Google Cup 2011 ACM-ICPC China Shanghai Fudan Invitational Programming Contest

Celebration for the 106th Anniversary of Fudan University (1905-2011)

Contest Date: May. 15, 2011

A. Avaricious Maryanna

[Description]

After Maryanna found the treasure buried by 27 pilots in a secret cave, she wanted to leave there immediately. Unfortunately, finding the door closed because of the overweight treasure she carried, she had to find out the password of the lock. She remembered someone had told her the password is an N-digit natural decimal integer (without extra leading zeroes), and the N least significant digits of any positive integral power of that integer is same as itself. She, being a smart girl, came up with all the possible answers within 1 minute. After a few times of tries, she escaped from that cave successfully. To show your intelligence you may solve the same task with your computer within only 10 seconds!

[Input]

The first line contains **T** (**T** <= 1000), the number of test cases. T lines follow, each contains a single positive integer **N** (**N** <= 500).

[Output]

For each test case, output a single line, which should begin with the case number counting from 1, followed by all the possible passwords sorted in increasing order. If no such passwords exist, output "Impossible" instead. See the sample for more format details.

Sample Input	Sample Output
2 2 1	Case #1: 25 76 Case #2: 0 1 5 6

[Problem Setter]

Guodong Feng

B. Boring Homework

[Description]

Professor Z. always gives his students lots of boring homework. Last week, after explaining binary search trees (BSTs), he asked his students to draw a picture of BST according to the list of numbers inserted into the tree sequentially. Maryanna spent so much time playing the game "Starcraft II" that she can't finish her homework in time. She needs your help.

A binary search tree, which may sometimes also be called ordered or sorted binary tree, is a node-based binary tree data structure which has the following properties:

The left subtree of a node contains only nodes with keys less than the node's key.

The right subtree of a node contains only nodes with keys greater than the node's key.

Both the left and right subtrees must also be binary search trees.

--from Wikipedia

To draw a picture of BST, you may follow the rules listed below:

1. The picture of a 1-node BST, whose size is 1*1, is a single 'o' (15th small Latin letter).
2. If a BST has a non-empty subtree, draw a single '|' just above the subtree's root, and a single '+' just above previous drawn '|'. Finally, in the row of '+', use the least number (including 0) of '-'s to connect '+' (denoting the left subtree and right subtree) and 'o' (denoting the parent node of the subtree)
3. The left subtree (if exists) must be drawn on the left side of its parent. Similarly, the right subtree (if exists) must be drawn on the right side of its parent.
4. The column of the BST's root must not contain any character from left subtree or right subtree.
5. Any column containing any characters from BST's left subtree must not contain any characters from BST's right subtree, and vice versa. That is, for a node of the BST, the picture of its left subtree and the picture of its right subtree do not share common columns in the picture of the whole tree.

The sample output may give a clear clarification about the format of the picture.

[Input]

The first line contains **T** (**T** <= 2500), the number of test cases. T lines follow. Each line contains a positive integer **N** (**N** < 80), followed by **N** integers - a permutation of 1 to **N**. The permutation indicates the insert order for the BST.

[Output]

For each test case:

Output the case number counting from 1 in the first line. The next lines should be the image described above without any trailing spaces. See the sample for more format details.

Notice that no trailing whitespaces after the last visible characters of each line are allowed.

Sample Input	Sample Output
3 3 3 1 2 6 4 5 6 1 3 2 5 3 4 5 2 1	Case #1: +-o | o+ | o Case #2: +--o+ | | o-+ o+ | | +o o | o Case #3: +o+ | | +o o+ | | o o

[Problem Setter]

Ji Hong, Guodong Feng

C. Complete the Set

[Description]

Archaeologists have made a discovery on the Temple of Topology. The temple was once used as a place for ritual ceremony thousands of years ago. Among the relics that were unearthed, a scroll of parchment raised the interest of scientists. The parchment contained many numbers written in ancient symbols.

By decrypting the words carved on a stone, scientists know that these numbers form an interesting set of integers satisfying the following two properties:

1. Bitwise AND any number of integers from the set result in an integer in that set again.
2. Bitwise OR any number of integers from the set result in an integer in that set again.

As the parchment is extremely old, some part of it were broken and the numbers were lost. Now you job is to complete the original set from the remaining integers such that the size of the set is as small as possible.

[Input]

The input contains several test cases. The total number of test cases is less than 1100. Each test case begins with a line containing an integer **n** (**n** > 1). The following line contains **n** integers **ai** (0 <= **ai** < 2^{16}), the remaining integers on the parchment. The integers are distinct.

[Output]

For each test case, output one line containing a single integer, the minimal number of additional integers to make the set complete. If these numbers are already a complete set, print 0.

Sample Input	Sample Output
4 5 0 1 3 5 7 2 2 4 3 3 7 11 3 1 2 4	Case #1: 0 Case #2: 2 Case #3: 1 Case #4: 5

[Problem Setter]

Aoxiang Cui

D. Detection of Extraterrestrial

[Description]

E.T. Inc. employs Maryanna as alien signal researcher. To identify possible alien signals and background noise, she develops a method to evaluate the signals she has already received. The signal sent by E.T is more likely regularly alternative.

Received signals can be presented by a string of small latin letters 'a' to 'z' whose length is **N**. For each **X** between 1 and **N** inclusive, she wants you to find out the maximum length of the substring which can be written as a concatenation of **X** same strings. For clarification, a substring is a consecutive part of the original string.

[Input]

The first line contains **T**, the number of test cases (**T** <= 200). Most of the test cases are relatively small. **T** lines follow, each contains a string of only small latin letters 'a' - 'z', whose length **N** is less than 1000, without any leading or trailing whitespaces.

[Output]

For each test case, output a single line, which should begin with the case number counting from 1, followed by **N** integers. The **X**-th (1-based) of them should be the maximum length of the substring which can be written as a concatenation of **X** same strings. If that substring doesn't exist, output 0 instead. See the sample for more format details.

Sample Input	Sample Output
2 arisetocrat noonnoonnoon	Case #1: 11 0 0 0 0 0 0 0 0 0 0 Case #2: 12 8 12 0 0 0 0 0 0 0 0 0

[Hint]

For the second sample, the longest substring which can be written as a concatenation of 2 same strings is "noonnoon", "oonnoonn", "onnoonno", "nnoonnoo", any of those has length 8; the longest substring which can be written as a concatenation of 3 same strings is the string itself. As a result, the second integer in the answer is 8 and the third integer in the answer is 12.

[Problem Setter]

Guodong Feng

E. Entertainment

[Description]

Maryanna and Lucyanna play tennis every Sunday afternoon during 10 years ago.

A tennis match is determined through 5 sets. Typically the first player to win 3 sets wins the match. A set consists of games, and games, in turn, consist of points.

A game consists of a sequence of points played with the same player serving. A game is won by the first player to have won at least four points in total and at least two points more than the opponent. The running score of each game is described in a manner peculiar to tennis: scores from zero to three points are described as "love", "fifteen", "thirty", and "forty" respectively. If at least three points have been scored by each player, and the scores are equal, the score is "deuce". If at least three points have been scored by each side and a player has one more point than his opponent, the score of the game is "advantage" for the player in the lead. During informal games, "advantage" can also be called "ad in" or "ad out", depending on whether the serving player or receiving player is ahead, respectively.

A set consists of a sequence of games played with service alternating between games, ending when the count of games won meets certain criteria. Typically, a player wins a set by winning at least six games and at least two games more than the opponent.

The first servers of two consecutive sets are different.

Maryanna has found out when she is the server, her winning probability of this point is M%; otherwise, Y%. She wants to know her winning probability of the whole match if she serves first.

[Input]

The first line contains **T** (**T** <= 10000), the number of test cases. T lines follow. Each line contains two space-separated positive integers **M**, **Y** (0 < **M**, **Y** < 100).

[Output]

For each test case, output a single line, which should begin with the case number counting from 1, followed by Maryanna's winning percentage accurate to 4 decimal places. See the sample for more format details.

Sample Input	Sample Output
2 50 50 51 51	Case #1: 50.0000% Case #2: 63.5654%

[Problem Setter]

Guodong Feng

F. Fudan Extracurricular Lives

[Background]

Students in Fudan University have various extracurricular lives. It seems to be widely known that many students in the Zhangjiang Campus prefer **Tractor**(a game of poker with four players, also named as "80-points" or "Upgrading"), but it is true that a considerable mass of students (especially in Department of Computer S&T) love another four player game, Mahjong, very much. Mahjong is one of the most historied multiplayer games in China. People in several provinces prefer this game most with various modified rules respectively.

To begin this problem, firstly, we shall introduce some basic concepts about the game:

- **Game Going**

The game of Mahjong is played with a special set of "Mahjong Tiles", which made by piece of wood or stones, each tile have a corresponding image on one face, which denotes particular signification. Initially, all tiles should be randomly shuffled and put into a pile with their (tiles') faces down, thus all their contents are hidden. After that, the participating players each get 13 tiles as there **Holding Tiles** (will be definitely explained and discussed later). Consequently, players come to the "Dealing Section" that everyone alternatively gets one tile from the hidden pile and includes it into this player's Holding Tiles, then chooses one tile from the current **Holding Tiles** to discard. This procedure will continue until there's no tile in the hidden pile or any player meet the **Win** condition (Practically, there are further modified rules such as "Bloody Battle", which pronounced as "Shyue-Chan" in Chinese, will break this traditional rule, but we do not take them into account in this problem).

- **Holding Tiles**

In the process of game, players should manage their own set of tiles; these tiles are considered as **Holding Tiles** of each player respectively. **Holding Tiles** of one player have two particular types: **Declared** and **Hidden**. **Declared Tiles** are shown into public while **Hidden Tiles** are kept private, thus one can know all the four players' **Declared Tiles** but only his or her own **Hidden Tiles**. The event causing **Hidden Tiles** into **Declared** will be discussed later.

- **Win**

Before introducing the concept of how to **Win** a game, we have to firstly give out two essential sub-concepts, please remember that they are quite important for solving this problem:

(a) **Architecture**

A player's Holding Tiles are just a set of tiles, they can be separated and assembled into special **Components**, and all these **Components** compose the **Architecture** of **Holding Tiles**.

(b) Analysis

The procedure and method to determine the **Architecture** of one's **Holding Tiles** is called **Analysis** (please ignore the changing part of speech). **Analyzing** one's **Holding Tiles** should be based on a system of related rules and might be somewhat complex and tricky, discussed later.

In the rules of Mahjong, the **Target Architecture** will be defined. As the name implies, if a player's **Holding Tiles** can be **Analyzed** into a particular **Architecture** coinciding the **Target Architecture**, then this player can **Win** the game.

- Score

After one player **Win** a game, the score of this **Win** should be judged under a scoring rules system. The score is used for determining the worth of this **Win** -- It is widely known that Mahjong is originally invited for gambling.

[Description]

Now in China, people in Sichuan Province prefer the game of Mahjong much more than people in other area; causally, most of students in Fudan University who prefer Mahjong come from Sichuan. To this reason, we would like to use the rule of Sichuan Mahjong to go on this problem. With related to the traditional Mahjong rule, regulation of this game in Sichuan Province have been changed most obviously in the following two ways:

- Tiles simplification

In the original rule of Chinese Mahjong, the game is based on a large set of tiles. All these tiles can be classified into several **Categories** -- including **Simple Category**, **Honor Category**, and **Bonus Category** -- with different functions when the game playing. The last two kinds of **Categories** are used in rules of some other area of China. (In other words, has nothing to do with this problem) Sichuan Mahjong only uses **Simple Category** in the game:

There are three different **Suits** in **Simple Category** and each **Suit** is numbered 1 to 9. They are **Dots**, **Bamboo** and **Myriads**.

Dots numbered 1 to 9.

Myriads numbered 1 to 9.

Bamboo numbered 1 to 9.

There are four matching tiles for each value (e.g. there are four Dots tiles with the number 2). Generally, there are two essential aspects of information for determining the **Kind** of a tile: the **Suit** and **number**, thus we can uniquely name a tile with its order of **number** and **Suit**, for example: name  as "**Third Bamboo**" and name  as "**First Dots**". Uniquely, players always call  (**First Bamboo**) as "Yao-Jhee". It is easy to calculate that there in total 108 tiles used in the game of Sichuan Mahjong, that is, there are nine kinds in each suit and four tiles of each **Kind**.

- **Two Suits Limitation**

This limitation constrains that at most two suits are permitted in one's **Holding Tiles**, that is, one with all the three suits in the **Holding Tiles** can never **Win** a game (Actually, would cause some penalty).

After clarifying these modifications, we can go on the description of Mahjong game.

During the process that every player gets new tiles alternatively, there are some situations which would cause case of **Interruption**. In detail, the rule of Mahjong allows players to meld ("meld" is a conventional glossary in Mahjong game, it is better to understand it as "**adopt**") tiles discarded by others and, after that, sometimes discard a **Holding Tiles** for exchange. In order to explain the concept of **Interruption** and the way to **Win** the game, we shall now define some concepts of **components** (we've mentioned before but never defined) of **Holding Tiles**. Please review the concept of **Target Architecture**, a **Target Architecture** is composed by a set of **Components**, and there are two kinds of **Components**:

- Sentence

There are two kinds of sentences:

(a) Sequence

A **Sequence** is a triple of tiles with sequential number, for example:

 and 

There must be no skipping of numbers, nor does 9 loop around to 1. **The sequence must be in the same Suit.**

(b) Chunk

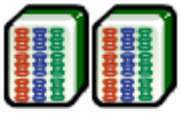
A **Chunk** is a triple of tiles with identical number, for example:

 and 

There **Chunk** must be in the same suit.

- **Pair**

  Similar to **Chunk**, A Pair is a couple of (two) tiles with identical number and same suit, see the following example:

 and 

With the definition of **Component**, there are three ways of melding others' discarded tiles, and each time if a player calls for a melding, then an **Interruption** was made. However, only two ways of **Interruption** are used in rule of Sichuan Mahjong:

- **Refine** (pronounced as "Pong" in Chinese)

  If someone discards a tile while you have a **Pair** of the same tile in your **Hidden Tiles**, you may (but not must) **Refine** the discarded tiles with the pair in your hand, **after that these three identical tiles should become Declared.** As we have mentioned before, each time you **Refine** a tile, you should discard another **Hidden Tile** for exchange.

- **Upgrade** (pronounced as "Kong" in Chinese)

  If you have got a sentence of **Chunk** in **Hidden Tiles** while another player discards the identical (in fact, the only one) tile -- or you get this only identical tile by yourself -- you may (but not must) **Upgrade** the **Chunk** into a **Component** with four same tiles (a "Kong") by melding the discarded tile. The same with **Refining**, after **Upgrading** these four identical tiles should become **Declared**. Specially, when making an **Upgrade**, you should never discard another **Hidden Tiles**. It seems that after an **Upgrade**, an extra tile is merged into one's **Holding Tiles** (Accurately, into **Hidden Tiles**), in fact these four same tiles should also be considered as a **Chunk**, the extra tile is only for "honor", which means being used for calculate the score but never used when Analyzing the Architecture.

According to the descriptions of **Refine** & **Upgrade**, we can consequently conclude that:

- **Declared Tiles** of a player always come from one or more melding **Interruption**s.
- Once a tile becomes **Declared**, it will never be back to **Hidden** and never be discarded any more.

Another type of **Interruption** is **Win** - Which has been defined above -- that is, once another player discards a particular tile, or you get such a tile by yourself, if you can merge such a tile into your **Holding Tiles** and then **Analyze** all the **Holding Tiles** (14 tiles now) as a **Target Architecture**, then you can **Win** this game. Now, we should give the detailed explanation of **Target Architecture** under the rule of Sichuan Mahjong. The **Target Architecture** of **Holding Tiles** can be considered as that:



- Following the Two Suits Limitation.
- **Component**s in the **Architecture** fits one of the following two structural styles:
  (a) Consist of FOUR **Sentence**s and ONE **Pair**, here we name the unique **Pair** as "Jong". We name this style as **Regular Target**.
  (b) Consist of SEVEN **Pair**s.

Remember that the **Declared Tiles** is a group of **Sentences** (an **Upgraded Chunk** can be considered as an ordinary **Chunk** with "honor" for scoring, discussed later). When **Analyzing** the **Holding Tiles**, the **Declared Tiles** and **Hidden Tiles** must be considered separately and can never be mixed; furthermore, **Declared Tiles** can only be **Analyzed** as several **Chunks** (ordinary **Chunk** or **Upgraded Chunk**) but never **Sequences.** To finish a **Target Architecture**, we need 14 tiles, (ignored extra tile from **Upgrading**), however, each one has only 13 **Holding Tiles**. Therefore, players always need another extra tile to complete the **Architecture**; this situation gives the concept of **Expected Tile**: If a player's **Holding Tiles** can fit the **Target Architecture** by adding a particular and **existing** tile, then such a tile is considered as one of the player's **Expecting Tiles**. Pay attention on the word "existing" please: that is, if you have held all such existing (four) tiles, then this kind of tile cannot be considered as **Expected Tile.**

Now we come to specify the rule of **Scoring**: once a player meets the **Target Architecture** (has got **Win**), he or she will have the basic score of ONE, that is, the lowest score of a **Win** is ONE. In addition, there are lots of specific conditions for awarding further extra scores. See the following items for description:

- **Dragon** (pronounced as "Kong" and "Gher" in Chinese, especially in Chengdu dialect)
  If a player has collected all the four tiles of same kind of tiles in his or her **Holding Tiles**, then we say this kind become a **Dragon**. Each Dragon gives the owner ONE additional score.
  Apparently, once a player making an **Interruption** of **Upgrade**, a **Declared Dragon** is created, but there's another situation: sometimes player shows a **Declared Chunk** in **Declared Tiles** but hides the rest one of that kind of tiles in **Hidden Tiles**, or even hides all the four same tiles in **Hidden Tiles**, these two cases will also be acknowledged as **Dragon**.

- **Seven Couples** (pronounced as "Chee Dwee" in Chinese)
  If the **Architecture** meets the second style of **Target Architecture** - in other words: consists by seven pairs - then such an **Architecture** can be identified as **Seven Couples**, this special style gives the owner TWO additional score.
- **Purely** (pronounced as "Ching Yi-Seh" in Chinese)
  If one players all **Holding Tiles** (Attention, not only the **Hidden** or **Declared Tiles**) have the same **Suit**, then **Purely** can be identified, TWO additional score

awarded.

- **Chunkious** (pronounced as "Dwee-Tsi Hoo" in Chinese)
  Chunkious is a special style of **Regular Target**, the condition is that all sentences of **Holding Tiles** are **Chunks**, **Sequence** never appears. **Chunkious** is worth ONE additional score.
- **Sky-ground Related** (pronounced as "Tai Yao-Jhew" in Chinese)
  This is a very unique and confusing rule in Sichuan Mahjong, even not all players in Sichuan admit this condition, but in this problem we take it into account for generalization.
  No matter what style of **Target Architectures**, the condition of **Sky-ground Related** asks each components in the **Architecture** have at least one tile has numbers of 1 or 9. For instance, the following components are acceptable:

  This special style gives the owner TWO additional scores.
- **Royal Chunkious** (pronounced as "Jong Dwee" in Chinese)
  As the name implies, the **Royal Chunkious** is a special style of **Chunkious**, thus if an Architecture does not meet the condition of **Chunkious**, it can never be judged as **Royal**. This style means in an Architecture of **Chunkious**, only tiles with numbers 2, 5 and 8 appear. As examples, the following components are valid:

  Royal Chunkious worth TWO **more** scores **than** normal **Chunkious**.
- There are some other items relating to Scoring such as Seizing ("Chiang Kong"), Lucky Flower ("Kong Hsiang-Hwa"), but are not considered in this problem.

In order to solve this problem, the concept of **CONSISTENCE** might be somewhat a critical logic. We've just listed all the special situation causing extra scores for the **Winning** player, you may come up with the idea that some of them can occur simultaneously, for example, the following **Holding Tiles** (has **Won**):

Hidden Tiles are: 

, and **Declared Tiles** are:  and 

.

This set of Holding Tiles can be obviously **Analyzed** as an **Architecture** which meeting

the condition of **Chunkious**, **Purely** and double **Dragon**s (just **Analyze** as how they are displayed in the example), and worth 1(basic) + 1(**Chunkious**) + 2(**Purely**) + 1(**Dragon**)*2 = 6 scores.

However, when we consider another example:

Hidden Tiles are (there is no **Declared Tiles** in this example):

At first glance, we can **Analyze** them as an **Architecture** fitting rule of **Seven Couples**, which is trivial; yet we can also **Analyze** them in another way (fitting the rule of **Sky-ground Related**):

This way, can we judge the score as 1(basic) + 2(**Seven Couples**) + 2(**Sky-ground Related**) = 5? The answer is negative. For explaining this paradox, please remind yourself to notice the definition of **Win**: a player gets Win means he or she can **Analyze** the **Holding Tiles** into a **Target Architecture**, when we calculate the score, we are in fact aiming at such an resulting **Architecture**, scilicet, scoring is based on a particular **Architecture** but not based on the set of **Holding Tiles**, that is, if a set of **Holding Tiles** can be **Analyzed** as more than one **Target Architectures**, they are independent when scoring and cannot merge. Now we can give the correct answer of the last example: if **Analyze** in first way, then 1(basic) + 2(**Seven Couples**) = 3; otherwise, if use the second way, then 1(basic) + 2(**Sky-ground Related**) = 3, therefore the final score should be considered as 3. The last remaining confusion of **scoring** is that: if two or more **Analyzed Architecture** has different scores, which should be the delegate? The policy for dealing with this question is totally determined by real players, in this problem we choose the higher one - we choose the highest possible score among all **Analyzed Target Architecture**.

Your task in this problem is: given a player's **Holding Tiles**, please determine all this player's **Expected Tiles** (if exists) and their corresponding scores.

[Input]

The first line of input file is a integer T (T <= 100) indicating number of test cases, then T lines are following. Each line describes a case of **Holding Tiles**: two

separated strings, the first string giving the **Hidden Tiles** and the second one giving the **Declared Tiles**. If there are no **Declared** ones, the second string would be "NONE" (quotes for clarity). Each tile contains two characters, the first one is a digit character which indicates this tile's number and the second is a capital letter within 'D','B' and 'M', that 'D' for **Dots** and 'B' for **Bamboo** and 'M' for **Myriads**, indicating tile's **Suit**. For example, "7M" means **Seventh Myriad** and "1B" means **First Bamboo** (also known as "Yao-Jhee"). You may assume that all the input cases are valid in the rule of Sichuan Mahjong.

[Output]

For each case, firstly output a single line containing the case number counting from 1. Then, if there's no **Expected Tiles**, output a single lines containing string of "NONE" (quotes for clarity); otherwise, output all the player's **Expected Tiles** (with the same format of input) and then output their corresponding scores after each **Expected Tiles**, separated with a colon (":") and a space. All the **Expected Tiles** are printed in separated lines. Any extra blanks (spaces or empty lines) will cause "Wrong Answer". The outputted **Expected Tiles** should be sorted in this way: first sorted by **Suit**s, 'D' at first and then 'B' and 'M' in the last; second sorted by their numbers, smaller number has higher priority.

Sample Input	Sample Output
7 8D8D8D5D2D2D2D 6D6D6D7D7D7D 8D8D8D5D2D2D2D6D6D6D7D7D7D NONE 1D1D1D1D 3D3D3D4D4D4D5D5D5D 2D2D2D5D5D5D2M2M2M5M5M8M8M NONE 1D1D1D1D2D2D2D2D3D3D3D3D9M NONE 1D1D1D1D9D9D9D9D1M1M1M1M9M NONE 2D2D3D3D4D5D5D6D6D7D7D8D8D NONE	Case #1: 5D: 4 Case #2: 4D: 3 5D: 4 6D: 4 7D: 4 8D: 4 Case #3: NONE Case #4: 5M: 4 8M: 4 Case #5: 9M: 6 Case #6: 9M: 8 Case #7: 1D: 3 4D: 5

[Hints]

Consider the third case of sample input, it seems that adding a tile "1D" may construct the Target Architecture, however, there does not exist another "1D" in all the tiles.

[Problem Setter]

Ji Hong

G. Google is Feeling Lucky

[Description]

Google is one of the most famous Internet search engines which hosts and develops a number of Internet-based services and products. On its search engine website, an interesting button "I'm feeling lucky" attracts our eyes. This feature could allow the user skip the search result page and goes directly to the first ranked page. Amazing! It saves a lot of time.

The question is, when one types some keywords and presses "I'm feeling lucky" button, which web page will appear? Google does a lot and comes up with excellent approaches to deal with it. In this simplified problem, let us just consider that Google assigns every web page an integer-valued relevance. The most related page will be chosen. If there is a tie, all the pages with the highest relevance are possible to be chosen.

Your task is simple, given 10 web pages and their relevance. Just pick out all the possible candidates which will be served to the user when "I'm feeling lucky".

[Input]

The input contains multiple test cases. The number of test cases **T** is in the first line of the input file.

For each test case, there are 10 lines, describing the web page and the relevance. Each line contains a character string without any blank characters denoting the URL of this web page and an integer **Vi** denoting the relevance of this web page. The length of the URL is between 1 and 100 inclusively. (1 <= **Vi** <= 100)

[Output]

For each test case, output several lines which are the URLs of the web pages which are possible to be chosen. The order of the URLs is the same as the input.

Please look at the sample output for further information of output format.

Sample Input	Sample Output
2 www.youtube.com 1 www.google.com 2 www.google.com.hk 3 www.alibaba.com 10 www.taobao.com 5 www.bad.com 10 www.good.com 7 www.fudan.edu.cn 8 www.university.edu.cn 9 acm.university.edu.cn 10	Case #1: www.alibaba.com www.bad.com acm.university.edu.cn Case #2: www.alibaba.com

www.youtube.com 1 www.google.com 2 www.google.com.hk 3 www.alibaba.com 11 www.taobao.com 5 www.bad.com 10 www.good.com 7 www.fudan.edu.cn 8 acm.university.edu.cn 9 acm.university.edu.cn 10	

[Problem Setter]

Lei Huang

H. Herbicide

[Description]

Professor Z. has a courtyard beside his house. In the past, he loved to clean it and prune the flowers in his yard. However, with the JavaFF class taught by Professor Z. being offered, he indulged in assigning boring homework to the students and had no time for caring his yard. Consequently, weeds begin to grow up and then his yard becomes overgrown.

In the last weekend, after Professor Z. assigned a mass of boring homework again, he suddenly brought his yard to mind. And after he saw the weedy ground, he decided to remove the weeds. But he did not have much time to deal with the garden's problem because of the uncompleted plans of further boring homework in the next JavaFF class. He sprayed herbicide on the ground optionally and, strangely, herbicide was sprayed as several simple polygons on the ground. In order to determine whether he should continue his work or not, Professor Z. needed to know how many weeds were covered by the herbicide. Notice that we assume that the weeds were not killed by the herbicide applied before. In the other words, a single weed can be counted several times in different polygon of herbicide.

It seems that counting this number is quite a tough job. Then he asked Maryanna, one of his students in JavaFF class, for help. But Maryanna has no time because her boring homework had never been finished. Please help her!

[Input]

The very first line of input contains an integer T (T <= 100) indicating the number of test cases.

The first line of each test case is an integer N (N <= 1000), N is the number of weeds in Professor Z's yard. In the following N lines, each line has a pair of integers (Xi, Yi) denoting the coordinate of a weed (-10000 <= Xi, Yi <= 10000). You can assume all the coordinates are unique. The next line is an integer M (M <= 1000) which denotes the number of polygons covered by herbicide, all these polygon's vertices are located on one of the N weeds, then M polygons descriptions are following. Each polygon Pi has two lines, the first line of ith polygon has an integer Si (3 <= Si <= N) denoting the number of vertices in this polygon. The following line contains Si integers, describing the vertices of this polygon in clockwise or counter-clockwise order. In detail, each integer in this line is an index of the coordinates of the N weeds listed before. Notice that the indexes start from 1. See the sample input for further details.

[Output]

For each test case, output M lines, each line has a single integer indicating the number of weeds covered by the corresponding polygon. If a weed lies on the edge of a polygon, then we consider such a weed as being covered.

Sample Input	Sample Output
1 6 1 1 3 2 2 3 1 -1 -1 -2 -1 1 3 3 5 4 3 3 1 5 6 5 5 4 2 3 6	Case #1: 4 3 6

[Problem Setter]

Zhan Yu, Ji Hong

I. Imitation

[Description]

Iris is a student of ethology studying the animal behavior. She is interested by the imitation behavior of animals. Imitation is an advanced behavior whereby an individual observes and replicates another's.

Iris is such a good researcher that she builds a mathematical model to describe the body figure of animals. She describes the body as a number of joints. And the figure or the status of animal body can be denoted as a set of ordered pairs of two different joints. To study the imitation of the animals, Iris defines the correlation of joints. She defines the correlation as the transitive closure of the ordered pairs of body status with its reflexive pairs eliminated. That is to say, the correlation is an anti-reflexive relation. In this case, we could say that one body status is imitating the other one when the correlations of both body statuses are the same.

For example, for the joint set {J1, J2, J3}. The first body status contains the ordered pair (J1, J2), (J2, J3). And the second body status contains the ordered pair (J1, J2), (J2, J3), (J1, J3). We could say that the first body status is imitating the other one because both of the correlation sets are (J1, J2), (J2, J3), (J1, J3) since the definition of the correlation is the transitive closure of body status.

For a given body status, that is, a given set of ordered pairs of joints, Iris want to get another body status, which is imitating the given one. At the same time, the desired body status must contain the minimum number of ordered pairs or the maximum number of ordered pairs. Your task is to calculate the minimum number and the maximum number.

[Input]

There are several test cases. The first line of the input contains a single integer T denoting the number of test cases. There are about 100 test cases, but 90% of them are relatively small.

For each test case, the first line contains two integers **N** and **M** where **N** is the number of joints of both the given body status and the desired body status, **M** is the number of ordered pairs of the given body status. (1 <= **N** <= 1000, 0 <= **M** <= 10000)

Next **M** lines, each line denoting an ordered pair (**Xi**, **Yi**). The **Xi** and **Yi** are integers between 1 and **N**. Note that we consider the joints as the number between 1 and **N**.

[Output]

For each test case, output two integers denoting the minimum and maximum number of ordered pairs of the desired set. Two integers are separated by a single whitespace.

Sample Input	Sample Output
3 3 3 1 2 2 3 1 3 3 3 1 2 2 3 3 1 9 9 1 2 2 3 3 1 4 5 5 6 6 4 7 8 8 9 9 7	Case #1: 2 3 Case #2: 3 6 Case #3: 9 18

[Hint]

In mathematics, the transitive closure of a binary relation R on a set X is the transitive relation R+ on set X such that R+ contains R and R+ is minimal. If the binary relation itself is transitive, then the transitive closure will be that same binary relation; otherwise, the transitive closure will be a different relation.

A relation R on a set X is transitive if, for all x, y, z in X, whenever x R y and y R z then x R z. Examples of transitive relations include the equality relation on any set, the "less than or equal" relation on any linearly ordered set, and the relation "x was born before y" on the set of all people. Symbolically, this can be denoted as: if x < y and y < z then x < z.

(From Wikipedia::transitive closure)

[Problem Setter]

Lei Huang, Guodong Feng

J. Juice Extractor

[Description]

Jerry loses himself in the interesting game: Fruit Ninja. Fruit Ninja is a game of iPhone and iPad in which the players cut the fruits coming from the bottom of the screen and gain the bonus from cutting more than two fruits with a single slice. Once a fruit is cut, it breaks into small pieces and cannot be cut any more.

After months of training, he becomes pro of this game. Actually, he can cut all the fruits on the screen at any time. Jerry also has a bad habit that he has no willing to leave some fruits for the future cutting. In the other words, after Jerry cuts the fruits, all the fruits on the screen breaks and no one left. That is why all his friends call him "Juice Extractor".

Now he only consider about the bonus, when he cuts more than two fruits, he can gain some bonus scores as same as the number of fruits he slice at that time. For example, if Jerry cuts 4 fruits with a single slice, he can get 4 scores from this slice.

After Jerry gets the fruit schedule, he knows the appearing time and the disappearing time for every single fruit. He can only cut a fruit into pieces between its appearing time and disappearing time inclusive. He wants to know the maximum possible bonus scores he can receive.

[Input]

There are several test cases; the first line of the input contains a single integer T, denoting the number of the test cases. (**T** <= 200)

For each test case, the first line contains an integer **N**, denoting the total number of fruits. (1 <= **N** <= 1000)

The next **N** lines, each line describe a fruit. For each line, there are two integers **Xi** and **Yi**, where **Xi** is the appearing time of the fruit and Yi is the disappearing time of this fruit. (0 <= **Xi** <= **Yi** <= 1000000000)

[Output]

For each test case, output a single integer denoting the maximum scores that Jerry could possibly gain. See the sample for further details.

Sample Input	Sample Output
1 10 1 10 2 11 3 12 4 13 13 14	Case #1: 10

14 15 13 19 20 22 21 23 22 24	

[Problem Setter]

Lei Huang