Round D APAC Test

A. Cube IV

B. GBus count

C. Sort a scrambled itinerary

D. Itz Chess

Questions asked 4

Submissions

Cube IV

8pt	Not attempted **1708/2380 users** correct (72%)
15pt	Not attempted **1492/1679 users** correct (89%)

GBus count

9pt	Not attempted **2048/2354 users** correct (87%)
15pt	Not attempted **1865/2018 users** correct (92%)

Sort a scrambled itinerary

11pt	Not attempted **1623/1914 users** correct (85%)
15pt	Not attempted **1483/1602 users** correct (93%)

Itz Chess

12pt	Not attempted **654/1008 users** correct (65%)
15pt	Not attempted **393/622 users** correct (63%)

Top Scores

dreamoon	100
Kriiii	100
Balajiganapathi	100
uws933	100
NExPlain	100
culaucon	100
fahimzubayer18	100
pattara.s	100
buaamm	100
lijiancheng	100

Problem A. Cube IV

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 8 points	Solve A-small
Large input 15 points	Solve A-large

Problem

Vincenzo decides to make cube IV but only has the budget to make a square maze. Its a perfect maze, every room is in the form of a square and there are 4 doors (1 on each side of the room). There is a big number written in the room. A person can only move from one room to another if the number in the next room is larger than the number in his current room by 1. Now, Vincenzo assigns unique numbers to all the rooms (1, 2, 3, ... S^2) and then places S^2 people in the maze, 1 in each room where S is the side length of the maze. The person who can move maximum number of times will win. Figure out who will emerge as the winner and the number of rooms he will be able to move.

Input

The first line of the input gives the number of test cases, **T**. **T** test cases follow. Each test case consists of **S** which is the side length of the square maze. Then S^2 numbers follow like a maze to give the numbers that have been assigned to the rooms.

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

Output

For each test case, output one line containing "Case #x: r d", where x is the test case number (starting from 1), r is the room number of the person who will win and d is the number of rooms he could move. In case there are multiple such people, the person who is in the smallest room will win.

Limits

1 ≤ **T** ≤ 100.

Small dataset

1 ≤ **S** ≤ 10

Large dataset

1 ≤ **S** ≤ 10^3.

Sample

Input	Output

2
2
3 4
1 2
 	
3
1 2 9
5 3 8
4 6 7
	Case #1: 1 2
	Case #2: 6 4

Round D APAC Test

A. Cube IV

B. GBus count

C. Sort a scrambled itinerary

D. Itz Chess

Questions asked 4

Submissions

Cube IV	
8pt	Not attempted **1708/2380 users** correct (72%)
15pt	Not attempted **1492/1679 users** correct (89%)
GBus count	
9pt	Not attempted **2048/2354 users** correct (87%)
15pt	Not attempted **1865/2018 users** correct (92%)
Sort a scrambled itinerary	
11pt	Not attempted **1623/1914 users** correct (85%)
15pt	Not attempted **1483/1602 users** correct (93%)
Itz Chess	
12pt	Not attempted **654/1008 users** correct (65%)
15pt	Not attempted **393/622 users** correct (63%)

Top Scores

dreamoon	100
Kriiii	100
Balajiganapathi	100
uws933	100
NExPlain	100
culaucon	100
fahimzubayer18	100
pattara.s	100
buaamm	100
lijiancheng	100

Problem B. GBus count

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 9 points	Solve B-small
Large input 15 points	Solve B-large

Problem

There exists a straight line along which cities are built.

Each city is given a number starting from 1. So if there are 10 cities, city 1 has a number 1, city 2 has a number 2,... city 10 has a number 10.

Different buses (named GBus) operate within different cities, covering all the cities along the way. The cities covered by a GBus are represented as 'first_city_number last_city_number' So, if a GBus covers cities 1 to 10 inclusive, the cities covered by it are represented as '1 10'

We are given the cities covered by all the GBuses. We need to find out how many GBuses go through a particular city.

Input

The first line contains the number of test cases (**T**), after which **T** cases follow each separated from the next with a **blank** line.
For each test case,
The first line contains the number of GBuses.(**N**)
Second line contains the cities covered by them in the form
$\mathbf{a_1}$ $\mathbf{b_1}$ $\mathbf{a_2}$ $\mathbf{b_2}$ $\mathbf{a_3}$ $\mathbf{b_3}$**...**$\mathbf{a_n}$ $\mathbf{b_n}$
where GBus1 covers cities numbered from a_1 to b_1, GBus2 covers cities numbered from a_2 to b_2, GBus3 covers cities numbered from a_3 to b_3, upto **N** GBuses.
Next line contains the number of cities for which GBus count needs to be determined (**P**).
The below **P** lines contain different city numbers.

Output

For each test case, output one line containing "Case #T_i:" followed by **P** numbers corresponding to the number of cities each of those **P** GBuses goes through.

Limits

1 <= **T** <= 10
$\mathbf{a_i}$ and $\mathbf{b_i}$ will always be integers.

Small dataset

1 <= **N** <= 50
1 <= $\mathbf{a_i}$ <= 500, 1 <= $\mathbf{b_i}$ <= 500
1 <= **P** <= 50

Large dataset

1 <= **N** <= 500
1 <= $\mathbf{a_i}$ <= 5000, 1 <= $\mathbf{b_i}$ <= 5000
1 <= **P** <= 500

Sample

```
Input

2
4
15 25 30 35 45 50 10 20
2
15
25

10
10 15 5 12 40 55 1 10 25 35 45 50 20 28 27 35 15 40 4 5
3
5

10
27
```

Output

```
Case #1: 2 1
Case #2: 3 3 4
```

Explanation for case 1:
2 GBuses go through city 15 (GBus1 [15 25] and GBus4 [10 20])
1 GBus goes through city 25 (GBus1 [15 25])

Submissions

Cube IV

8pt	Not attempted **1708/2380 users** correct (72%)
15pt	Not attempted **1492/1679 users** correct (89%)

GBus count

9pt	Not attempted **2048/2354 users** correct (87%)
15pt	Not attempted **1865/2018 users** correct (92%)

Sort a scrambled itinerary

11pt	Not attempted **1623/1914 users** correct (85%)
15pt	Not attempted **1483/1602 users** correct (93%)

Itz Chess

12pt	Not attempted **654/1008 users** correct (65%)
15pt	Not attempted **393/622 users** correct (63%)

Top Scores

dreamoon	100
Kriiii	100
Balajiganapathi	100
uws933	100
NExPlain	100
culaucon	100
fahimzubayer18	100
pattara.s	100
buaamm	100
lijiancheng	100

Problem C. Sort a scrambled itinerary

This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input 11 points	Solve C-small
Large input 15 points	Solve C-large

Problem

Once upon a day, Mary bought a one-way ticket from somewhere to somewhere with some flight transfers.

For example: SFO->DFW DFW->JFK JFK->MIA MIA->ORD.

Obviously, transfer flights at a city twice or more doesn't make any sense. So Mary will not do that.

Unfortunately, after she received the tickets, she messed up the tickets and she forgot the order of the ticket.

Help Mary rearrange the tickets to make the tickets in correct order.

Input

The first line contains the number of test cases **T**, after which **T** cases follow.
For each case, it starts with an integer **N**. There are **N** flight tickets follow.
Each of the next 2 lines contains the source and destination of a flight ticket.

Output

For each test case, output one line containing "Case #x: itinerary", where **x** is the test case number (starting from 1) and **itinerary** is sorted list of flight tickets which represents the actual itinerary. Each flight segment in the itinerary should be outputted as pair of source-destination airport codes.

Limits

1 ≤ **T** ≤ 100.
For each case, the input tickets are messed up from an entire itinerary bought by Mary. In other words, it's ensured can be recovered to a valid itinerary.

Small dataset

1 ≤ **N** ≤ 100.

Large dataset

1 ≤ **N** ≤ 10^4.

(The segment for second case in sample can be seen as below) MIA-ORD, DFW-JFK, SFO-DFW, JFK-MIA

Sample

```
Input    Output

2        Case #1: SFO-DFW
1        Case #2: SFO-DFW DFW-JFK JFK-MIA MIA-ORD
SFO
DFW
4
MIA
ORD
DFW
JFK
SFO
DFW
JFK
MIA
```

Round D APAC Test

A. Cube IV

B. GBus count

C. Sort a scrambled itinerary

D. Itz Chess

Questions asked 4

− Submissions

Cube IV

8pt Not attempted
1708/2380 users
correct (72%)

15pt Not attempted
1492/1679 users
correct (89%)

GBus count

9pt Not attempted
2048/2354 users
correct (87%)

15pt Not attempted
1865/2018 users
correct (92%)

Sort a scrambled itinerary

11pt Not attempted
1623/1914 users
correct (85%)

15pt Not attempted
1483/1602 users
correct (93%)

Itz Chess

12pt Not attempted
654/1008 users
correct (65%)

15pt Not attempted
393/622 users
correct (63%)

− Top Scores

dreamoon	100
Kriiii	100
Balajiganapathi	100
uws933	100
NExPlain	100
culaucon	100
fahimzubayer18	100
pattara.s	100
buaamm	100
lijiancheng	100

Problem D. **Itz Chess**

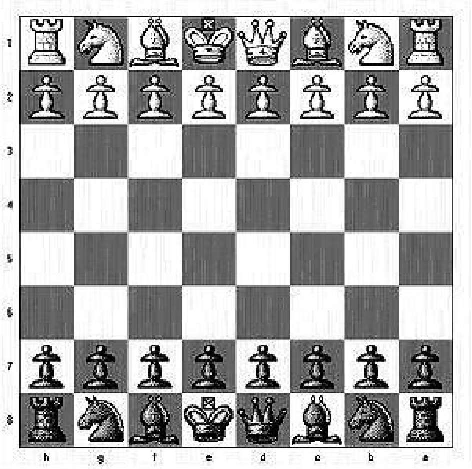
This contest is open for practice. You can try every problem as many times as you like, though we won't keep track of which problems you solve. Read the Quick-Start Guide to get started.

Small input
12 points
Solve D-small

Large input
15 points
Solve D-large

Problem

Given an arranged chess board with pieces, figure out the total number of different ways in which any piece can be killed **in one move**. Note: in this problem, the pieces can be killed despite of the color.



For example, if there are 3 pieces King is at B2, Pawn at A1 and Queen at H8 then the total number of pieces that an be killed is 3. H8-Q can kill B2-K, A1-P can kill B2-K, B2-K can kill A1-P

A position on the chess board is represented as A1, A2... A8,B1.. H8

Pieces are represented as

- (K) King can move in 8 direction by one place.
- (Q) Queen can move in 8 direction by any number of places, but can't overtake another piece.
- (R) Rook can only move vertically or horitonzally, but can't overtake another piece.
- (B) Bishop can only move diagonally, but can't overtake another piece.
- (N) Knights can move to a square that is two squares horizontally and one square vertically **OR** one squares horizontally and two square vertically.
- (P) Pawn can only kill by moving diagonally upwards (towards higher number i.e. A -> B, B->C and so on).

Input

The first line of the input gives the number of test cases, **T**. **T** Test cases follow. Each test case consists of the number of pieces , **N**. **N** lines follow, each line mentions where a piece is present followed by **-** with the piece type

Output

For each test case, output one line containing "Case #x: y", where x is the test case number (starting from 1) and y is the the total number of different ways in which any piece can be killed.

Limits

1 ≤ **T** ≤ 100.

Small dataset

1 ≤ **N** ≤ 10.
Pieces can include K, P

Large dataset

1 ≤ **N** ≤ 64.

Sample

```
Input   Output

2       Case #1: 1
2       Case #2: 3
A1-K
A8-Q

3
B2-K
A1-P
H8-Q
```