ANZAC League
Round 1
2014

Problem Set

Compiled by Malcolm Corney
Queensland University of Technology

Notes:

1) Problems in this set have a **2 second time limit**.

2) For problems requiring a floating point answer, maintain an absolute error of less than 1E-6 to ensure accuracy with the expected answer unless otherwise indicated.

3) All of these problems have been sourced from different places on the Internet. We do not reveal their identity here, but none-the-less, credit, authorship, intellectual property, copyright, etc, remains with the original problem setters.

ANZAC 2014 - Round 1

Problem A: What Does the Fox Say?

Determined to discover the ancient mystery – the sound that the fox makes – you went into the forest, armed with a very good digital audio recorder. The forest is, however, full of animals' voices, and on your recording, many different sounds can be heard. But you are well prepared for your task: you know exactly all the sounds which other animals make. Therefore the rest of the recording – all the unidentified noises – must have been made by the fox.

Input

The first line of input contains the number of test cases T. The descriptions of the test cases follow:

The first line of each test case contains the recording – words over lower case English alphabet, separated by spaces. Each contains at most 100 letters and there are no more than 100 words. The next few lines are your pre-gathered information about other animals, in the format <**animal**> **goes** <**sound**>. There are no more than 100 animals, their names are not longer than 100 letters each and are actual names of animals in English. There is no **fox goes ...** among these lines.

The last line of the test case is exactly the question you are supposed to answer: *what does the fox say?*

Output

For each test case, output one line containing the sounds made by the fox, in the order from the recording. You may assume that the fox was not silent (contrary to popular belief, foxes do not communicate by Morse code).

Sample Input

```
1
toot woof wa ow ow ow pa blub blub pa toot pa blub pa pa ow pow toot
dog goes woof
fish goes blub
elephant goes toot
seal goes ow
what does the fox say?
```

Sample Output

```
wa pa pa pa pa pa pow
```

ANZAC 2014 - Round 1

Problem B: Influence

In any society there are social influence relations between individuals, where a person `x` can influence another person `y`. The same is true for Softopia, a special society where social influence relations are also transitive, meaning that if `x` can influence `y` and `y` can influence `z`, then `x` also influences `z`. Moreover, the rules of social influence guarantee that if `x` influences any other person `y`, then `x` cannot be influenced by `y` as well. Using these simple rules, if a person `x` from Softopia wants something, then all the nodes influenced by `x` also want the same thing.

Although, Softopia is almost a perfect society, there is a set of certain individuals, `X`, that would spread false demands through the social influence mechanism used by the society. And there is also an evil entity that would want to find out which of these people should be selected to spread a demand of their own. Because the society can only select one person from `X`, it would like to select one that is able to influence as many people as possible from Softopia. If there is more than one person that satisfies this request, you should pick the one with the lowest identifier.

Input

The input starts with a line containing two integers separated by one space: n ($n \leq 5000$) the number of individuals in the society, and k, the number of elements in set `X`. The next line contains the elements from `X`, thus k different integers from $1..n$ separated by one space. Then follow n lines and each line $i, 1 \leq i \leq n$, contains first the identifier of the current person followed by the identifiers of the persons that can be directly influenced by person i, all of them separated by a space. The persons are labeled from 1 to n. The total number of influences in a society is less than 250000. Additional whitespaces in the input should be skipped.

Output

The result, representing the identifier of the person that satisfies the conditions mentioned above, will be written on a single line.

The table below describes two samples of input and output.

Sample Input	Sample Output
5 2 1 2 1 3 4 2 3 4 3 5 4 5 5	1
6 3 1 2 3 1 2 2 5 3 4 2 4 6 5 6	3

ANZAC 2014 - Round 1

Problem C: Fraud Busters

The number of cars in Default City that travel to the city center daily vastly exceeds the number of available parking spots. The City Council had decided to introduce parking fees to combat the problem of overspill parking on the city streets. Parking fees are enforced using an automated vehicle registration plate scanners that take a picture of the vehicle registration plate, recognize the sequence of digits and letters in the code on the plate, and check the code against a vehicle registration database to ensure that parking fees are dutifully paid or to automatically issue a fine to the vehicle owner otherwise.

As soon as parking fees were introduced, a parking fee fraud had appeared. Some vehicle owners had started to close one or several digits or letters on their vehicle registration plate with pieces of paper while they park, thus making it impossible for the current version of the automated scanner to recognize their vehicle's registration code and to issue them a fine.

The Default City Council had instituted the Fraud Busters Initiative (FBI) to design a solution to prevent this kind of fraud. The overall approach that FBI had selected is to expand the number of vehicle features that scanners recognize (including features like vehicle type and color), as well as excluding from the list any vehicles that are detected to be elsewhere at this time. This information should help to identify the correct vehicle by narrowing down the search in the vehicle registration database.

You are working for FBI. Your colleagues had already written all the complex pieces of the recognition software that analyses various vehicle features and provides you with a list of registration codes that might potentially belong to a scanned car. Your task it to take this list and a recognized code from the license plate (which may be partially unrecognized) and find all the registration codes that match.

Input

The first line of input contains 9 characters of the code as recognized by the scanner. Code that was recognized by the the scanner is represented as a sequence of 9 digits, uppercase English letters, and characters "$*$" (star). Star represents a digit or a letter that the scanner could not recognize.

The second line of input contains a single integer number n ($1 \leq n \leq 1000$) the number of vehicle registration codes from the vehicle registration database.

The following n lines contain the corresponding registration codes, one code per line. Vehicle registration codes are represented as a sequence of 9 digits and uppercase English letters. All codes on these n lines of the input file are different.

Output

On the first line of output write a single integer k ($0 \leq k \leq n$) the number of codes from the input that match the code that was recognized by the scanner. The code from the scanner matches the code from the database if the characters on all the corresponding positions in the codes are equal or the character from the scanner code is "$*$".

On the following k lines write the matching codes, one code per line, in the same order as they are given in the input.

Sample Input	**Sample Output**
A**1MP19* 4 A001MP199 E885EE098 A111MP199 KT7351TTB	2 A001MP199 A111MP199

ANZAC 2014 - Round 1

Problem D: Shopping Malls

We want to create a smartphone application to help visitors of a shopping mall and you have to calculate the shortest path between pairs of locations in the mall. Given the current location of the visitor and his destination, the application will show the shortest walking path (in meters) to arrive to the destination. The mall has N places in several floors connected by walking paths, lifts, stairs and escalators (automated stairs). Note that the shortest path in meters may involve using an escalator in the opposite direction. We only want to count the distance that the visitor has walked so each type of movement between places has a different cost in meters:

- If *walking* or taking the *stairs* the distance is the Euclidean distance between the points.
- Using the *lift* has a cost of 1 meter because once we enter the lift we do not walk at all. One lift can only connect 2 points. An actual lift connects the same point of different floors, in the map all the points connected by a lift have the corresponding edge. So you do not need to worry about that. For instance, if there are three floors and one lift at position (1,2) of each floor, the input contains the edges (0,1,2) $\rightarrow$ (1,1,2), (1,1,2) $\rightarrow$ (2,1,2) and (0,1,2) $\rightarrow$ (2,1,2). In some maps it can be possible that a lift does not connect all the floors, then some of the edges will not be in the input.
- The *escalator* has two uses:
  - Moving from A to B (proper direction) the cost is 1 meter because we only walk a few steps and then the escalator moves us.
  - Moving from B to A (opposite direction) has a cost of the Euclidean distance between B and A multiplied by a factor of 3.

The shortest walking path must use only these connections. All the places are connected to each other by at least one path.

Input

Input contains the map of a unique shopping mall and a list of queries.

The first line contains two integers N ($N \leq 200$) and M ($N - 1 \leq M \leq 1000$), the number of places and connections respectively. The places are numbered from 0 to $N - 1$. The next N lines contain floor and the coordinates x, y of the places, one place per line. The distance between floors is 5 meters. The other two coordinates x and y are expressed in meters.

The next M lines contain the direct connections between places. Each connection is defined by the identifier of both places and the type of movement (one of the following: `walking`, `stairs`, `lift`, or `escalator`). Check the cost of each type in the description above. The type for places in the same floor is walking. The next line contains an integer Q ($1 \leq Q \leq 1000$) that represents the number of queries that follow. The next Q lines contain two places each `a` and `b`. We want the shortest walking path distance to go from `a` to `b`.

Output

For each query write a line with the shortest path in walked meters from the origin to the destination, with each place separated by a space.

Sample Input	Sample Output
6 7 3 2 3 3 5 3 2 2 3 2 6 4 1 1 3 1 4 2 0 1 walking 0 2 lift 1 2 stairs 2 3 walking 3 4 escalator 5 3 escalator 4 5 walking 5 0 1 1 2 3 5 5 3 5 1	0 1 1 0 2 3 4 5 5 3 5 3 2 0 1

ANZAC 2014 - Round 1

Problem E: Grachten

Damn! I did not only oversleep (and today is the contest day!) but I also got stuck somewhere in Delft on my way from the hotel to the contest site. Everywhere around me are grachten, these city-canals that are part of many cities in the Netherlands. I am in a bit of hurry, because the contest starts in a few minutes.

To make matters even worse, some bridges in Delft are closed due to a cycling race through the city. Thus, I decided to jump over some of the grachten instead of searching for open bridges.

Everyone knows that computer scientists like me are good at algorithms but not very good athletes. Besides, I am a bit faint-hearted and don't want to get wet. So I need your help to calculate the distance I have to jump over a gracht.

Luckily, I did attend the excursion in Delft city center yesterday, where I learned that all paving stones in Delft are squares and have the same size. This way, I can do some measurements on my side of the gracht (my units are paving stones):

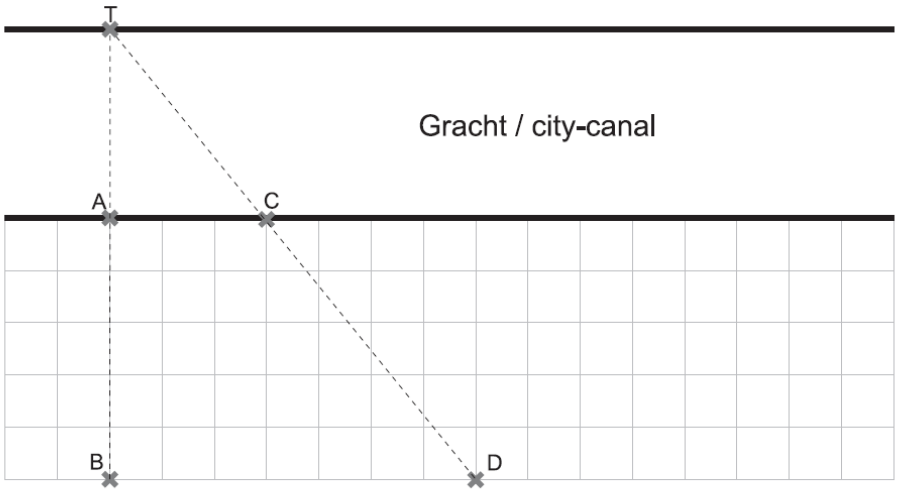

Figure 1 – Illustration of first sample input.

I walked from point C to point D via points A and B while counting the paving stones.

Points A and C are always on the edge of the gracht. Points B and D have the same distance to the gracht. The target point T is always on the edge of the other side of the canal; it is the intersection point of the line through B and A, and the line through D and C. The angle between AT and AC is 90 degrees, and the two edges of the canal are parallel lines.

Please calculate the distance between A and T (necessary jump distance) for me.

Input

For each test case, the input consists of one line containing three positive integers that specify the distances between A and B, A and C, and B and D.

You may safely assume that no distance is larger than 1000 and the distance between B and D is larger than the distance between A and C.

Output

For each test case, print one line of output: the distance between A and T as a reduced fraction (*i.e.* remove all common factors of numerator and denominator).

Sample Input	**Sample Output**
5 3 7 5 3 8 1 2 3 23 42 47 500 500 1000 1 1 1000	15/4 3/1 2/1 966/5 500/1 1/999

ANZAC 2014 - Round 1

Problem F: Digit Sum

When Grace was in third grade, her elementary school teacher assigned her the following problem:

> *What is the smallest possible sum of two numbers that together use the numerals 1, 2, 7, 8, and 9?*

Grace figured out that the answer to this problem is 207 (for example, as 78 + 129), but when the teacher assigned four pages of similar problems as homework, Grace got bored. It turns out that Grace was a rather advanced third grader, so she decided that it would be more fun to write a computer program to solve such problems. Surely you can do the same!

Input

Each problem is described on a single line. The line begins with an integer N, such that $2 \leq N \leq 14$, designating the number of numerals included in the problem. Following that are those N numerals. There will always be at least 2 numerals that are nonzero. The end of the input is designated by a line containing only the value 0.

Output

For each case, output a line with the minimum sum S that can be achieved. Please keep in mind that by standard convention, the numeral 0 cannot appear as the first digit of either summand.

Sample Input	Sample Output
5 1 2 7 8 9 6 3 4 2 2 2 2 9 0 1 2 3 4 0 1 2 3 0	207 447 11257

ANZAC 2014 - Round 1

Problem G: Cuckoo for Hashing

An integer hash table is a data structure that supports insert, delete and lookup of integer values in constant time. Traditional hash structures consist of an array (the *hash table*) of some size n, and a *hash function* $f(x)$ which is typically $f(x) = x \bmod n$. To insert a value x into the table, you compute its *hash value* $f(x)$ which serves as an index into the hash table for the location to store x. For example, if $x = 1234$ and the hash table has size 101, then 1234 would be stored in location $22 = 1234 \bmod 101$. Of course, it's possible that some other value is already stored in location 22 (x = 22 for example), which leads to a collision. Collisions can be handled in a variety of ways which you can discuss with your faculty advisor on the way home from the contest.

Cuckoo hashing is a form of hashing that employs two hash tables T_1 and T_2, each with its own hash function $f_1(x)$ and $f_2(x)$. Insertion of a value x proceeds as follows: you first try to store x in T_1 using $f_1(x)$. If that location is empty, then simply store x there and you're done. Otherwise there is a collision which must be handled. Let y be the value currently in that location. You replace y with x in T_1, and then try to store y in T_2 using $f_2(y)$. Again, if this location is empty, you store y there and you're done. Otherwise, replace the value there (call it z) with y, and now try to store z back in T_1 using $f_1(z)$, and so on. This continues, bouncing back and forth between the two tables until either you find an empty location, or until a certain number of swaps have occurred, at which point you rehash both tables (again, something to discuss with your faculty advisor). For the purposes of this problem, this latter occurrence will never happen, *i.e.*, the process should always continue until an empty location is found, which will be guaranteed to happen for each inserted value.

Given the size of the two tables and a series of insertions, your job is to determine what is stored in each of the tables.

(For those interested, cuckoo hashing gets its name from the behavior of the cuckoo bird, which is known to fly to other bird's nests and lay its own eggs in it alongside the eggs already there. When the larger cuckoo chick hatches, it pushes the other chicks out of the nest, thus getting all the food for itself. Gruesome but efficient.)

Input

Input for each test case starts with 3 positive integers n_1 n_2 m, where n_1 and n_2 are the sizes of the tables T_1 and T_2 (with $n_1,\ n_2 \leq 1000$ and $n_1 \neq n_2$) and m is the number of inserts. Following this will be m integer values which are the values to be inserted into the tables. All of these values will be non-negative. Each table is initially empty, and table T_i uses the hash function $f_i(x) = x \bmod n_i$. A line containing 3 zeros will terminate input.

Output

For each test case, output the non-empty locations in T_1 followed by the non-empty locations in T_2. Use one line for each such location and the form `i:v`, where `i` is the index location of the table, and `v` is the value stored there. Output values in each table from lowest index to highest. If either table is empty, output nothing for that table.

Sample Input	**Sample Output**

`5 7 4`
`8 18 29 4`
`6 7 4`
`8 18 29 4`
`1000 999 2`
`1000`
`2000`
`0 0 0`
	`Case 1:`
	`Table 1`
	`3:8`
	`4:4`
	`Table 2`
	`1:29`
	`4:18`
	`Case 2:`
	`Table 1`
	`0:18`
	`2:8`
	`4:4`
	`5:29`
	`Case 3:`
	`Table 1`
	`0:2000`
	`Table 2`
	`1:1000`

ANZAC 2014 - Round 1

Problem H: Generations of Tribbles

Tribbles are the cute, fuzzy, cuddly animals that have voracious appetites and reproduction rates that rival any complex organism in the galaxy (tribbles are born pregnant!). After being introduced to the Enterprise and its crew, it was quickly discovered what a nuisance tribbles could be. In a very short amount of time, tribbles were everywhere on the ship.

Fortunately for the Enterprise, Engineer Scott was able to transport them to a nearby Klingon vessel. The Klingons were unaware of the issues tribbles could cause and brought them into Klingon space, where the tribbles spread like locusts and devastated ecosystems of planets across the Klingon Empire.

Members of the United Federations of Planets (The Federation) found this extremely amusing and used the calculation of tribble reproduction as an academic exercise for first year students at its academy.

The following sequence of numbers represents how tribbles reproduce. The first number represents generation 0, the second generation 1, and so on.

1,1,2,4,8,15,29,56

The following recurrence can be used to represent the above sequence, where n represents the generation number:

$n < 2$: 1
$n = 2$: 2
$n = 3$: 4
$n > 3$: $\text{gen}(n-1) + \text{gen}(n-2) + \text{gen}(n-3) + \text{gen}(n-4)$

Those at the academy that know something about old Earth history have jokingly called the recurrence 'Tribblenacci'.

Your job as a first year student at the academy is to accurately and rapidly calculate how many tribbles there will be for a given 'Tribblenacci' number. The fact is, evaluating the above recurrence recursively is slower than chemical propulsion for interstellar travel! To do so for more than a handful of generations would clearly be illogical.

Input

The first line of input will be an integer t ($0 < t < 69$) representing the number of test cases. Following this will be t integer values, one per line. Each of these will represent a generation number g ($0 \leq g \leq 67$) to calculate.

Output

For each generation number read, display the corresponding 'Tribblenacci' value.

Sample Input	Sample Output
8 0 1 2 3 4 5 30 67	1 1 2 4 8 15 201061985 7057305768232953720

ANZAC 2014 - Round 1

Problem I: Exponential Towers

The number 729 can be written as a power in several ways: 3^6, 9^3 and 27^2. It can be written as 729^1, of course, but that does not count as a power. We want to go some steps further. To do so, it is convenient to use '^' for exponentiation, so we define $a \hat{} b = a^b$. The number 256 then can be also written as 2^2^3, or as 4^2^2. Recall that '^' is right associative, so 2^2^3 means 2^(2^3).

We define a *tower of powers of height k* to be an expression of the form $a_1 \hat{} a_2 \hat{} a_3 \hat{} \ldots \hat{} a_k$, with $k > 1$, and integers $a_i > 1$. Given a tower of powers of height 3, representing some integer n, how many towers of powers of height *at least* 3 represent n?

Input

The input file contains several test cases, each on a separate line. Each test case has the form $a \hat{} b \hat{} c$, where a, b and c are integers, $1 < a, b, c \leq 9585$.

Output

For each test case, print the number of ways the number $n = a \hat{} b \hat{} c$ can be represented as a tower of powers of height at least three.

The magic number 9585 is carefully chosen such that the output is always less than 263.

Sample Input	Sample Output
4^2^2 8^12^2 8192^8192^8192 2^900^576	2 10 1258112 342025379

ANZAC 2014 - Round 1

Problem J: The Urge to Merge

The Acme Consulting Group has sent you into a new technology park to enhance dynamism, synergy and sustainability. You're not sure what any of these terms mean, but you're pretty good at making money, which is what you plan on doing. The park consists of a 3 $\times n$ grid of facilities. Each facility houses a start-up with an inherent value. By facilitating mergers between neighboring start-ups, you intend to increase their value, thereby allowing you to fulfil your life-long dream of opening your own chain of latte-and-burrito shops.

Due to anti-trust laws, any individual merger may only involve two start-ups and no start-up may be involved in more than one merger. Furthermore, two start-ups may only merge if they are housed in adjacent facilities (diagonal doesn't count). The added value generated by a merger is equal to the product of the values of the two start-ups involved. You may opt to not involve a given start-up in any merger, in which case no added value is generated. Your goal is to find a set of mergers with the largest total added value generated. For example, the startup values shown in the figure on the left, could be optimally merged as shown in the figure on the right for a total added value of 171.

Input

The first line of each test case will contain a single positive integer $n \leq 1000$ indicating the width of the facilities grid. This is followed by three lines, each containing n positive integers (all ≤ 100) representing the values of each start-up. A line containing a single 0 will terminate input.

Output

For each test case, output the maximum added value attainable via mergers for that set of start-ups.

Sample Input	Sample Output
4 7 2 4 9 3 5 9 3 9 5 1 8 0	Case 1: 171

ANZAC 2014 - Round 1

Problem K: Zombie Invasion

A group of survivors has arrived by helicopter to an isolated island. The island is made up of a long narrow strip of villages. The infected survivors arrived in the village to the far east and accidentally infected the native population. The islanders are now attempting to escape the zombies that have appeared on the east coast.

You are given N cases with 20 non-negative integers that represent the number of islanders at a given village. The villages are represented from west to east (left to right, respectively), with the zombies moving in from the east. The islanders have peculiar customs for traveling and will only move between villages in pairs. Curiously, for every pair that travels between two villages, only one of them ever survives the trip. As the zombies move west, islanders will travel to the village immediately west of their current village as long as there are at least two islanders there. If there are an odd number people in a village then one stays in the village and the rest move to the next village in pairs. Once the islanders reach the village on the west coast, they will stop traveling. Determine how many islanders remain at each village and the number that make it safely to the village on the west coast (far left).

Input

The first line of data represents the number of data sets you will read in, N ($1 \leq N \leq 50$). There will then be N lines of twenty 20 non-negative integers each. Each integer (≤ 1000) represents the number of islanders who reside in a village. The leftmost integer represents the village on the west coast, and the rightmost integer represents the village on the east coast.

Output

Your output will be N lines of twenty 20 non-negative integers. The left most number will represent the number of islanders that reached the west. Each number to the right will represent the number of people that stayed behind in each village.

Sample Input

```
1
0 0 0 0 77 0 0 99 0 0 0 40 0 0 0 17 0 1 13 10
```

Sample Output

```
5 1 0 0 1 1 0 1 1 0 0 1 0 0 1 1 1 0 0 0
```

ANZAC 2014 - Round 1

Problem L: Trapezoid Walkway

You are planning to place a small, prefabricated gazebo in your back yard, with a paved walkway connecting it to your back porch. You'll build the walkway out of paving stones purchased at your local home improvement store. The paving stones come in a variety of sizes, but they are all shaped like isosceles trapezoids. As illustrated on the left of Figure 1, an isosceles trapezoid is what you get if you take an isosceles triangle and cut off a corner with a line that is parallel to the base. We can describe such a paving stone with three parameters: the length of one of its parallel edges, a, the length of the other parallel edge, b, and the perpendicular distance, h, between these edges.

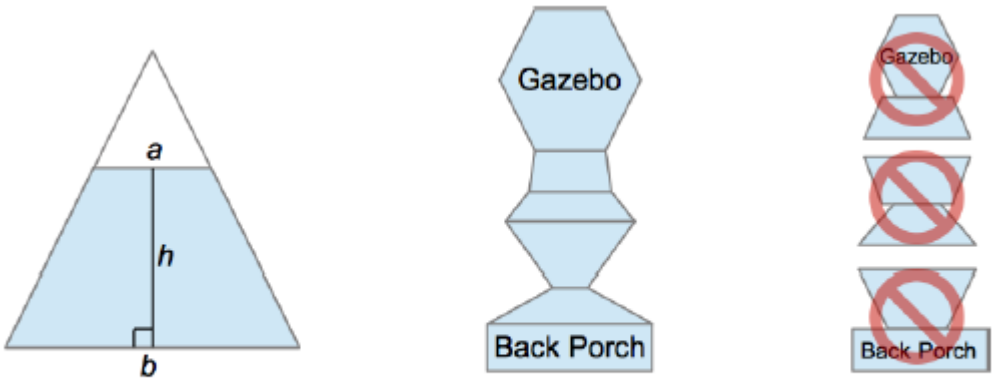

Figure 1: Isosceles trapezoid shape (left), walkway example (center) and join requirements (right)

You're going to build your walkway by joining the parallel edges of the paving stones, with one parallel edge meeting the back porch at the start and another meeting the gazebo at the end. You want your walkway to look nice, so you won't permit any of the situations illustrated on the right of Figure 1. Where two paving stones meet, their edges must be exactly the same length. Likewise, where a paving stone meets the back porch or the gazebo, its edge must be exactly the same length. Fortunately, your home improvement store has a wide selection of different trapezoid-shaped paving stones, so you are confident you can build a walkway that satisfies these requirements.

Paving stones are priced at two cents per square centimeter of surface area. You have a big back yard, so you don't care how long your walkway is. You just want the one that's the least expensive.

Input

Input will consist of multiple test cases. Each test case will start with a positive integer, n, the number of different types of paving stones available. The value, n, will be at least 1 and no greater than 1000. The next n lines each describe a type of stone by giving the lengths of its sides, a, b then h. These three values will be positive integers, measured in centimeters and not greater than 1000. No two types of stones are identical, but your home improvement store has a large stock of paving stones, so you can buy as many of each type as you need. The last line of each test case gives the width of the back porch, where

the walkway will start, followed by the width of the edge of the gazebo where the walkway will end. A value of zero for n will mark the end of all test cases.

Output

For each test case, print the total cost, in dollars, for the least expensive walkway that meets your requirements. It will always be possible to build such a walkway.

Sample Input	Sample Output
6 120 350 60 120 150 95 240 300 60 240 350 220 150 300 100 300 350 120 120 240 2 100 140 50 100 140 80 140 100 2 150 250 100 150 250 60 150 150 0	1030.50 120.00 0.00

ANZAC 2014 - Round 1

Problem M: Ternarian Weights

Back in the Hellenic era, there was a small island in the Mediterranean Sea known as Ternaria. It was close to Sparta, but because of its mountainous terrain the Spartans found it difficult to conquer and it remained an independent state until the great earthquake of 729BC when it sank beneath the sea. It had a remarkable civilization and some modern historians think it is the basis for the mythological city of Atlantis. Ternaria is still known for its fundamental contributions to science and mathematics, many of which were adopted by the Greeks and later by the Romans. For example, Ternarians were the first group to use the standard weight of pounds, which we still use today. Ternarian mathematics used base 3 for all its calculations. (Historians speculate that this was out of respect for King Ternary who lost two fingers on each hand while battling the Spartans.)

Ternarian trade scales were a standard for many centuries. They were known for their accuracy and ease of use. They were the first to construct a scale with weighing pans on each side and a fulcrum in the middle. The object to be weighed was placed on the left side of the scale and weights were placed on both sides, until balance was obtained. This sounds strange by modern standards, because typically on modern scales we would only place weights on the right side. However, the modern method requires additional weights. The Ternarian method only requires one weight for each power of three pounds, *e.g.* one weight of 1 pound, one weight of 3 pounds, one weight of 9 pounds, *etc.*

Say you are weighing a 2-pound Ternarian hen (known for their succulent white meat). You place the hen on the left side. Place the 3-pound weight on the right side. This is too heavy, so you place the 1-pound weight with the hen on the left side to achieve balance. Note that the sum of the weights on the right side minus the sum of the weights on the left side equals the weight of the object.

As another example, consider weighing a 21-pound Ternarian squash. Using the Ternarian system, you would place weights of 27 pounds and 3 pounds in the right pan and a weight of 9 pounds in the left pan (along with the object) again achieving balance.

Write a program that accepts as input the weight of an object in base 10 and outputs the weights to be placed in both pans.

Input

The first line contains an integer $1 \leq n \leq 100$, indicating how many test cases are to be solved. On each of the next n lines there is an integer $0 \leq x \leq 10^9$ giving the weight of the object placed on the left scale.

Output

For each test case the program should produce two lines of output. The first line should contain `left pan:` followed by a space, followed by the weights to be placed in the left pan in descending order. The second line should contain `right pan:` followed by a space, followed by the weights to be placed in the right pan in descending order. Print a blank line **between** each pair of test cases.

Sample Input	Sample Output
4 2 3 21 250	left pan: 1 right pan: 3 left pan: right pan: 3 left pan: 9 right pan: 27 3 left pan: 3 right pan: 243 9 1

ANZAC 2014 - Round 1

Problem N: Erratic Ants

The ants of a particular colony are in search of food. Unfortunately hidden dangers are all around the colony which makes foraging difficult. There are traps, obstacles, and predators lurking about. Fortunately, the colony has the perfect ant for the job. Max is neither a smart ant nor an efficient ant but he has got blind luck on his side. In all of his wanderings, he has always managed to stay on safe ground and he (eventually) always finds a source of food to report back to the colony.

The problem is that Max rarely takes anything resembling an optimal (shortest) route to find a food source. However, Max can reliably bring back the exact details of the (often winding and convoluted) path that he took to get to the food source. Your job is to help the colony by finding the optimal route located within Max's convoluted directions to allow the colony to forage more efficiently.

Input

The first integer in the input, $1 \leq n \leq 100$, denotes the number of paths to food that Max has reported back. This is followed by a blank line and then descriptions of each of the n paths. Each path description begins with an integer, s ($0 \leq s \leq 60$), which denotes the number of steps taken in the path. The next s lines contain directional steps expressed as upper-case characters `N`, `E`, `S`, and `W` corresponding to steps taken in the directions north, east, south, and west respectively. Each step moves Max one unit of distance. Max's paths always start at the colony and end at a food source. Between each pair of path descriptions is a blank line.

When searching for an optimal path, the only directional steps that may be taken are ones that have previously been taken by Max, or the same steps in reverse.

Output

For each given path, give the number of steps found to be in an optimal (shortest) path.

Sample Input	Sample Output
3 8 S E E E N W S S 4 S E N W 3 S E N	4 0 3