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 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
```

3 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
```

3.0.1 Constraints

T$\leq$ 100

2$\leq$ N$\leq$ 50

All ingredient quantities are between 1 and 1000, inclusive.

4 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
```

5 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
```

6 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
```