        while((s = in.readLine()) != null){
            System.out.println("Hello, "+s+"!");
        }
    }
}
```

Example Ruby Code

```
s = gets
puts "Hello #{s}"
```

Example Python Code

```
s = raw_input()
print "Hello "+s
```

Java script code (Uses Node.js)

```
process.stdin.resume();
process.stdin.setEncoding('utf8');

process.stdin.on('data', function (text) {
    process.stdout.write('Hello '+text);
    process.exit();
});
```

1 Hello World! - NC2P0

To test the submission system, write a program which greets a person.

Input

The name of the person

Output

"Hello" followed by the name and an exclamation mark.

Example

Input

Dave

Output

Hello Dave!

2 Sum Them Digits - NC2P1

As a crude form of hashing function, Lars wants to sum the digits of a number. Then he wants to sum the digits of the result, and repeat until he have only one digit left. He learnt that this is called the digital root of a number, but the Wikipedia article is just confusing him.

Can you help him implement this problem in your favourite programming language?

It is possible to treat the number as a string and work with each character at a time. This is pretty slow on big numbers, though, so Lars wants you to at least try solving it with only integer calculations (the modulo operator may prove to be useful!).

Input

A positive integer, possibly 0.

Output

An integer between 0 and 9, the digital root of the input number.

3 Maximum Random Walk - NC2P2

Consider the classic random walk: at each step, you have a $\frac{1}{2}$ chance of taking a step to the left and a $\frac{1}{2}$ chance of taking a step to the right. Your expected position after a period of time is zero; that is the average over many such random walks is that you end up where you started. A more interesting question is what is the expected rightmost position you will attain during the walk.

Input

The input consists of an integer n, which is the number of steps to take ($1 \leq n \leq 1000$). The final two are double precision floating-point values L and R which are the probabilities of taking a step left or right respectively at each step ($0 \leq L \leq 1, 0 \leq R \leq 1, 0 \leq L + R \leq 1$). Note: the probability of not taking a step would be 1-L-R.

Output Description

A single double precision floating-point value which is the expected rightmost position you will obtain during the walk (to, at least, four decimal places).

Sample Input

```
walk(1,.5,.5)
walk(4,.5,.5)
walk(10,.5,.4)
```

Sample Output

```
0.5000
1.1875
1.4965
```

4 Happy Days - NC2P3

Johnny has a pool in his garden. There are several islands in the pool. Some islands are connected by bridges. Any bridge can be removed. Every day Johnny removes some bridges so that there is only one way from any island to any other. In the evening he returns removed bridges to their places. Also he has some favorite bridges which he never removes. Johnny will be happy if he is able to make a configuration of bridges on the given day which he has never made before. You have to count the amount of days he will be happy. Of course, if the favorite bridges themselves don't satisfy the happiness condition Johnny will not be happy for even single day.

Input

The first line of input file contains number t the number of test cases. Then the description of each test case follows. The first line of each test case contains number n the number of islands. Islands are numbered with integers from 1 to n. Then n lines follow each containing n characters defining the connectivity matrix of those islands. Character in column x of line y will be 1 if the islands with numbers x and y are connected and 0 otherwise. The next line is number p the number of favorite bridges. The next p lines contain the pairs of islands that are connected by favorite bridges.

Output

For each test case print the number of days Johnny will be happy in this situation. *There is an additional new line at the end of the input.*

Constraints

```
1 <= t <= 5
2 <= n <= 30
1 <= p <= min(6, n-1)
```

Example

Input

```
1
4
0111
1011
1101
1110
2
1 2
3 4
```

Output

```
4
```

5 Paying up - NC2P4

In the mysterious country of Byteland, everything is quite different from what you'd normally expect. In most places, if you were approached by two mobsters in a dark alley, they would probably tell you to give them all the money that you have. If you refused, or didn't have any - they might even beat you up.

In Byteland the government decided that even the slightest chance of someone getting injured has to be ruled out. So, they introduced a strict policy. When a mobster approaches you in a dark alley, he asks you for a specific amount of money. You are obliged to show him all the money that you have, but you only need to pay up if he can find a subset of your banknotes whose total value matches his demand. Since banknotes in Byteland can have any positive integer value smaller than one thousand you are quite likely to get off without paying. Both the citizens and the gangsters of Byteland have very positive feelings about the system. No one ever gets hurt, the gangsters don't lose their jobs, and there are quite a few rules that minimize that probability of getting mugged (the first one is: don't go into dark alleys - and this one is said to work in other places also).

Input

The first line contains integer t, the number of test cases (about 100). Then t test cases follow. Each test case starts with n, the number of banknotes in your wallet, and m, the amount of money the muggers asked of you. Then n numbers follow, representing values of your banknotes. Your wallet does not hold more than 20 banknotes, and the value of a single banknote is never more than 1000.

Output

For each test case output a single line with the word 'Yes' if there is a subset of your banknotes that sums to m, and 'No' otherwise. *Add an extra newline character to the end of your output.*

Example

Input

```
3
3 3
1
1
1
5 11
1
2
4
8
16
5 13
1
5
5
10
10
```

Output

```
Yes
Yes
No
```

6 Mixtures - NC2P5

Harry Potter has n mixtures in front of him, arranged in a row.Each mixture has one of 100 different colors (colors have numbers from 0 to 99). He wants to mix all these mixtures together. At each step, he is going to take two mixtures that stand next to each other and mix them together, and put the resulting mixture in their place. When mixing two mixtures of colors a and b, the resulting mixture will have the color (a+b) mod 100. Also, there will be some smoke in the process. The amount of smoke generated when mixing two mixtures of colors a and b is a*b. Find out what is the minimum amount of smoke that Harry can get when mixing all the mixtures together.

Input

There will be a number of test cases in the input. The first line of each test case will contain n, the number of mixtures, $1 \leq n \leq 100$. The second line will contain n integers between 0 and 99 - the initial colors of the mixtures.

Output

For each test case, output the minimum amount of smoke. *You will need to include an additional trailing newline character.*

Example

Input

```
2
18 19
3
40 60 20
```

Output

```
342
2400
```

7 Cutting Recipes - NC2P6

The chef has a recipe he wishes to use for his guests, but the recipe will make far more food than he can serve to the guests. The chef therefore would like to make a reduced version of the recipe which has the same ratios of ingredients, but makes less food. The chef, however, does not like fractions. The original recipe contains only whole numbers of ingredients, and the chef wants the reduced recipe to only contain whole numbers of ingredients as well. Help the chef determine how much of each ingredient to use in order to make as little food as possible.

Input

Input will begin with an integer T, the number of test cases. Each test case consists of a single line. The line begins with a positive integer N, the number of ingredients. N integers follow, each indicating the quantity of a particular ingredient that is used.

Output

For each test case, output exactly N space-separated integers on a line, giving the quantity of each ingredient that the chef should use in order to make as little food as possible. *Add an extra newline character to the end of your output*

Example

Input

```
3
2 4 4
3 2 3 4
4 3 15 9 6
```

Output

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

7.0.1 Constraints

T$\leq$ 100

2$\leq$ N$\leq$ 50

All ingredient quantities are between 1 and 1000, inclusive.

8 A Simple Equation - NC2P7

Statement

Given N,A,B,C, find how many solutions exist to the equation : a + b + c $\leq$ N, such that $0 \leq a \leq A$, $0 \leq b \leq B$, $0 \leq c \leq C$.

Input

The first line contains the number of test cases T.

Each test case contains 4 integers, N,A,B,C.

$0 \leq N,A,B,C \leq 2500$

Output

Output T lines, one for each test case. *Add a trailing newline character to the end of your output*

Examples

Input

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

Output

```
20
4
```

9 Buying Sweets - NC2P8

Banknotes in the state of Strangeland don't have any regulated values like 1, 5, 10, 20, 50, 100, etc. In fact, it's possible to see any positive integer on a banknote of Strangeland. Indeed, this isn't the most convenient thing. Ann is working as a sweet seller at a shop in Strangeland. Every kind of sweets in this shop has its own cost, and sweets of the same kind have the same cost.

Customers in Strangeland are strange. A customer points at some kind of sweets and gives several banknotes to the seller. This means that he wants to buy a positive number of sweets of that kind. He doesn't tell the exact number of sweets he wants to buy. The only thing Ann knows is: an 'adequate' customer won't give any extra banknotes. It means that if you throw away any banknote, the resulting amount of money won't be enough to buy the wanted number of sweets.

Ann has to determine the number of sweets the customer wants. Help Ann write a program which determines this number or tells that it's impossible.

Input

The first line of the input contains a single integer T, the number of test cases (no more than 20). T test cases follow. Each test case consists of two lines. The first of these lines contains two integers N and X ($1 \leq N$, $X \leq 100$) separated by a single space. N is the number of banknotes given by the customer. X is the cost of a single sweet of the chosen kind. The second of these lines contains N space-separated integers Ai ($1 \leq Ai \leq 100$), the values of the banknotes.

Output

For each test case output exactly one line containing a single integer:

-1 if the customer is inadequate and has given extra banknotes, or

K, the number of sweets the customer wants to buy. If there are several possible answers, output the largest of them.

Add a trailing newline character to the end of your output

Example

Input

```
3
4 7
10 4 8 5
1 10
12
2 10
20 50
```

Output

```
-1
1
7
```

10 Reversing directions - NC2P9

Chef recently printed directions from his home to a hot new restaurant across the town, but forgot to print the directions to get back home. Help Chef to transform the directions to get home from the restaurant.

A set of directions consists of several instructions. The first instruction is of the form "Begin on XXX", indicating the street that the route begins on. Each subsequent instruction is of the form "Left on XXX" or "Right on XXX", indicating a turn onto the specified road.

When reversing directions, all left turns become right turns and vice versa, and the order of roads and turns is reversed. See the sample input for examples.

Input

Input will begin with an integer T, the number of test cases that follow. Each test case begins with an integer N, the number of instructions in the route. N lines follow, each with exactly one instruction in the format described above.

Output

For each test case, print the directions of the reversed route, one instruction per line. Print a blank line after each test case. *Add an additional trailing newline character*

Constraints

$1 \leq T \leq 15$

$2 \leq N \leq 40$

Each line in the input will contain at most 50 characters, will contain only alphanumeric characters and spaces and will not contain consecutive spaces nor trailing spaces. By alphanumeric characters we mean digits and letters of the English alphabet (lowercase and uppercase).

Examples

Input

```
2
4
Begin on Road A
Right on Road B
Right on Road C
Left on Road D
6
Begin on Old Madras Road
Left on Domlur Flyover
Left on 100 Feet Road
Right on Sarjapur Road
Right on Hosur Road
Right on Ganapathi Temple Road
```

Output

```
Begin on Road D
Right on Road C
Left on Road B
Left on Road A

Begin on Ganapathi Temple Road
Left on Hosur Road
Left on Sarjapur Road
Left on 100 Feet Road
Right on Domlur Flyover
Right on Old Madras Road
```

11 Tautology - NC2P10

Write a program that checks if the given logical expression is a tautology. The logical expression is a tautology if it is always true, regardless of logical value of its variables.

Input

On the first line there is the number of expressions to check (at most 35). The expression is in a prefix notation, that means that operator precedes its arguments. The following logical operators will be used: C - and D - or I - implies E - if, and only if N - not The variables will be lowercase letters (a-z). There will be no more than 16 different letters in the expression. The length of the expression will not exceed 111 characters.

Output

For each expression write one word: YES if it is a tautology, NO in other case. (Add an additional trailing newline to your output)

Example

Input

```
7
IIpqDpNp
NCNpp
Iaz
NNNNNNNp
IIqrIIpqIpr
Ipp
Ezz
```

Output

```
YES
YES
NO
NO
YES
YES
YES
```

12 Logging Game - NC2P11

Logging can be a very mundane job, but Alice and Bob have devised a game to help them pass the time. They take turns choosing a log, and cutting it into 2 smaller logs. The sum of the lengths of the 2 logs equals the length of the original log. The only restriction is that neither of the resulting logs may be shorter than 1 meter in length (but exactly 1 meter is fine). In other words, non-integer lengths are allowed. Alice makes the first cut, and the first logger who cannot make a legal cut loses.

Input

Input begins with an integer T, the number of test cases (less than 450). T test cases follow, each on its own line. Each test case begins with an integer N, the number of logs at the start of the game. N integers follow, giving the initial lengths of the logs. There are at most 7 logs, and the total length of the logs will not exceed 250 meters. Note that the initial lengths of the logs are integers, but logs may be cut to non-integer lengths.

Output

For each test case, output a single line containing the name of the winner of the game, assuming both loggers choose their cuts optimally. *Add a trailing newline character to your output*

Example

Input

```
3
1 2
2 3 4
3 7 8 9
```

Output

```
Alice
Alice
Bob
```

13 Squash the Bugs - NC2P12

Bugs have gotten into The Chefs kitchen! Help him trap them all and hell make you a batch of his famous chocolate chip cookies. You are given a given a square map of the kitchen divided into tiles, and in each tile sits some known number of bugs. You also have a square trap, which can be dropped to cover a certain number of tiles (the trap may only cover tiles from within the map, and must be aligned to the borders of the kitchen). However, the trap only catches bugs from one of the tiles which it has covered, having a minimum number of bugs on it. For all possible positions at which the trap can be dropped, determine number of bugs that will be caught.

Input

Two numbers, $0 < n \leq 1000$ (size of the map), and $0 < k \leq n$ (size of the trap), followed by n rows with n numbers, determining the number of bugs on each tile. The number of bugs on each tile will fit in an signed 32-bit integer. *Add a trailing newline character to your output*

Output

You should output n-k+1 rows with n-k+1 numbers in each row.

Example

Input

```
4 2
0 1 2 3
4 5 6 7
8 9 0 1
2 3 4 0
```

Output

```
0 1 2
4 0 0
2 0 0
```

14 Quadratic Equations - NC2P13

Knowing Johnny's mathematical talent, our teacher has prepared a new interesting problem for him, hoping he will enjoy solving it. The problem description is given below.

"There is a rectangular room of length l and width w (l and w are integers). The length and width of the room fulfill the relation l=Aw+B, where A and B are given integer constants. The room is divided into square cells of unit dimensions. You have observed that, after adding an integer C to the number of cells in the room, the number of cells becomes divisible by the prime number P. Find all the possible values of the width of the room."

Input

The first line contains t, the number of test cases (about 10000). Then t test cases follow. Each test case is given in one line containing 4 integers A, B, C and P ($2 \leq P < 106, 0 < A < P, 0 \leq B, C < P$). P is always a prime number.

Output

For each test case, write the result in one line. The first number K is the number of solutions. Then K numbers X1, X2, ...,XK follow ($0 \leq Xi < P$) in increasing order, which are the solutions to the corresponding problems. *Add a trailing newline character to your output*

Example

Input

```
2
1 1 0 2
1 2 2 3
```

Input

```
2 0 1
0
```

15 Some More Homework - NC2P14

In the computer science class at the school in Byteland, the teacher handed out the following assignment as homework:

"For an integer **n**, let **bn** denote the bit parity of the binary representation of **n**, i.e. **bn**=0 if **n** has an even number of ones when written down in the binary system, and **bn**=1 otherwise. The numbers **bn**, for **n** $\geq$0, form an infinite sequence of bits (zeros and ones). Given a sequence **c**=(c0, ..., cp-1) of **p** bits, find the first occurrence of sequence **c** as a subsequence of **b** (i.e., the smallest value of index **k** such that for all **i** between 0 and p-1, we have ci = bi+k)."

And the teacher gave his students several short sequences c, asking them to provide the answers next day. Most, as expected, wrote programs to solve the task. Only Johnny computed the results by hand, claiming (quite correctly) that it was quicker that way. The teacher, slightly exasperated, decided to teach Johnny a lesson, and prepared a harder assignment, just for him.

"Given a sequence **c**=(c0, ..., cp-1) of **p** bits, for each s between 0 and p-1, compute the first occurrence of the prefix (c0, ..., cs) of sequence **c** as a subsequence of **b**."

And to be doubly sure that Johnny does his homework using a computer, the teacher gave him some really long sequences to deal with. Now, Johnny is in a bit of a spot, because he has never bothered to learn to program. Please help him out!

Input

The first line of input contains a positive integer $t < 10$, describing the number of tests. Exactly t test cases follow. Each test case is given in two lines. The first line contains integer p ($1 \leq p \leq 106$, the length of sequence k). The next line contains exactly p space-separated numbers (0 or 1), denoting successive elements of sequence c.

Output

For each test case, print a line containing exactly p space-separated numbers, corresponding to the indexes of the first occurrence of successive prefixes of c as subsequences of b. All indexes are numbered starting from zero. If there

is no such occurrence, output -1. (Add a trailing newline character to your output)

Example

Input

```
1
9
1 0 0 1 0 1 1 1 0
```

Output

```
1 2 4 4 8 8 8 -1 -1
```

16 Traffic jam - NC2P15

Little Johnny has a selection of boxes. Each box has a number on its side. The boxes are placed in a sequence, and Johnny wants to sort them (in ascending order). He has a device to manipulate the boxes, which performs the following operation. Johnny can select a subset of boxes, and the machine will lift the selected subset, shift the selected subset to the right (keeping the order in the subset), shift the not-selected subset to the left, filling up empty spaces (keeping the order in the subset), then finally move the raised boxes to be in one level again.

For example: if Johnny has the sequence: 1,2,**3**,**4**,5,**6**, and selects the subset in bold: 1,2,3,4,5,6, then the result is: 1,2,5,**3**,**4**,**6**.

Help Johnny to write a program that will calculate the minimal number of moves required to sort the given sequence of boxes in ascending order of numbers.

Input

First, $1 \leq t \leq 10$, the number of test cases. Then, t test cases follow. Each starts with $1 \leq n \leq 105$, the number of boxes. Then, n integer values describing the sizes of boxes in the sequence.

Output

For each test case, in a separate line, print the answer to that test case. *Add an additional newline character to the end of your output*

Example

Input

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

Output

```
2
```

17 Se7en - NC2P16

Tomek and his numerous friends are standing in a circle. They are all numbered with consecutive identifiers, from 1 to 1337 in the clockwise direction. Starting from person 1, who says "1", successive people read out successive positive integers. The starting direction is clockwise, and there is rule, that whenever integer is divisible by 7 or contains digit 7, the direction is reversed.

So, the identifiers of the persons who read-out successive numbers, are: 1, 2, 3, 4, 5, 6, 7 (person 7 has just read "7" and reversed the direction), 6, 5, 4, 3, 2, 1, 1337 (person 1337 has just read "14" and reversed the direction), 1, 2, 3 (person 3 has just read "17" and reversed the direction), 2, 1, 1337, 1336, and so on. Tomek has his favorite number, and he wants to calculate where in the circle he should stand to read that number out loud. Even though he is skilled programmer, he is a bit little lazy and would like you to help him out.

Input

First, $1 \leq t \leq 1000$, the number of test cases. Each test case is in a separate line, and contains a positive integer smaller than 10100, representing Tomek's favorite number.

Output

For each testcase, output one integer, the identifier which Tomek should choose in the circle to read his favorite integer loud. *Add an additional newline character to the end of your output*

Example

Input

```
3
10
100
1000
```

Output

```
4
2
1311
```

18 Short - NC2P17

Given n and k, find the number of pairs of integers (a, b) such that $n < a < k, n < b < k$ and ab-n is divisible by (a-n)(b-n).

Input

The first line contains the number of test cases t ($1 \leq t \leq 5$). Then t test cases follow, each test case consists of a line containing two integers n and k ($0 \leq n \leq 100000, n < k \leq 1018$).

Output

For each test case output one line containing the required number. *Add an additional newline character to the end of your output.*

Example

Input

```
2
1 5
2 5
```

Output

```
2
3
```

19 A Game of Thrones - NC2P18

Bran and Tyrion are the last two high lords fighting for the Iron Throne. With a mutual agreement that included all knights of the realm, it was decided to settle the issue with a game of thrones, of course not a game of swords but a game of numbers, after all one of them is a cripple and other is a dwarf. Seven wisest men of the realm came forward and forged rules of this game which are as follows :

Initially N numbers were written down on a notebook (possibly with multiple copies of every number).

Players alternate their turns with Bran playing first.

In the first turn Bran gets to choose a number of his choice from the numbers written on the notebook and declare it as the current number of the game.

After that in every move this is what they do : let's say the current number of the game is u. They erase u from the notebook (if u was written multiple times, they erase it only once) and declare one of the numbers still written on the notebook v as current number of the game. v can be chosen iff prime factorization of u and v differ by exactly 1 prime factor. (Read notes for a more formal definition)

The player who can't make a move loses the game.

You're one of Varys' spider and he has asked you to predict the outcome of this game beforehand so that he can devise future strategy. So you've to find out who has a winning strategy assuming both players play optimally.

Notes

1) v can be chosen after u iff either of the following conditions hold :

$v > u$ and u — v and (v / u) is prime

$u > v$ and v — u and (u / v) is prime

2) A natural number if prime if it has exactly two distinct positive factors. 1 is not a prime number.

Input

First line of the input contains a single integer N denoting number of different numbers written in the notebook. Then follow N lines. Each of the following lines contain two space separated integers : ui and ci where ui is the ith

distinct integer written on notebook and it has been repeated ci number of times.

Output

Your program should print Bran if he has a a winning strategy else it should print Tyrion. Also in case Bran could win, your program must output the smallest number Bran could choose in the first turn to ensure a win. See sample output for details.

Add an additional newline character to the end of your output

Example

Input

```
3
2 3
14 3
21 2
```

Output

```
Bran 21
```

19.1 Constraints

```
1 <= N <= 500
1 <= ui <= 10^18
1 <= ci <= 10^9
All ui are distinct.
```