icpc north america sponsor

icpc global programming tools sponsor

Official Problem Set

DO NOT OPEN UNTIL CONTEST BEGINS

2017 ACM ICPC North Central North America Regional Contest

October 28, 2017

acm International Collegiate Programming Contest

icpc.foundation

This page is intentionally left blank.

Problems

A	Stoichiometry
B	Pokemon Go Go
C	Urban Design
D	Smooth Array
E	Is-A? Has-A? Who Knowz-A?
F	Atlantis
G	Sheba's Amoebas
H	Zebras and Ocelots
I	Racing Around the Alphabet
J	Lost Map

This page is intentionally left blank.

Problem A
Stoichiometry

You have landed a lucrative contract with Amalgamated Chemical Manufacturing (ACM), to help their chemists with stoichiometry. Stoichiometry is the calculation of reactants and products in chemical reactions, based on the law of conservation of mass, which states that the total mass of the reactants equals the total mass of the products. The relations among quantities of reactants and products typically form a ratio of positive integers. If the amounts of the separate reactants are known, then the amount of the product can be calculated, and vice-versa. The relationship of reactants to products can be described using a soichiometric equation such as:

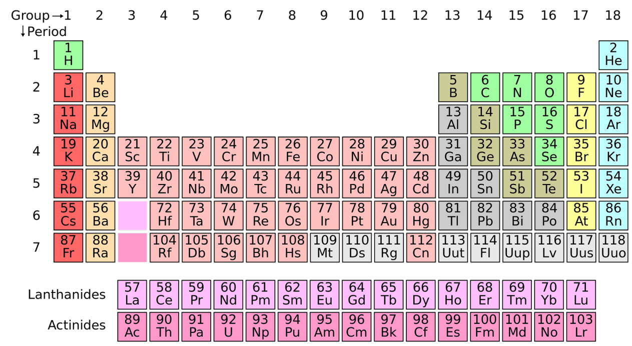


Image by Gerd Altmann.

$$\mathrm{CH_4 + 2O_2 \rightarrow CO_2 + 2H_2O}, \tag{1}$$

which can be read as: "One molecule of CH_4 and two molecules of O_2 yield one molecule of CO_2 and two molecules of H_2O." The total number of atoms of each element on the left hand side of the stoichiometric equation must match the number of atoms of that element on right hand side. Your task is to write a program that, given an equation of the form:

$$\mathrm{\_H_2O + \_CO_2 \rightarrow \_O_2 + \_C_6H_{12}O_6}, \tag{2}$$

will fill in the blanks to produce a balanced equation. For example, the above equation could be balanced as follows:

$$\mathrm{6H_2O + 6CO_2 \rightarrow 6O_2 + 1C_6H_{12}O_6}. \tag{3}$$

Input

An equation is input in the form of a sequence of M $(1 < M \leq 20)$ lines, one for each molecule in the formula (e.g., H_2O or CO_2). Each line m $(1 \leq m \leq M)$ has the following fields:

$$sign_m \quad N_m \quad element_{m,1} \quad count_{m,1} \quad \ldots \quad element_{m,N_m} \quad count_{m,N_m}$$

where $sign_m$ is either `+1` or `-1`, with a sign of `+1` indicating that this molecule appears on the left of the equation, and `-1` indicating that it appears on the right. N_m, where $0 < N_m < 20$, is the number of *element*/*count* pairs following on the line. Each $element_{m,n}$, where $1 \leq n \leq N_m$, is an element name consisting of one or two upper or lowercase letters, and each $count_{m,n}$ is a positive integer, $1 \leq count_{m,n} \leq 12$. For example, the element/count pair "`Fe 2`" indicates that the molecule contains two atoms of the element Fe (iron). There will be no more than 10 unique elements in a single equation.

Note that an element may be repeated in a given line of input, as in

```
+1 6 C 1 H 5 C 1 O 1 O 1 H 1
```

which specifies that at least one molecule of CH_5COOH appears on the left side of the equation. Note that CH_5COOH can be written as $C_2H_6O_2$.

Input ends with a line of the form

```
0 0
```

Output

The program must output the numbers $C_1, \ldots, C_M$ $(0 < C_i \leq 1\,000)$, in order, to fill in the blanks in the equation. Each number, $C_m \mid 1 \leq m \leq M$, must be the minimum number for which the equation is balanced (i.e. there is no common factor that could reduce all of the C_m coefficients). You may assume that every input test case has exactly one valid solution meeting these criteria.

Sample Input 1	Sample Output 1
+1 2 H 2 O 1 +1 2 C 1 O 2 -1 1 O 2 -1 3 C 6 H 12 O 6 0 0	6 6 6 1

Sample Input 2	Sample Output 2
+1 5 Be 2 C 1 O 3 O 2 H 2 +1 3 Ac 1 O 1 H 1 -1 4 Be 4 O 1 Ac 6 O 6 -1 2 H 2 O 1 -1 2 C 1 O 2 0 0	2 6 1 5 2

Problem B
Pokemon Go Go

Always Catch your Mon, Inc., (also know as ACM), wants to create a new product, called Pokémon Go Go. Users can purchase this application to help them play Pokémon go. The software accesses the poké stop locations near the current location as well as a list of Pokémon that can be found at each stop. The application then computes the shortest route one can follow to catch all the unique Pokémon, and return to the starting point.

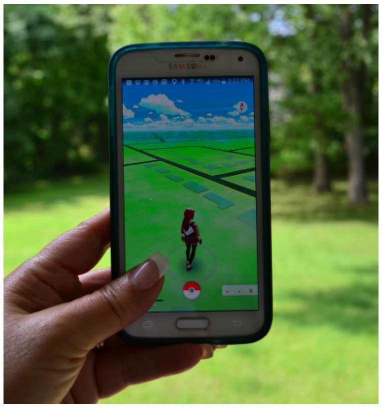


Image by Ana Verrusio.

The program assumes that the user is in a city where travel is restricted to moving only in the north–south and east–west directions. The program also assumes that all poké stops are on the intersection of two roads.

For example, consider a case where the application finds five nearby poké stops. Each stop's location is indicated by two integers, (r,c), where r is the number of blocks north of your starting position and c is the number of blocks west of your starting position. Consider if the locations of the five poké stops are $(5, 9)$, $(20, 20)$, $(1, 1)$, $(1, 8)$ and $(2, 8)$ while the names of the Pokémon found at these stops are Evevee, Flareon, Flareon, Jolteon, and Umbreon, respectively. It is clear that one does not have to visit both the second and third stops, since the same Pokémon can be caught at either of them. The best route is to visit the first, fifth, fourth, and third stops (in that order) for a total distance of 28 blocks, since:

- The distance from $(0, 0)$ to $(5, 9)$ is 14.
- The distance from $(5, 9)$ to $(2, 8)$ is 4.
- The distance from $(2, 8)$ to $(1, 8)$ is 1.
- The distance from $(1, 8)$ to $(1, 1)$ is 7.
- The distance from $(1, 1)$ to $(0, 0)$ is 2.

Input

The input holds a single test case. The test case begins with a single integer n, $0 < n \leq 20$, which indicates the number of poké stops to consider. Each of the next n lines specifies the location of a poké stop and the name of a Pokémon that can be found there. The location is specified by two integers r and c separated by a single space, $-100 \leq r, c \leq 100$. The integers r and c indicate that the stop is r blocks north and c blocks east of the starting point. The location is followed by a single space and followed by the string p indicating the name of the Pokémon that can be caught there. Names have between 1 and 25 letters (using only a–z and A–Z). The number of unique Pokémon is always less than or equal to 15. Multiple pokémon can reside at a single poké stop and are listed on separate lines.

Output

Give the shortest distance, in blocks, required to catch all the unique Pokémon.

Sample Input 1	Sample Output 1
5 5 9 Eevee 20 20 Flareon 1 1 Flareon 1 8 Jolteon 2 8 Umbreon	28

Problem C
Urban Design

Image by Dietmar Silber.

A new town is being planned, and the designers have some very specific ideas about how things should be laid out. First, they lay out the streets. Each street is perfectly straight and passes completely from one end of the town to the other. These streets divide the town into regions, and each region is to be designated either "residential" or "commercial." The town planners require that any two regions directly across the street from one another must have different designations. On this one particular day, all of the streets have been planned, but none of the regions have been designated. One town planner wishes to purchase two properties, and it is important to him that the properties eventually have different designations. For this problem, the streets can be modeled by lines in the plane that extend forever in both directions and have no width, and properties may be modeled by points. Given the lines and two points, can you decide whether or not they must get different designations, "commercial" or "residential?"

Input

Input begins with an integer S on a single line, giving the number of streets ($1 \le S \le 10\,000$). The next S lines of input each contain four integers x_1, y_1, x_2, and y_2, specifying the coordinates of two distinct points (x_1, y_1) and (x_2, y_2). The unique line through these two points gives one of the streets. Each coordinate is in the range $[0, 10\,000]$, and no two lines will be identical. That is, the town will have S distinct streets. The next line contains an integer T, the number of pairs of properties to test ($1 \le T \le 1\,000$). This is followed by T lines of input, each containing four integers x_3, y_3, x_4, and y_4, representing two distinct points (x_3, y_3) and (x_4, y_4), where each point lies within one of the two properties to test. None of these points will lie on any of the streets, nor will both points lie within the same property. Again, each coordinate is in the range $[0, 10\,000]$.

Output

For each of the T pairs of properties to be tested, output either "same" if the properties are guaranteed to receive the same designation or "different" if they are guaranteed to receive different designations.

Sample Input 1

```
2
1 1 2 1
1 1 1 2
3
2 0 2 2
2 0 0 3
0 0 2 2
```

Sample Output 1

```
different
same
same
```

Sample Input 2	Sample Output 2
4 1 3 2 4 1 3 2 5 1 3 3 4 7 9 8 8 2 14 7 10 13 1 4 2 3	same different

Problem D
Smooth Array

We always hope things in our lives will run smoothly, and having smooth arrays may help. An array A of N non-negative integers is K_S-smooth if the sum of every set of K consecutive integers is exactly S. Unfortunately, not all arrays are K_S-smooth. In fact, all K_S-smooth arrays must contain a repeating pattern of length K. The image to the right shows an array of smoothies, and is totally unrelated to this problem, but may help you relax.

Image of an array of smoothies by Silvia.

Any array can be made K_S-smooth by changing its elements. In each change one element may be modified to any integer between 0 and S, inclusive. You want to make all of your arrays smooth, but you don't want to make any more changes than necessary. So the question is: What is the minimum number of changes you have to make so that a given array would become K_S-smooth?

Input

The first line of input will consist of three integers of the form:

$$N \quad K \quad S$$

where N is the size of the array, The remainder of the file will consist of N integers, $a_n \in A$, separated by white space. All input will be within the following constraints:

$$1 \leq K \leq N \leq 5000$$

$$\forall a_n \in A, \; 0 \leq a_n \leq S \leq 5000$$

Output

Your program must output a single integer specifying the minimum number of changes that must be made in order to make the array K_S smooth.

Sample Input 1	Sample Output 1
3 3 5 1 2 3	1

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

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

Problem E
Is-A? Has-A? Who Knowz-A?

Two familiar concepts in object oriented programming are the is-a and has-a relationships. Given two classes A and B, we say that A is-a B if A is a subclass of B; we say A has-a B if one of the fields of A is of type B. For example, we could imagine an object-oriented language (call it ICPC++) with code like that in Figure E.1, where the class `Day` is-a `Time`, the class `Appointment` is both a `DateBook` and a `Reminder`, and class `Appointment` has-a `Day`.

```
class Day extends Time
{
   ...
}
```

```
class Appointment extends Datebook, Reminder
{
   private Day date;
   ...
}
```

Figure E.1: Two ICPC++ classes.

These two relationships are transitive. For example if A is-a B and B is-a C then it follows that A is-a C. This holds as well if we change all the is-a's in the last sentence to has-a's. It also works with combinations of is-a's and has-a's: in the example above, `Appointment` has-a `Time`, since it has-a `Day` and `Day` is-a `Time`. Similarly, if class `DateBook` has-a `Year` then `Appointment` has-a `Year`, since `Appointment` is-a `DateBook`.

In this problem you will be given a set of is-a and has-a relationships and a set of queries of the form A is/has-a B. You must determine if each query is true or false.

Input

Input starts with two integers n and m, ($1 \leq n, m \leq 10,000$), where n specifies the number of given is-a and has-a relationships and m specifies the number of queries. The next n lines each contain one given relationship in the form c_1 r c_2 where c_1 and c_2 are single-word class names, and r is either the string "is-a" or "has-a". Following this are m queries, one per line, using the same format. There will be at most 500 distinct class names in the $n + m$ lines, and all class names in the last m lines will appear at least once in the initial n lines. All is-a and has-a relationships between the given classes can be deduced from the n given relationships. Is-a relationships can not be circular (apart from the trivial identity "x is-a x").

Output

For each query, display the query number (starting at one) and whether the query is true or false.

Sample Input 1	Sample Output 1

`5 5`
`Day is-a Time`
`Appointment is-a Datebook`
`Appointment is-a Reminder`
`Appointment has-a Day`
`Datebook has-a Year`
`Day is-a Time`
`Time is-a Day`
`Appointment has-a Time`
`Appointment has-a Year`
`Day is-a Day`
	`Query 1: true`
	`Query 2: false`
	`Query 3: true`
	`Query 4: true`
	`Query 5: true`

Problem F
Atlantis


Image by Enrique Meseguer.

You may have heard of the lost city of Atlantis. As legend goes, Atlantis was a city of great wealth and power. Then the Gods became displeased with Atlantis and sank it into the ocean. What you may not have heard is the story of Demetrios, the only person in Atlantis with a ship at the time the city was pushed underwater.

Demetrios was an incredibly selfish man and thought of only one thing when he noticed the water level in the city rising: collecting the wealth of accessible gold in Atlantis and transporting it into his ship before the treasure was lost forever. Luckily for Demetrios, when the water level began rising, the many guards keeping safe the gold stores abandoned their posts to seek safety.

Demetrios knows the location of every gold store in Atlantis and how long it will take him to get to a particular store and back to his ship. He also knows the altitude at which each gold store resides, which determines when the store will be swallowed by the rising sea. The trouble is that he's not sure he'll have time to make it to every store before they become submerged. He now wonders the maximum number of stores he can visit prior to their respective submersion, if he picks his schedule optimally.

During the 2017 NCNA Regional, the following clarification was posted: "Important: The gold store must remain above water during the ENTIRE trip to and from the store."

Input

The first line of input will contain the integer n, $(1 \leq n \leq 200\,000)$, the number of gold stores in Atlantis. The next n lines will contain two integers on each line, t_i and h_i, $(1 \leq t_i, h_i \leq 10^9)$, the round-trip time in seconds it will take Demetrios to visit store i and return to his ship with the gold, and the feet above sea level of store i, respectively.

Output

Output the maximum number of gold stores Demetrios can visit such that each is visited prior to it becoming submerged. Assume sea level rises by one foot every second, starts at height 0, and begins rising immediately.

Sample Input 1	Sample Output 1
5 5 8 5 6 3 4 5 13 6 10	3

Sample Input 2	Sample Output 2
5 5 10 6 15 2 7 3 3 4 11	4

Problem G
Sheba's Amoebas

After a successful Kickstarter campaign, Sheba Arriba has raised enough money for her mail-order biology supply company. "Sheba's Amoebas" can ship Petri dishes already populated with a colony of those tiny one-celled organisms. However, Sheba needs to be able to verify the number of amoebas her company sends out. For each dish she has a black-and-white image that has been pre-processed to show each amoeba as a simple closed loop of black pixels. (A loop is a minimal set of black pixels in which each pixel is adjacent to exactly two other pixels in the set; adjacent means sharing an edge or corner of a pixel.) All black pixels in the image belong to some loop.

Sheba would like you to write a program that counts the closed loops in a rectangular array of black and white pixels. No two closed loops in the image touch or overlap. One particularly nasty species of cannibalistic amoeba is known to surround and engulf its neighbors; consequently there may be amoebas within amoebas. For instance, each of the images in Figure G.1 contains four amoebas.

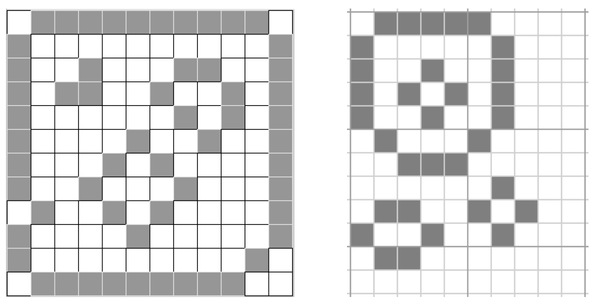


Figure G.1: Two Petri dishes, each with four amoebas.

Input

The first line of input contains two integers m and n, $(1 \leq m, n \leq 100)$. This is followed by m lines, each containing n characters. A '#' denotes a black pixel, a '.' denotes a white pixel. For every black pixel, exactly two of its eight neighbors are also black.

Output

Display a single integer representing the number of loops in the input.

Sample Input 1

```
12 12
.##########.
#..........#
#..#...##..#
#.##..#..#.#
#......#.#.#
#....#..#..#
#...#.#....#
#..#...#...#
.#..#.#....#
#....#.....#
#.........#.
.#########..
```

Sample Output 1

```
4
```

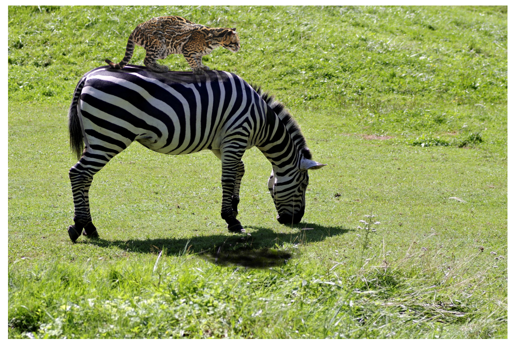
Sample Input 2

```
12 10
.#####....
#.....#...
#..#..#...
#.#.#.#...
#..#..#...
.#...#....
..###.....
......#...
.##..#.#..
#..#..#...
.##.......
..........
```

Sample Output 2

```
4
```

Problem H
Zebras and Ocelots



Ocelot by Jitze Couperus. Zebra by Henryk Niestrój. Composed by Larry Pyeatt.

The famous City Central Zoo houses just two creatures, Zebras and Ocelots. These magical creatures spend their time in a single, tall column, each standing atop the back of the creature below it. Whenever the zoo bell rings, the ocelot lowest in the pile turns into a zebra, and all zebras (if there are any) below that creature simultaneously turn into ocelots. If there is no ocelot in the pile when the bell tolls, then nothing happens. The zookeeper has been watching this interesting process for years. Today, suddenly, he begins to wonder how much longer this can go on. That is, given a pile of zebras and/or ocelots, how many times may the bell toll before there are no more ocelots?

Input

Input consists of a number N in the range 1 to 60, followed by N lines, each of which is a single character, either `Z` (for zebra) or `O` (for ocelot). These give the order of the creatures from top (first) to bottom (last).

Output

Output should be a single integer giving the number of times the bell must toll in order for there to be no more ocelots.

Sample Input 1	Sample Output 1
3 Z O Z	2

Sample Input 2	Sample Output 2
4 O Z Z O	9

This page is intentionally left blank.

Problem I
Racing Around the Alphabet

Soccer coach Joe has invented a new fitness game that he thinks will help his players' agility and fitness. The center circle of the soccer pitch has a diameter of 60 feet. He marks 28 evenly-spaced points on the circumference of the circle. On the first point (arbitrarily chosen) he places a pile of tokens all inscribed with the letter 'A'. On the second point he places a pile of 'B' tokens, and so on, covering 26 of the 28 points. He places a pile of blanks (or space-characters) on the 27^{th} point and a pile of disks imprinted with single quotation marks, on the last point.

He gives each player a slip of paper containing an aphorism, such as "WINNING ISN'T EVERYTHING IT'S THE ONLY THING" Notice that the only punctuation marks allowed are spaces and single quotation marks. Each player gets a different aphorism.

A player starts outside the circle next to the first letter of his aphorism and, on Joe's "GO" signal, the player picks up his first disk and then takes off running around the outside of the circle, picking up the remaining disks in the order that the corresponding letters appear in the aphorism. A smart player will take the shortest route, possibly changing direction, between consecutive letters of his aphorism.

One of the assistant coaches makes a note of the player's time. Although Joe initially envisioned all of his 20 players running around the circle simultaneously, this idea was scrapped due to contusions and fractures.

Joe wants you to write a program to estimate the expected time for **one** smart player to complete each of the aphorisms. Assume for simplicity that the player runs at 15 feet per second between stops for pickups and that each pickup takes 1 second.

Input

The input begins with a number N, $(1 \leq N \leq 20)$ giving the number of aphorisms to follow. N lines follow, each one containing an aphorism made up of upper-case letters, single quotation marks, and spaces. Each aphorism will be between 8 and 120 characters inclusive.

Output

For each aphorism give the time for a smart player to complete the task. Your answer should be accurate to within 10^{-6} seconds.

Sample Input 1	Sample Output 1
3	187.6156641641
WINNING ISN'T EVERYTHING IT'S THE ONLY THING	190.4108599662
WINNERS DON'T WAIT FOR CHANCES THEY TAKE THEM	193.1036536692
PAIN IS ONLY TEMPORARY BUT VICTORY IS FOREVER	

This page is intentionally left blank.

Problem J
Lost Map


Image by Pixabay Member 12019.

Somewhere in a mountainous region of the world is a collection of n villages. Connecting these villages to one another is a series of roads, always running directly between two villages and allowing travel in both directions. Due to the treacherous landscape, building these roads is expensive, so the minimum number of roads have been constructed such that each village can reach every other village via a sequence of roads.

Trade between these villages is very important, since not every village has access to the same supply of natural resources. Many villages produce the same resource, however, so it is useful for villages to know their relative distance to other villages so that they can choose trading partners to minimize overall trading costs. Note that the distance between two villages a and b is the sum of the lengths of the individual roads on the shortest path that connects a and b.

A project has been underway to compute the distance between every pair of villages. This information has been incorporated in a table, along with a map that shows the layout of villages and roads that run between them. You have been assigned the task of distributing the table and map to all the villages for the betterment of the regional economy.

Unfortunately, not long after you were given this task, a gust of wind blew the map right out of your hand and into the countryside. Despite your best efforts of searching for it, you have been unable to locate it. You know that you could visit all the villages to reconstruct the map and THEN start distributing the map and table, but this will take twice as long as the original task and the villages will be very displeased with you. You wonder, maybe it's possible to reconstruct the map given only the table?

Input

The first line of input will contain the integer n ($2 \leq n \leq 2\,500$), the number of villages in this region. The next n lines will contain n integers each. The j^{th} integer of the i^{th} line is the distance from village i to village j. All distances are greater than zero unless $i = j$, less than 10^7, and it is guaranteed that the distance from village i to village j is the same as the distance from village j to village i.

Output

For each test case, output $n - 1$ lines with two integers u and v on each line, indicating that there is a road connecting villages u and v in this region. Assume the villages are numbered from 1 to n. Any solution that outputs the original set of roads will be accepted.

Sample Input 1

```
4
0 1 1 2
1 0 2 3
1 2 0 3
2 3 3 0
```

Sample Output 1

```
1 2
1 3
1 4
```

Sample Input 2	Sample Output 2
7 0 2 9 1 4 3 3 2 0 11 1 6 3 3 9 11 0 10 5 12 12 1 1 10 0 5 2 2 4 6 5 5 0 7 7 3 3 12 2 7 0 4 3 3 12 2 7 4 0	1 4 1 5 2 4 3 5 4 6 4 7