Problem 1: Cyphering[1]

Vigenere cipher is similar to the Caesar cipher. In Caesar cipher, each letter of the alphabet is shifted along some number of places; for example, in a Caesar cipher of shift 3, “A” would become “D”, “B” would become “E", “Y" would become “B" and so on. The Vigenere cipher consists of several Caesar ciphers in sequence with different shift values.

For example, suppose that the plaintext to be encrypted is:

ATTACKATDAWN

The person sending the message chooses a keyword and repeats it until it matches the length of the plaintext, for example, the keyword “LEMON":

LEMONLEMONLE

Each letter in the plaintext is shifted according to the corresponding letter in the keyword. So the first letter “A” is shifted according to “L”, second letter “T” is shifted according to “E”, the third letter “T” is shifted according to “M”, and so on.

The shifting itself works as follows. If we are shifting according to, say, “L”, then: “A” becomes “L”, “B” becomes “M”, “C” becomes “N”, and so on. If we reach the end of the alphabet, then we wrap around to the beginning. So “O” becomes “Z”, and “P” becomes “A”. If the keyword letter is “E”, then each letter is shifted by 4 (so “T” becomes “X” and so on). The plain text above is encrypted to cypher text: LXFOPVEFRNHR

Input Format:
The first line contains the number of problem instances. Each following line will contain the keyword and the plaintext message. Both will be all caps and will contain only capital letters

Output Format:
Each line of output should contain the resulting cypher text.

Example:
Input:
2
LEMON ATTACKATDAWN
MERLIN KNOWSALL

Output:
LXFOPVEFRNHR
WRFHANXP

[1] Thanks to the University of Maryland 2012 High School competition for the description of this problem

Problem 2: Exam Proctors

The administrators in S University want to decide how many exam proctors they need for the coming final exams. The university has plenty rooms for final exams. Each exam will be held in a different room. Each exam will have one and only one proctor. An exam proctor will not be able to proctor two exams at the same time. In other words, if the exam time of two finals overlap, two proctors will be needed for these two exams. A proctor can continue proctoring another exam as long as the second one starts *strictly* after the end time of the first exam.

You will write a program to determine the minimum number of proctors needed given the finals schedule.

Input Format

The first line of input is *N*, the number of exams. The following *N* lines contain the start time and the end time of a certain final exam on each line. The start time and the end time are separated by a space. The time format is HH:mm. *HH* is the hour of the day which is an integer from 0 to 23. *mm* is the minute of the hour which is an integer from 0 to 59. The end time is guaranteed to be at least one minute later than the start time. All the exams will start and end on the same day.

Output Format

The minimum number of proctors needed for the finals.

Example:

===========

Input:

6
01:12 02:45
01:01 01:35
02:40 02:58
02:42 03:14
01:00 02:41
03:00 03:25

Output:

3

===========

Input:

2
15:10 15:20
15:20 15:30

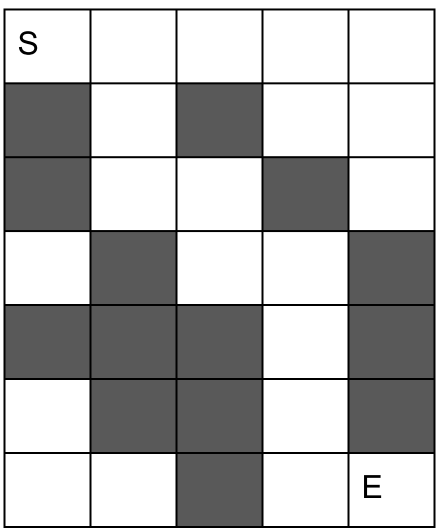
Output:

2

Problem 3: Emergency Escape

A researcher is examining new emergency safety features for a building. In the event of an emergency in the new building, a robot will be used to guide employees to safety. A robot is placed at a given location in a room. Upon activation of the emergency alert system, the robot's light flashes and it provides an audible alert to lead employees the floor's exit.
During an emergency, the room may be cluttered with debris from several sources. The building floor is represented as an N*N binary matrix of blocks. Your job is to help the research begin the prototype phase for his rescue robot by first programming it to solve a maze to demonstrate proof of concept. For proof of concept purposes, the robot will always start at a source at 0,0 and must reach the exit at N-1, N-1. The robot can move up, down, left, and right.
In the maze matrix, 0 means the block is dead end and 1 means the block can be used in the path from the robot's start position (S) to the room's exit (E).
The following is an example maze.

Gray blocks are dead ends (value = 0).



The following is a binary matrix representation of the above maze.

1 1 1 1 1
0 1 0 1 1
0 1 1 0 1
1 0 1 1 0
0 0 0 1 0
1 0 0 1 0
1 1 0 1 1

(continue on next page)

As stated previously, the **S** represents the robot's start position and the **E** represents the room's exit location.

Input Format:
The first line contains the number that represents the number of trials in the exercise. The next line of input represents the number of rows and the number of columns in the room array. The remainder of the trial is the binary input of the room separated by spaces.

Output Format:
Your output should be a solution matrix of the same size. Each matrix will be separated by a blank line.

Example:

Input:

```
2
4 4
1 0 0 0
1 1 0 1
0 1 0 0
1 1 1 1
7 5
1 1 1 1 1
0 0 0 0 1
1 1 1 0 1
1 0 1 1 1
1 0 0 0 0
1 0 1 1 1
1 1 1 0 1
```

Output:

```
1  0  0  0
1  1  0  0
0  1  0  0
0  1  1  1

1  1  0  0  0
0  1  0  0  0
0  1  1  0  0
0  0  1  1  0
0  0  0  1  0
```

Problem 4: Pythagorean Triples

A Pythagorean triple is a sequence x, y, and z such that $x < y < z$ and $x^2 + y^2 = z^2$. Your job is to find Pythagorean triples hidden in a set of integers.

Input Format
The first line contains a number N which the number of instances of the problem. Each following line of input will be one instance of the problem. It is a space-separated sequence of unique integers not necessarily in any particular order.

Output Format
For each line of input, you will output all of the Pythagorean triples in the input sequence in the format {x, y, z} with x < y < z. If there are multiple triples in the line, they should be space-separated and sorted by x with ties being sorted by y. Note that one integer in the input can be used in multiple triples in the output.

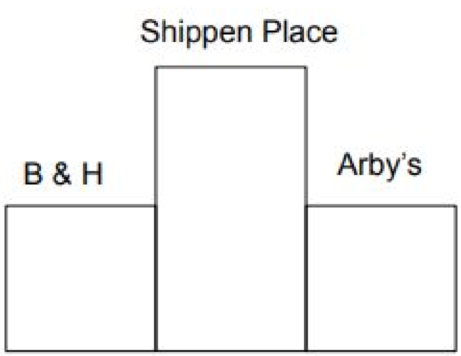
Example:
============
Input:
2
5 10 3 6 4
6 4 3 5 13 21 12

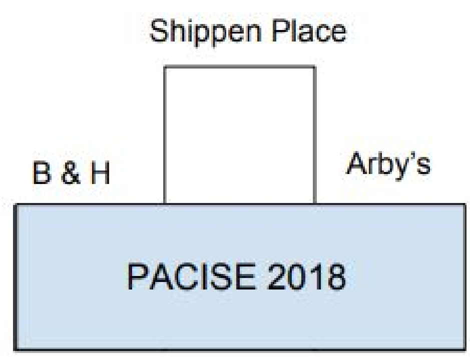
Output:
{3, 4, 5}
{3, 4, 5} {5, 12, 13}

Problem 5: PACISE Poster on King Street, Shippensburg

The chair of PACISE 2018 wants to hang a PACISE poster on King Street to promote the conference. The poster should be kept on the side of the street by being hung on the walls of the buildings. The poster must be rectangular in shape. Any part of the poster must be on the wall of a building. According to the city code, each building on King Street has the same height for each storey. The width of each building is the same as the height of a storey. Each building may have different number of storeys. Each building is rectangular in shape. Given a segment of the street, the chair wants to figure out what could the maximum size of the poster be. (The unit of the size will then be height_of_storey^2.)

For example, consider the following segment of King Street: Arby's, Shippen Place, B&H. The Arby's is one-storey high. Shippen Place two. B&H one. Then the maximum size of the poster shall be 3. Shown as following:





Input Format
The input file will have 2 lines. The first line contains the number N which is the number of buildings. The following line contains the heights (in storeys) of the N buildings, separated by a space.

Output Format
A single integer which is the maximum size of the poster.

Example:
Input:
6
2 1 5 6 2 3

Output:
10