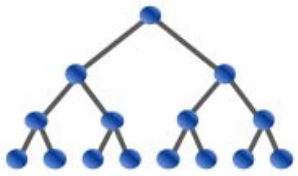


USACO 2017 US OPEN CONTEST, SILVER
PROBLEM 1. PAIRED UP

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Farmer John finds that his cows are each easier to milk when they have another cow nearby for moral support. He therefore wants to take his M cows ($M \leq 1,000,000,000$, M even) and partition them into $M/2$ pairs. Each pair of cows will then be ushered off to a separate stall in the barn for milking. The milking in each of these $M/2$ stalls will take place simultaneously.

To make matters a bit complicated, each of Farmer John's cows has a different milk output. If cows of milk outputs A and B are paired up, then it takes a total of $A+B$ units of time to milk them both.

Please help Farmer John determine the minimum possible amount of time the entire milking process will take to complete, assuming he pairs the cows up in the best possible way.

INPUT FORMAT (file pairup.in):

The first line of input contains N ($1 \leq N \leq 100,000$). Each of the next N lines contains two integers x and y, indicating that FJ has x cows each with milk output y ($1 \leq y \leq 1,000,000,000$). The sum of the x's is M, the total number of cows.

OUTPUT FORMAT (file pairup.out):

Print out the minimum amount of time it takes FJ's cows to be milked, assuming they are optimally paired up.

SAMPLE INPUT:

```
3
1 8
2 5
1 2
```

SAMPLE OUTPUT:

```
10
```

Here, if the cows with outputs 8+2 are paired up, and those with outputs 5+5 are paired up, the both stalls take 10 units of time for milking. Since milking takes place simultaneously, the entire process would therefore complete after 10 units of time. Any other pairing would be sub-optimal, resulting in a stall taking more than 10 units of time to milk.

Problem credits: Brian Dean

Contest has ended. No further submissions allowed.

Tree Sales

Filename: treesales

The Problem

Several well-known companies use a pyramid sales scheme. Being both an entrepreneur AND a computer scientist, however, you prefer to model your new business as a tree sales scheme, where the hierarchical structure of the company can be modeled as a tree.

In particular, the company initiator, or CEO, is the root of the tree structure of the company. From there, any current member of the company can hire a direct subordinate. So, at the beginning, it's up to the CEO to hire other employees who will be directly below the CEO in the tree structure. At any point in time, any employee can make a sale. Total compensation of any employee is calculated based on the sum of sales of all members of the subtree rooted at that employee, so it's important at any point in time to be able to calculate the total sales in any subtree of the company structure.

In order to start your company and allow others to start similarly structured companies, you have decided to write a computer program that will read in a set of operations from the following set:

1) Add an employee (first add is the CEO, rest are made by current employees)
2) An employee makes a sale
3) Query for the total sales in an employee's subtree at that point in time

and execute the appropriate command, in the order given, producing output for all commands of type three.

The Input

The first line of input will contain a single integer, T $(T \leq 10)$, representing the number of company structures to analyze. The first line of each company structure to analyze will contain a single positive integer, n $(n \leq 100000)$, representing the number of operations to execute for that company. The following n lines will each contain a single command with one of the following three formats:

```
ADD SPONSOR NEWEMPLOYEE
SALE EMPLOYEE X
QUERY EMPLOYEE
```

All employee names will be strings of 1 to 10 uppercase letters. In the first format, SPONSOR will be the current employee who is hiring a new employee, and the NEWEMPLOYEE will be the new employee to be added directly below the sponsor. The very first command for each company will be an add command with the sponsor "ROOT", indicating that NEWEMPLOYEE is the root of the tree structure for that company. No employee of any company will be named "ROOT". In the second format, EMPLOYEE will be the employee in question and X will be a positive integer less than 1000 representing the value of the sale made by the given employee. In the third format, EMPLOYEE will be the employee in question for which we must find the total

sales of her subtree in the company. All names given for current employees for all three types of commands are guaranteed to be valid current employees.

It is guaranteed that all employees added will be identified by distinct strings and that the tree structure produced will not have a height greater than 100. (Note: The height of a tree with two nodes is 1.)

The Output

For each company output a single line header of the form

```
COMPANY K
```

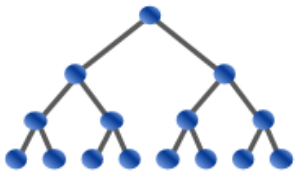
where K is the number of the company, starting with 1. For each query (command of type 3 in the input), output a single line with a positive integer representing the current total sales of the subtree of the employee queried. Note: Each company will have at least one query.

Sample Input

```
2
14
ADD ROOT BILL
ADD BILL EVELYN
ADD BILL SARAH
SALE BILL 25
SALE EVELYN 75
SALE SARAH 10
QUERY BILL
ADD EVELYN MATT
ADD MATT ANYA
SALE ANYA 1000
QUERY MATT
QUERY EVELYN
QUERY BILL
QUERY SARAH
11
ADD ROOT MARILYN
ADD MARILYN GARY
ADD MARILYN REMY
ADD MARILYN BRIANNE
ADD MARILYN TAJ
SALE TAJ 10
SALE REMY 20
SALE BRIANNE 40
SALE MARILYN 30
QUERY GARY
QUERY MARILYN
```

Sample Output

```
COMPANY 1
110
1000
1075
1110
10
COMPANY 2
0
100
```



USACO 2017 JANUARY CONTEST, SILVER
PROBLEM 1. COW DANCE SHOW

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

After several months of rehearsal, the cows are just about ready to put on their annual dance performance; this year they are performing the famous bovine ballet "Cowpelia".

The only aspect of the show that remains to be determined is the size of the stage. A stage of size K can support K cows dancing simultaneously. The N cows in the herd ($1 \leq N \leq 10,000$) are conveniently numbered $1 \ldots N$ in the order in which they must appear in the dance. Each cow i plans to dance for a specific duration of time $d(i)$. Initially, cows $1 \ldots K$ appear on stage and start dancing. When the first of these cows completes her part, she leaves the stage and cow $K+1$ immediately starts dancing, and so on, so there are always K cows dancing (until the end of the show, when we start to run out of cows). The show ends when the last cow completes her dancing part, at time T.

Clearly, the larger the value of K, the smaller the value of T. Since the show cannot last too long, you are given as input an upper bound T_{max} specifying the largest possible value of T. Subject to this constraint, please determine the smallest possible value of K.

INPUT FORMAT (file cowdance.in):

The first line of input contains N and T_{max}, where T_{max} is an integer of value at most 1 million.

The next N lines give the durations $d(1) \ldots d(N)$ of the dancing parts for cows $1 \ldots N$. Each $d(i)$ value is an integer in the range $1 \ldots 100,000$.

It is guaranteed that if $K = N$, the show will finish in time.

OUTPUT FORMAT (file cowdance.out):

Print out the smallest possible value of K such that the dance performance will take no more than T_{max} units of time.

SAMPLE INPUT:

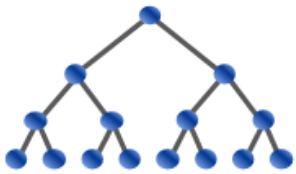
```
5 8
4
7
8
6
4
```

SAMPLE OUTPUT:

```
4
```

Problem credits: Delphine and Brian Dean

Contest has ended. No further submissions allowed.



USACO 2016 JANUARY CONTEST, GOLD
PROBLEM 1. ANGRY COWS

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Bessie the cow has designed what she thinks will be the next big hit video game: "Angry Cows". The premise, which she believes is completely original, is that the player shoots a cow with a slingshot into a one-dimensional scene consisting of a set of hay bales located at various points on a number line; the cow lands with sufficient force to detonate the hay bales in close proximity to her landing site, which in turn might set of a chain reaction that causes additional hay bales to explode. The goal is to use a single cow to start a chain reaction that detonates all the hay bales.

There are N hay bales located at distinct integer positions $x_1, x_2, \ldots, x_N$ on the number line. If a cow is launched with power R landing at position x, this will causes a blast of "radius R", engulfing all hay bales within the range $x - R \ldots x + R$. These hay bales then themselves explode (all simultaneously), each with a blast radius of $R - 1$. Any not-yet-exploded bales caught in these blasts then all explode (all simultaneously) with blast radius $R - 2$, and so on.

Please determine the minimum amount of power R with which a single cow may be launched so that, if it lands at an appropriate location, it will cause subsequent detonation of every single hay bale in the scene.

INPUT FORMAT (file angry.in):

The first line of input contains N ($2 \leq N \leq 50,000$). The remaining N lines all contain integers $x_1 \ldots x_N$ (each in the range $0 \ldots 1,000,000,000$).

OUTPUT FORMAT (file angry.out):

Please output the minimum power R with which a cow must be launched in order to detonate all the hay bales. Answers should be rounded and printed to exactly 1 decimal point.

SAMPLE INPUT:

```
5
8
10
3
11
1
```

SAMPLE OUTPUT:

```
3.0
```

In this example, a cow launched with power 3 at, say, location 5, will cause immediate detonation of hay bales at positions 3 and 8. These then explode (simultaneously) each with blast radius 2, engulfing bales at positions 1 and 10, which next explode (simultaneously) with blast radius 1, engulfing the final bale at position 11, which finally explodes with blast radius 0.

Problem credits: Brian Dean

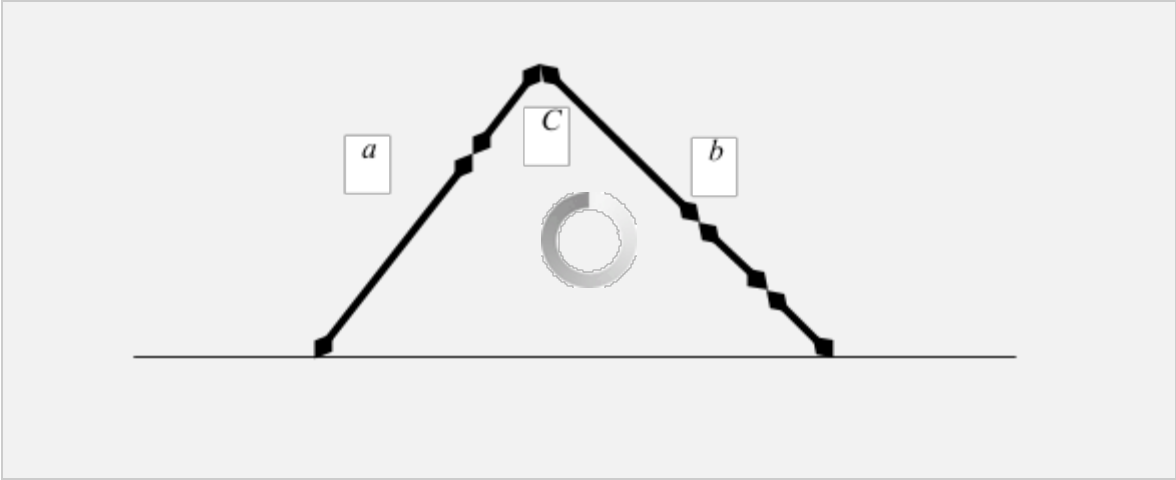
Contest has ended. No further submissions allowed.

Queen Dido’s New Challenge (prob8)

The Problem

Queen Dido of Carthage (850 BC) solved the first isoperimetric problem. She was promised as much land along the coast as she could enclose with an ox hide. According to the legend, she cut up a bull's hide into very thin strips which she sewed together into one long string. Then she took the seashore as one edge for the piece of land and laid the skin into a semi-circle. Modern calculus of variations would prove that Queen Dido’s solution is optimal and with a fixed string length the semi-circle encloses maximum area along a straight coastline.

Now Queen Dido was facing a new challenge. Given a bunch of twigs of various lengths, she needed to form a triangle along the coast to enclose the largest possible area.



One side of the triangle was the seashore and would not use any twigs. The other two sides should be formed by laying the twigs. The twigs could not be cut down further.

From the area formula Area = (½)ab sin C, it is easy to see that the optimal triangle must be a right triangle. An isosceles triangle (a=b) is clearly optimal, but it may not be possible to form because the twigs are discrete. However, triangles closer to the isosceles triangle will have larger areas, as seen from the formula Area = (½)ab = (½)(m^2 - $(a-m)^2$), where m = (a + b)/2 is a constant.

Input

Each input line consists of a positive integer n ($2 \le n \le 100$), the number of twigs, followed by n positive integers representing the lengths of the twigs:

n L1 L2 ... Ln

The sum of the twig lengths is odd and $L1 + L2 + \cdots + Ln < 100000$.

Output

For each input line, print the area of the largest triangle, rounded to the nearest integer.

Sample Input

```
3 4 2 5
4 1 3 7 10
4 3 3 8 11
```

Sample Output

```
15
55
77
```

Counting Sequences

Filename: countseq

Given a sequence $c_0, c_1, \ldots c_{n-1}$ of letters, we define a subsequence index list to be some ordered list of integers $a_0, a_1, \ldots a_{m-1}$. The corresponding subsequence is the string $c[a_0]c[a_1]\ldots c[a_{m-1}]$. For example, given the string, s = "engineering", the subsequence defined by the index list 1, 3, 4 and 6 is "nine", since s[1] = 'n', s[3] = 'i', s[4] = 'n', and s[6] = 'e'. The number of times a subsequence occurs in a given sequence is simply the number of unique index lists that generate the given subsequence. For the problem at hand, there are two occurrences of the subsequence "nine". One is the given one and the other is generated by the index list 1, 3, 4 and 5.

The Problem

Given a sequence of letters as well as subsequence from the original list, you are to count how many times the particular subsequence occurs.

The Input

The first line of the input will contain a single positive integer, *n*, representing the number of test cases in the file. The following *n* lines will contain 1 test case each. Each of these lines will have two non-empty strings of lowercase letters separated by a space. The first of these strings is the given input sequence while the second string will be the subsequence for which to search. It is guaranteed that the length of the first string is 1000 or less and that the length of the subsequence is less than or equal to the length of the given sequence. Furthermore, it's guaranteed that the subsequence appears at least once in the sequence.

The Output

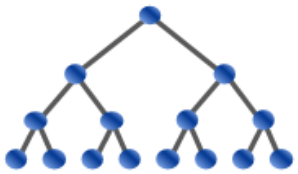
For each test case, output a single line with the number of occurrences of the given subsequence. The cases will be such that this value is less than 10^{18}.

Sample Input

```
2
engineering
nine
sallysellsseashells
sell
```

Sample Output

```
2
21
```



USACO 2016 JANUARY CONTEST, GOLD
PROBLEM 2. RADIO CONTACT

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Farmer John has lost his favorite cow bell, and Bessie the cow has agreed to help him find it! They both fan out and search the farm along different paths, but stay in contact via radio so they can keep in touch with each-other. Unfortunately, the batteries in their radios are running low, so they want to plan their movements so as to conserve power, by trying to stay always within a short distance apart.

Farmer John starts at location (f_x, f_y) and plans to follow a path consisting of N steps, each of which is either 'N' (north), 'E' (east), 'S' (south), or 'W' west. Bessie starts at location (b_x, b_y) and follows a similar path consisting of M steps. Both paths may share points in common. At each time step, Farmer John can either stay put at his current location, or take one step forward along his path, in whichever direction happens to be next (assuming he has not yet reached the final location in his path). Bessie can make a similar choice. At each time step (excluding the first step where they start at their initial locations), their radios consume energy equal to the square of the distance between them.

Please help FJ and Bessie plan a joint movement strategy that will minimize the total amount of energy consumed up to and including the final step where both of them first reach the final locations on their respective paths.

INPUT FORMAT (file radio.in):

The first line of input contains N and M ($1 \leq N, M \leq 1000$). The second line contains integers f_x and f_y, and the third line contains b_x and b_y ($0 \leq f_x, f_y, b_x, b_y \leq 1000$). The next line contains a string of length N describing FJ's path, and the final line contains a string of length M describing Bessie's path.

It is guranteed that Farmer John and Bessie's coordinates are always in the range ($0 \leq x, y \leq 1000$) throughout their journey. Note that East points in the positive x direction and North points in the positive y direction.

OUTPUT FORMAT (file radio.out):

Output a single integer specifying the minimum energy FJ and Bessie can use during their travels.

SAMPLE INPUT:

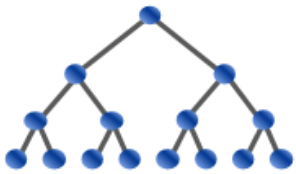
```
2 7
3 0
5 0
NN
NWWWWWN
```

SAMPLE OUTPUT:

```
28
```

Problem credits: Brian Dean

Contest has ended. No further submissions allowed.



USACO 2018 FEBRUARY CONTEST, GOLD
PROBLEM 3. TAMING THE HERD

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Early in the morning, Farmer John woke up to the sound of splintering wood. It was the cows, and they were breaking out of the barn again!

Farmer John was sick and tired of the cows' morning breakouts, and he decided enough was enough: it was time to get tough. He nailed to the barn wall a counter tracking the number of days since the last breakout. So if a breakout occurred in the morning, the counter would be 0 that day; if the most recent breakout was 3 days ago, the counter would read 3. Farmer John meticulously logged the counter every day.

The end of the year has come, and Farmer John is ready to do some accounting. The cows will pay, he says! But something about his log doesn't look quite right...

Farmer John wants to find out how many breakouts have occurred since he started his log. However, he suspects that the cows have tampered with his log, and all he knows for sure is that he started his log on the day of a breakout. Please help him determine, for each number of breakouts that might have occurred since he started the log, the minimum number of log entries that must have been tampered with.

INPUT FORMAT (file taming.in):

The first line contains a single integer N ($1 \leq N \leq 100$), denoting the number of days since Farmer John started logging the cow breakout counter.

The second line contains N space-separated integers. The ith integer is a non-negative integer a_i (at most 100), indicating that on day i the counter was at a_i, unless the cows tampered with that day's log entry.

OUTPUT FORMAT (file taming.out):

The output should consist of N integers, one per line. The ith integer should be the minimum over all possible breakout sequences with i breakouts, of the number of log entries that are inconsistent with that sequence.

SAMPLE INPUT:

```
6
1 1 2 0 0 1
```

SAMPLE OUTPUT:

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

If there was only 1 breakout, then the correct log would look like 0 1 2 3 4 5, which is 4 entries different from the given log.

If there were 2 breakouts, then the correct log might look like 0 1 2 3 0 1, which is 2 entries different from the given log. In this case, the breakouts occurred on the first and fifth days.

If there were 3 breakouts, then the correct log might look like 0 1 2 0 0 1, which is just 1 entry different from the given log. In this case, the breakouts occurred on the first, fourth, and fifth days.

And so on.

Problem credits: Brian Dean and Dhruv Rohatgi

Contest has ended. No further submissions allowed.

Visiting Relatives

Filename: `relatives`

You must visit each of your relatives during the summer after you graduate college, because after that you start a job and have no idea when you'll have an extended amount of free time. Of course, since you don't have the job yet, you have little money and want to minimize your traveling costs.

The Problem:

Given the cost of traveling between each pair of locations, write a program to calculate the minimum cost to start at your home, travel to each relative exactly once, and return home. (Note: Thus, if you are at location 3 and want to go to location 5 next, and this cost is $45.99, but the cost of going from location 3 to 4 is $20.00 and the cost of going from 4 to 5 is $20,00, if you had previously visited location 4, you must still pay $45.99 for the direct route, since going via location 4 means visiting it twice, and you'd hate to show any one relative more love than the others, since they'd get jealous!)

The Input:

The first line of the input file will have a single positive integer, *T (T ≤ 20)*, representing the number of test cases in the file. The first line of each test case will contain a single positive integer*, n (n ≤ 15)*, representing the total number of locations you must visit, including your own home. The data for the case follows on the following *n* lines. The first of these *n* lines will contain the cost of traveling from your home (location 0) to all locations, numbered 0 through *n-1*, respectively. All of these costs will be positive real numbers expressed to exactly 2 decimal places, less than 1000. The following *n-1* lines will contain the corresponding traveling costs from locations 1 through *n-1*, respectively.

The Output:

For each test case, output the cost in dollars, rounded to two decimal places for the minimal traveling cost of starting from your home, visiting each relative exactly once and returning home.

Sample Input:

```
1
3
0.00 2.00 4.00
3.00 0.00 5.00
2.50 5.50 0.00
```

Sample Output:

```
9.50
```

J: You Win!

You just achieved the High Score on your favorite video game! Now, you get to enter your name! You have to use the controller to enter your name, which can be awkward. Here's how it works:

- There are only the 26 capital letters **A** to **Z**, in order. There are no numbers, spaces, lower case letters, or any other characters.
- Pushing UP or DOWN changes the active letter one letter forward (UP) or backward (DOWN). The active letter starts at **A**. It will not reset when you move around in the name. It also wraps: UP from **Z** goes to **A**, DOWN from **A** goes to **Z**.
- Pushing LEFT or RIGHT moves the cursor one letter left or right in the current name. Note that once the cursor is at either end of the current name, it cannot move any further in that direction.
- Pushing the FIRE button adds the active letter to the name.

For example, consider the name 'ALMA'. One way you could enter 'ALMA' is like this:

Action	# of Pushes	Name (| = Cursor)	Active Letter
FIRE	1	A|	A
UP	11	A|	L
FIRE	1	AL|	L
UP	1	AL|	M
FIRE	1	ALM|	M
DOWN	12	ALM|	A
FIRE	1	ALMA|	A

This would take 28 button pushes. However, consider entering 'ALMA' like this:

Action	# of Pushes	Name (| = Cursor)	Active Letter
FIRE	1	A|	A
FIRE	1	AA|	A
LEFT	1	A|A	A
UP	11	A|A	L
FIRE	1	AL|A	L
UP	1	AL|A	M
FIRE	1	ALM|A	M

This takes only 17 button pushes. Given a name, what is the fewest number of button pushes needed to enter that name? Assume that the active letter starts at **A**, and that it doesn't matter where the cursor ends up when you're done.

Input

There will be several test cases in the input. Each test case will consist of a single string on its own line, with from 1 to 18 capital letters, representing a name that must be entered into the High Score list. The input will end with a line with a single **0**.

Output

For each test case, output a single integer representing the smallest number of button pushes needed to enter the name. Output no spaces, and do not separate answers with blank lines.

Sample Input	Sample Output
ALMA YES 0	17 21

Problem B

Spreading News

You are the manager of a company, and you want all of your employees to be notified of an important news item as quickly as possible. Your company is organized in a tree-like structure: each employee has exactly one direct supervisor, no employee is his own direct or indirect supervisor, and every employee is your direct or indirect subordinate. You will make a phone call to each of your direct subordinates, one at a time. After hearing the news, each subordinate must notify each of his direct subordinates, one at a time. The process continues this way until everyone has heard the news. Each person may only call direct subordinates, and each phone call takes exactly one minute. Note that there may be multiple phone calls taking place simultaneously. Compute the minimum amount of time, in minutes, required for this process to be completed. Employees will be numbered starting from 1, while you will be numbered 0. Furthermore, every supervisor is numbered lower than his or her direct subordinates.s

Input

First line of the input contains T the number of test cases. Each test case contains an integer N ($1 \leq N \leq 70$) denoting the number of employees in your company including you. Next line contains N-1 integer. The i'th integer denote the supervisor of i'th employee(i starts from 1) . Look you(employee 0) do not have any supervisor.

Output

For each test case output the minimum amount of time, in minutes, required.

Sample Input	Sample output
5 3 0 0 5 0 0 2 2 9 0 0 1 1 2 2 3 4 5 0 1 2 3 7 0 1 2 3 3 3	2 3 4 4 6

USACO 2018 FEBRUARY CONTEST, GOLD
PROBLEM 2. DIRECTORY TRAVERSAL

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Bessie the cow is surprisingly computer savvy. On her computer in the barn, she stores all of her precious files in a collection of directories; for example:

```
bessie/
  folder1/
    file1
    folder2/
      file2
  folder3/
    file3
  file4
```

There is a single "top level" directory, called bessie.

Bessie can navigate to be inside any directory she wants. From a given directory, any file can be referenced by a "relative path". In a relative path, the symbol ".." refers to the parent directory. If Bessie were in folder2, she could refer to the four files as follows:

```
../file1
file2
../../folder3/file3
../../file4
```

Bessie would like to choose a directory from which the sum of the lengths of the relative paths to all the files is minimized.

INPUT FORMAT (file dirtraverse.in):

The first line contains an integer N ($2 \leq N \leq 100,000$), giving the total number of files and directories. For the purposes of input, each object (file or directory) is assigned a unique integer ID between 1 and N, where ID 1 refers to the top level directory.

Next, there will be N lines. Each line starts with the name of a file or directory. The name will have only lower case characters a-z and digits 0-9, and will be at most 16 characters long. Following the name is an integer, m. If m is 0, then this entity is a file. If $m > 0$, then this entity is a directory, and it has a total of m files or directories inside it. Following m there will be m integers giving the IDs of the entities in this directory.

OUTPUT FORMAT (file dirtraverse.out):

Output the minimal possible total length of all relative paths to files. Note that this value may be too large to fit into a 32-bit integer.

SAMPLE INPUT:

```
8
bessie 3 2 6 8
folder1 2 3 4
file1 0
folder2 1 5
file2 0
folder3 1 7
file3 0
file4 0
```

SAMPLE OUTPUT:

```
42
```

This input describes the example directory structure given above.

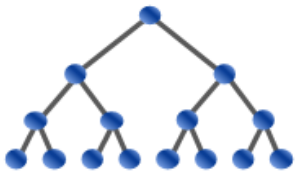
The best solution is to be in folder1. From this directory, the relative paths are:

```
file1
folder2/file2
../folder3/file3
../file4
```

Problem credits: Mark Gordon

Contest has ended. No further submissions allowed.

USA Computing Olympiad



USACO 2018 DECEMBER CONTEST, GOLD
PROBLEM 1. FINE DINING

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

The cows are heading back to the barn at the end of a long day, feeling both tired and hungry.

The farm consists of N pastures ($2 \leq N \leq 50,000$), conveniently numbered $1 \ldots N$. The cows all want to travel to the barn in pasture N. Each of the other $N-1$ pastures contains a cow. Cows can move from pasture to pasture via a set of M undirected trails ($1 \leq M \leq 100,000$). The ith trail connects a pair of pastures a_i and b_i, and requires time t_i to traverse. Every cow can reach the barn through a sequence of trails.

Being hungry, the cows are interested in potentially stopping for food on their way home. Conveniently, K of the pastures contain tasty haybales ($1 \leq K \leq N$), with the ith such haybale having a yumminess value of y_i. Each cow is willing to stop at a single haybale along her trip to the barn, but only if the amount of time this adds to her path is at most the yumminess of the haybale she visits. Note that a cow only "officially" visits at most one haybale for dining purposes, although it is fine if her path takes her through other pastures containing haybales; she simply ignores these.

INPUT FORMAT (file dining.in):

The first line contains three space-separated integers N, M, and K. Each of the next M lines contains three integers a_i, b_i, and t_i, describing a trail between pastures a_i and b_i which takes t_i time to traverse (a_i and b_i are different from each-other, and t_i is a positive integer at most 10^4)

The next K lines each describe a haybale in terms of two integers: the index of its pasture, and its yumminess value (a positive integer at most 10^9). Multiple haybales can reside in the same pasture.

OUTPUT FORMAT (file dining.out):

The output should consist of $N-1$ lines. Line i contains the single integer 1 if the cow at pasture i can visit and dine on a haybale on the way to the barn, and 0 otherwise.

SAMPLE INPUT:

```
4 5 1
1 4 10
2 1 20
4 2 3
2 3 5
4 3 2
2 7
```

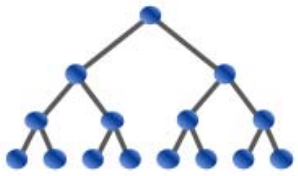
SAMPLE OUTPUT:

```
1
1
1
```

In this example, the cow in pasture 3 should stop for a meal, since her route would only increase by 6 (from 2 to 8), and this increase is at most the yumminess 7 of the haybale. The cow in pasture 2 should obviously eat the hay in pasture 2, since this causes no change in her optimal route. The cow in pasture 1 is an interesting case, as it may first appear that her optimal route (length 10) would increase too much to justify stopping for the hay. However, she actually does have a route that makes stopping at the hay beneficial: move to pasture 4, then to pasture 2 (eating the hay), then back to pasture 4.

Problem credits: Dhruv Rohatgi

Contest has ended. No further submissions allowed.



USACO 2016 FEBRUARY CONTEST, SILVER
PROBLEM 2. LOAD BALANCING

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Farmer John's N cows are each standing at distinct locations $(x_1, y_1) \ldots (x_n, y_n)$ on his two-dimensional farm ($1 \leq N \leq 1000$, and the x_i's and y_i's are positive odd integers of size at most $1,000,000$). FJ wants to partition his field by building a long (effectively infinite-length) north-south fence with equation $x = a$ (a will be an even integer, thus ensuring that he does not build the fence through the position of any cow). He also wants to build a long (effectively infinite-length) east-west fence with equation $y = b$, where b is an even integer. These two fences cross at the point (a, b), and together they partition his field into four regions.

FJ wants to choose a and b so that the cows appearing in the four resulting regions are reasonably "balanced", with no region containing too many cows. Letting M be the maximum number of cows appearing in one of the four regions, FJ wants to make M as small as possible. Please help him determine this smallest possible value for M.

INPUT FORMAT (file balancing.in):

The first line of the input contains a single integer, N. The next N lines each contain the location of a single cow, specifying its x and y coordinates.

OUTPUT FORMAT (file balancing.out):

You should output the smallest possible value of M that FJ can achieve by positioning his fences optimally.

SAMPLE INPUT:

```
7
7 3
5 5
7 13
3 1
11 7
5 3
9 1
```

SAMPLE OUTPUT:

```
2
```

Problem credits: Brian Dean

Contest has ended. No further submissions allowed.

Problem B: Polling

Filename: `polling`
Timelimit: `1 second`

Midterm elections are here! Help your local election commission by counting votes and telling them the winner. If more than one candidate ties with the most votes, print out all of their names in alphabetical order.

Input

Each input will consist of a single test case. Note that your program may be run multiple times on different inputs. Each test case will begin with an integer ***n*** (1<=***n***<=1,000), indicating the number of votes. The next ***n*** lines will hold the votes. The candidates' names will appear one per line, and consist of between 1 and 20 capital letters only.

Output

Output the name of the candidate with the most votes. If there is a tie, output out all of the names of candidates with the most votes, one per line, in alphabetical order. Do not output any spaces, and do not output blank lines between names.

Samples

Input	Output
5 FRED BARNEY FRED FRED BARNEY	FRED
5 PORTHOS ATHOS ARAMIS PORTHOS ATHOS	ATHOS PORTHOS

Sorting Student Presentations

Filename: `sorting`

Your teacher is tired of sorting the students in the class in regular alphabetical order, since this means that certain students always go first for presentations. She has an idea to mix up the order of the students by sorting them in a different manner. In particular, she will not care about the order of the letters in the last name of a student. Instead, she will put the student who has the highest number of A's in his or her last name first. If there is a tie between two students based on this value, she'll compare the number of B's in their last names. If this is tied also, she will go on to C's, then D's, etc. The second student will be found using the same method, and so on until all of the students have presented.

The Problem:

Your job is to sort a list of student names based on this criterion. It is guaranteed that no class has two students who have names that are anagrams of one another; thus, there will not be any ties between two students (an anagram is a different arrangement of the same exact letters).

The Input:

The first line of the input file consists of a single positive integer, n, representing the number of classes in the file. The first line of each class contains a single positive integer, m ($1 < m < 100$), which represents the number of students in that class. Each of the following m lines will contain the last name of one student in the class. It is guaranteed that each name will only contain uppercase letters and will be between 1 and 20 characters long, inclusive.

The Output:

For the k^{th} ($1 \leq k \leq n$) class, output a single header line with the following format:

`Class #k ordering`

Then, output each name in the order given by the new sorting scheme, with one name per line. Put one blank line at the end of the output for each class.

(Sample Input and Sample Output follow on next page)

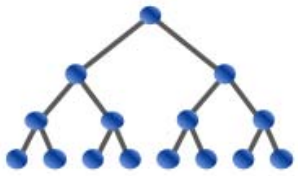
Sample Input:

```
2
2
JONES
LINDT
3
WALLACE
DAVIS
MADISON
```

Sample Output:

```
Class #1 ordering
LINDT
JONES

Class #2 ordering
WALLACE
MADISON
DAVIS
```



USACO 2016 US OPEN CONTEST, SILVER
PROBLEM 2. DIAMOND COLLECTOR

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Bessie the cow, always a fan of shiny objects, has taken up a hobby of mining diamonds in her spare time! She has collected N diamonds ($N \leq 50,000$) of varying sizes, and she wants to arrange some of them in a pair of display cases in the barn.

Since Bessie wants the diamonds in each of the two cases to be relatively similar in size, she decides that she will not include two diamonds in the same case if their sizes differ by more than K (two diamonds can be displayed together in the same case if their sizes differ by exactly K). Given K, please help Bessie determine the maximum number of diamonds she can display in both cases together.

INPUT FORMAT (file diamond.in):

The first line of the input file contains N and K ($0 \leq K \leq 1,000,000,000$). The next N lines each contain an integer giving the size of one of the diamonds. All sizes will be positive and will not exceed $1,000,000,000$.

OUTPUT FORMAT (file diamond.out):

Output a single positive integer, telling the maximum number of diamonds that Bessie can showcase in total in both the cases.

SAMPLE INPUT:

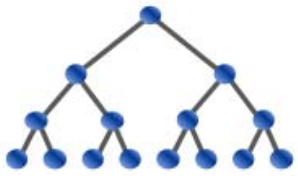
```
7 3
10
5
1
12
9
5
14
```

SAMPLE OUTPUT:

```
5
```

Problem credits: Nick Wu and Brian Dean

Contest has ended. No further submissions allowed.



USACO 2016 JANUARY CONTEST, SILVER
PROBLEM 2. SUBSEQUENCES SUMMING TO SEVENS

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Farmer John's N cows are standing in a row, as they have a tendency to do from time to time. Each cow is labeled with a distinct integer ID number so FJ can tell them apart. FJ would like to take a photo of a contiguous group of cows but, due to a traumatic childhood incident involving the numbers $1 \ldots 6$, he only wants to take a picture of a group of cows if their IDs add up to a multiple of 7.

Please help FJ determine the size of the largest group he can photograph.

INPUT FORMAT (file div7.in):

The first line of input contains N ($1 \leq N \leq 50,000$). The next N lines each contain the N integer IDs of the cows (all are in the range $0 \ldots 1,000,000$).

OUTPUT FORMAT (file div7.out):

Please output the number of cows in the largest consecutive group whose IDs sum to a multiple of 7. If no such group exists, output 0.

You may want to note that the sum of the IDs of a large group of cows might be too large to fit into a standard 32-bit integer. If you are summing up large groups of IDs, you may therefore want to use a larger integer data type, like a 64-bit "long long" in C/C++.

SAMPLE INPUT:

```
7
3
5
1
6
2
14
10
```

SAMPLE OUTPUT:

```
5
```

In this example, 5+1+6+2+14 = 28.

Problem credits: Brian Dean

Contest has ended. No further submissions allowed.

Problem D: Bodies of Water

Filename: `water`
Time limit: `1 second`

For those who don't like regular images, ASCII Maps Inc. has created maps that are fully printable ASCII characters. Each map is a rectangular grid of lowercase English letters, where each letter stands for various locations. In particular, 'w' stands for water and the other 25 letters represent various different land locations. For this problem, we are interested in counting the number of bodies of water on a given ASCII map. A body of water is a maximal set of contiguous grid squares on the ASCII map where each square in the body of water shares a boundary with at least one other square in the body of water. Thus, for two grid squares to be part of the same body of water, one must be above, below, to the left, or to the right of the other grid square.

Input

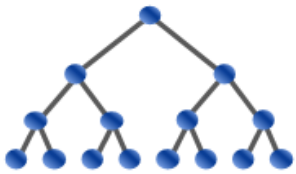
The first line of input consists of two space separated integers, r ($1 \le r \le 50$) and c ($1 \le c \le 50$), the number of rows and columns, respectively for the input map. The next r lines will each contain c lowercase English letters, representing the corresponding row of the input map.

Output

On a line by itself, output the number of bodies of water in the input map.

Samples

Input	Output
5 6 waaaww wawawc bbbbwc wwwwww dddddd	3
2 8 wxwxwxwx xwxwxwxw	8



USACO 2015 DECEMBER CONTEST, SILVER
PROBLEM 1. SWITCHING ON THE LIGHTS

Return to Problem List

Contest has ended.

Log in to allow submissions in analysis mode

English (en)

Farmer John has recently built an enormous barn consisting of an $N \times N$ grid of rooms ($2 \leq N \leq 100$), numbered from $(1,1)$ up to (N,N). Being somewhat afraid of the dark, Bessie the cow wants to turn on the lights in as many rooms as possible.

Bessie starts in room $(1,1)$, the only room that is initially lit. In some rooms, she will find light switches that she can use to toggle the lights in other rooms; for example there might be a switch in room $(1,1)$ that toggles the lights in room $(1,2)$. Bessie can only travel through lit rooms, and she can only move from a room (x,y) to its four adjacent neighbors $(x-1,y)$, $(x+1,y)$, $(x,y-1)$ and $(x,y+1)$ (or possibly fewer neighbors if this room is on the boundary of the grid).

Please determine the maximum number of rooms Bessie can illuminate.

INPUT FORMAT (file lightson.in):

The first line of input contains integers N and M ($1 \leq M \leq 20,000$).

The next M lines each describe a single light switch with four integers x, y, a, b, that a switch in room (x,y) can be used to toggle the lights in room (a,b). Multiple switches may exist in any room, and multiple switches may toggle the lights of any room.

OUTPUT FORMAT (file lightson.out):

A single line giving the maximum number of rooms Bessie can illuminate.

SAMPLE INPUT:

```
3 6
1 1 1 2
2 1 2 2
1 1 1 3
2 3 3 1
1 3 1 2
1 3 2 1
```

SAMPLE OUTPUT:

```
5
```

Here, Bessie can use the switch in $(1,1)$ to turn on lights in $(1,2)$ and $(1,3)$. She can then walk to $(1,3)$ and turn on the lights in $(2,1)$, from which she can turn on the lights in $(2,2)$. The switch in $(2,3)$ is inaccessible to her, being in an unlit room. She can therefore illuminate at most 5 rooms.

Problem credits: Austin Bannister and Brian Dean

Contest has ended. No further submissions allowed.

J: Underground Cables

A city wants to get rid of their unsightly power poles by moving their power cables underground. They have a list of points that all need to be connected, but they have some limitations. Their tunneling equipment can only move in straight lines between points. They only have room for one underground cable at any location except at the given points, so no two cables can cross.

Given a list of points, what is the least amount of cable necessary to make sure that every pair of points is connected, either directly, or indirectly through other points?

Input

There will be several test cases in the input. Each test case will begin with an integer **N** ($2 \le N \le 1{,}000$), which is the number of points in the city. On each of the next **N** lines will be two integers, **X** and **Y** ($-1{,}000 \le X,Y \le 1{,}000$), which are the (**X**,**Y**) locations of the **N** points. Within a test case, all points will be distinct. The input will end with a line with a single 0.

Output

For each test case, output a single real number, representing the least amount of cable the city will need to connect all of its points. Print this number with exactly two decimal places, rounded. Print each number on its own line with no spaces. Do not print any blank lines between answers.

Sample Input

```
4
0 0
0 10
10 0
10 10
2
0 0
10 10
0
```

Sample Output

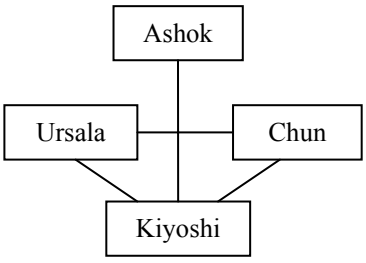
```
30.00
14.14
```

Problem I
Degrees of Separation
Input File: relatives.in

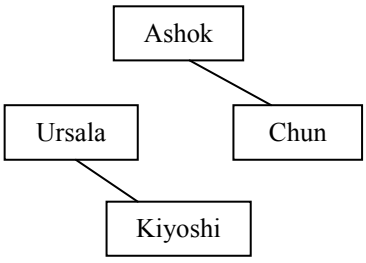
In our increasingly interconnected world, it has been speculated that everyone on Earth is related to everyone else by no more than six degrees of separation. In this problem, you must write a program to find the maximum degree of separation for a network of people.

For any two people, the degree of separation is the minimum number of relationships that must be traversed to connect the two people. For a network, the maximum degree of separation is the largest degree of separation between any two people in the network. If there is a pair of people in the network who are not connected by a chain of relationships, the network is disconnected.

As shown below, a network can be described as a set of symmetric relationships each of which connects two people. A line represents a relationship between two people. Network A illustrates a network with 2 as the maximum degree of separation. Network B is disconnected.



Network A:
Max. degree of separation = 2



Network B:
Disconnected

Input

The input consists of data sets that describe networks of people. For each data set, the first line has two integers: P ($2 \le P \le 50$), the number of people in the network, and R ($R \ge 1$), the number of network relationships. Following that first line are R relationships. Each relationship consists of two strings that are names of people in the network who are related. Names are unique and contain no blank spaces. Because a person may be related to more than one other person, a name may appear multiple times in a data set.

The final test case is followed by a line containing two zeroes.

Output

For each network, display the network number followed by the maximum degree of separation. If the network is disconnected, display `DISCONNECTED`. Display a blank line after the output for each network. Use the format illustrated in the sample output.

Sample Input	Output for the Sample Input
4 4 Ashok Kiyoshi Ursala Chun Ursala Kiyoshi Kiyoshi Chun 4 2 Ashok Chun Ursala Kiyoshi 6 5 Bubba Cooter Ashok Kiyoshi Ursala Chun Ursala Kiyoshi Kiyoshi Chun 0 0	Network 1: 2 Network 2: DISCONNECTED Network 3: DISCONNECTED

D: Dueling Philosophers

Following a sad and strange incident involving a room full of philosophers, several plates of spaghetti, and one too few forks, the faculty of the Department of Philosophy at ACM University have been going through the papers of a recently deceased colleague. The faculty members were amazed to find numerous unpublished essays. They believe that the essays, collected into one volume, may constitute a major work of scholarship that will give their department some much-needed positive publicity. Naturally, all of the faculty members began to vie for the honor (to say nothing of the fame) of serving as editor of the collection.

After much debate, the faculty members have narrowed the list to two candidates. Both applicants were asked to explain how they would arrange the essays within the final book. Both have noted that many of the essays define terminology and concepts that are explored in other essays. Both have agreed to the basic principle that an essay that *uses* a term must appear *after* the essay that *defines* that term. One of the candidates has presented what he claims is the only possible arrangement of the essays, under those constraints, and is arguing that he should be given the job simply because he has already done this major part of the work. The second candidate scoffs at this claim, insisting that there are many possible arrangements of the essays, and that an editor of true skill (himself) is needed to choose the optimal arrangement.

Write a program to determine if zero, one, or more than one arrangement of the essays is possible.

The Input

There will be multiple test cases in the input. Each test case will begin with a line with two integers, n ($1 \le n \le 1{,}000$) and m ($1 \le m \le 500{,}000$), where n is the number of essays, and m is the number of relationships between essays caused by sharing terms. They will be separated by a single space. On each of the next m lines will be two integers, d followed by u ($1 \le d,u \le n$, $d \ne u$) which indicate that some term is defined in essay d and used in essay u. Integers d and u will be separated by a single space. The input will end with two **0**s on their own line.

The Output

For each test case, output a **0** if no arrangement is possible, a **1** if exactly one arrangement is possible, or a **2** if multiple arrangements are possible (output **2** no matter how many arrangements there are). Output no extra spaces, and do not separate answers with blank lines.

Sample Input	Sample Output
5 4 1 5 5 2 3 2 4 3 5 4 3 1 4 2 1 5 5 4 2 2 1 2 2 1 0 0	2 1 0