|  |
| --- |
| Some of UVA Problems |
|  |
|  |
|  |

**Index of problems**

1. 10189 – Minesweeper
2. 10004 - Bicoloring
3. 10041 - Vito's Family
4. 10050 - Hartals
5. 10405 - Longest Common Subsequence
6. 10127 - Ones
7. 10106 - Product
8. 10003 - Cutting Sticks
9. 10137 - The Trip
10. 10131 - Is Bigger Smarter?
11. 10010 - Where's Waldorf?
12. 10066 - The Twin Towers
13. 10220 - I Love Big Numbers !
14. 10494 - If We Were a Child Again
15. 11192 - Group Reverse
16. 11461 - Square Numbers
17. 11479 - Is this the easiest problem?
18. 11650 - Mirror Clock
19. 11586 - Train Tracks
20. 11689 - Soda Surpler
21. 11877 - The Coco-Cola Store
22. 12199 - Lights
23. 11783 - Nails
24. 11080 - Place the Guards
25. 10150 - Doublets
26. 10336 - Rank the Languages
27. 10946 - You want what filled?
28. 11742 - Social Constraints
29. 11242 - Tour de France
30. 11364 - Parking
31. 10801 - Lift Hopping
32. 10986 - Sending email
33. 11005 - Cheapest Base
34. 11494 - Queen
35. 12247 - Jollo
36. 11389 - The Bus Driver Problem
37. 11201 - The problem of the crazy linguist
38. 10718 - Bit Mask
39. 10503 - The dominoes solitaire
40. 11108 - Tautology
41. **10189 - Minesweeper**

Time limit: 3.000 seconds

|  |
| --- |
| **Problem B: Minesweeper** |

## The Problem

Have you ever played Minesweeper? It's a cute little game which comes within a certain Operating System which name we can't really remember. Well, the goal of the game is to find where are all the mines within a MxN field. To help you, the game shows a number in a square which tells you how many mines there are adjacent to that square. For instance, supose the following 4x4 field with 2 mines (which are represented by an \* character):

\*...

....

.\*..

....

If we would represent the same field placing the hint numbers described above, we would end up with:

\*100

2210

1\*10

1110

As you may have already noticed, each square may have at most 8 adjacent squares.

## The Input

The input will consist of an arbitrary number of fields. The first line of each field contains two integers n and m (0 < n,m <= 100) which stands for the number of lines and columns of the field respectively. The next n lines contains exactly m characters and represent the field. Each safe square is represented by an "." character (without the quotes) and each mine square is represented by an "\*" character (also without the quotes). The first field line where n = m = 0 represents the end of input and should not be processed.

## The Output

For each field, you must print the following message in a line alone:

Field #x:

Where x stands for the number of the field (starting from 1). The next n lines should contain the field with the "." characters replaced by the number of adjacent mines to that square. There must be an empty line between field outputs.

## Sample Input

4 4

\*...

....

.\*..

....

3 5

\*\*...

.....

.\*...

0 0

## Sample Output

Field #1:

\*100

2210

1\*10

1110

Field #2:

\*\*100

33200

1\*100

**© 2001 Universidade do Brasil (UFRJ). Internal Contest Warmup 2001.**

### 10004 - Bicoloring

Time limit: 3.000 seconds

|  |
| --- |
| **Bicoloring** |

In 1976 the ``Four Color Map Theorem" was proven with the assistance of a computer. This theorem states that every map can be colored using only four colors, in such a way that no region is colored using the same color as a neighbor region.

Here you are asked to solve a simpler similar problem. You have to decide whether a given arbitrary connected graph can be bicolored. That is, if one can assign colors (from a palette of two) to the nodes in such a way that no two adjacent nodes have the same color. To simplify the problem you can assume:

* no node will have an edge to itself.
* the graph is nondirected. That is, if a node *a* is said to be connected to a node *b*, then you must assume that *b* is connected to *a*.
* the graph will be strongly connected. That is, there will be at least one path from any node to any other node.

## Input

The input consists of several test cases. Each test case starts with a line containing the number *n* ( 1 < *n* < 200) of different nodes. The second line contains the number of edges *l*. After this, *l* lines will follow, each containing two numbers that specify an edge between the two nodes that they represent. A node in the graph will be labeled using a number *a* ( $0 \le a < n$).

An input with *n* = 0 will mark the end of the input and is not to be processed.

## Output

You have to decide whether the input graph can be bicolored or not, and print it as shown below.

## Sample Input

3

3

0 1

1 2

2 0

9

8

0 1

0 2

0 3

0 4

0 5

0 6

0 7

0 8

0

## Sample Output

NOT BICOLORABLE.

BICOLORABLE.

Miguel Revilla   
2000-08-21

### 10041 - Vito's Family

Time limit: 3.000 seconds

|  |
| --- |
| **Problem C: Vito's family** |

## Background

The world-known gangster Vito Deadstone is moving to New York. He has a very big family there, all of them living in Lamafia Avenue. Since he will visit all his relatives very often, he is trying to find a house close to them.

## Problem

Vito wants to minimize the total distance to all of them and has blackmailed you to write a program that solves his problem.

## Input

The input consists of several test cases. The first line contains the number of test cases.

For each test case you will be given the integer number of relatives *r* ( 0 < *r* < 500) and the street numbers (also integers) $s_1, s_2, \ldots, s_i, \ldots, s_r$ where they live ( 0 < *si*< 30000 ). Note that several relatives could live in the same street number.

## Output

For each test case your program must write the minimal sum of distances from the optimal Vito's house to each one of his relatives. The distance between two street numbers *si* and *sj* is *dij*= |*si*-*sj*|.

## Sample Input

2

2 2 4

3 2 4 6

## Sample Output

2

4

Miguel Revilla   
2000-11-19

### 10050 - Hartals

Time limit: 3.000 seconds

|  |
| --- |
| **Problem D: Hartals** |

A social research organization has determined a simple set of parameters to simulate the behavior of the political parties of our country. One of the parameters is a positive integer *h* (called the hartal parameter) that denotes the average number of days between two successive hartals (strikes) called by the corresponding party. Though the parameter is far too simple to be flawless, it can still be used to forecast the damages caused by hartals. The following example will give you a clear idea:

Consider three political parties. Assume *h*1 = 3, *h*2 = 4 and *h*3 = 8 where *hi* is thehartal parameter for party *i* ( *i* = 1, 2, 3). Now, we will simulate the behavior of these three parties for *N* = 14 days. One must always start the simulation on a Sunday and assume that there will be no hartals on weekly holidays (on Fridays and Saturdays).

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 |
| Days |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
|  | Su | Mo | Tu | We | Th | Fr | Sa | Su | Mo | Tu | We | Th | Fr | Sa |
| Party 1 |  |  | x |  |  | x |  |  | x |  |  | x |  |  |
| Party 2 |  |  |  | x |  |  |  | x |  |  |  | x |  |  |
| Party 3 |  |  |  |  |  |  |  | x |  |  |  |  |  |  |
| Hartals |  |  | 1 | 2 |  |  |  | 3 | 4 |  |  | 5 |  |  |

The simulation above shows that there will be exactly 5 hartals (on days 3, 4, 8, 9 and 12) in 14 days. There will be no hartal on day 6 since it is a Friday. Hence we lose 5 working days in 2 weeks.

In this problem, given the hartal parameters for several political parties and the value of*N*, your job is to determine the number of working days we lose in those *N* days.

## Input

The first line of the input consists of a single integer *T* giving the number of test cases to follow.

The first line of each test case contains an integer *N* ( $7 \le N \le 3650$) giving the number of days over which the simulation must be run. The next line contains another integer *P* ( $1 \le P \le 100$) representing the number of political parties in this case. The *i*­th of the next *P* lines contains a positive integer *hi* (which will never be a multiple of 7) giving the hartal parameter for party *i* ( ![$1 \le i \le
P$](data:image/gif;base64,R0lGODlhTwAgAOMAAAAAAJmZmXd3d1VVVe7u7jMzM8zMzBEREaqqqoiIiGZmZv///0RERN3d3SIiIru7uyH5BAEAAAsALAAAAABPACAAQAT+cMlJq704682vAF0zFEnHiaTJAA11HNYjFIAitOZiYIPCEDndRUYTBILIpHLJbDqRn02iMMBpDATGIDOt5hQA4HNMLpvP6PT5oXCAguyCpmeKbxoAwaRxYGCiGCIOCCYEAwRvFYKEQQYAWwsNCAAlf4kbBAkONxkCDpmYmpwZDzVlBA+pqqo7SKirq61qs7S1tre4ubq7vCYNCkpWFwF6OcIWASt9DAybYhaAHQYDB4xO09UcCZQUK5YbBgwFRxwPA8UX4eNBDnITk5XQlxUPBwrHnQGTQ/b4Gx82BBjxNyGaBk+HwAF48M9BQg59ghjkQCCAPQwMBEzrUPEihgAhAKz1WvDg2UgNCJipXAkpSMqVLE/KnEmzps2bOHPqXBABADs=)).

## Output

For each test case in the input output the number of working days we lose. Each output must be on a separate line.

## Sample Input

2

14

3

3

4

8

100

4

12

15

25

40

## Sample Output

5

15

Miguel Revilla   
2000-12-26

### 10405 - Longest Common Subsequence

Time limit: 3.000 seconds

## Problem C: Longest Common Subsequence

Sequence 1: http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405b.gif

Sequence 2: http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405a.gif http://uva.onlinejudge.org/external/104/p10405c.gif http://uva.onlinejudge.org/external/104/p10405b.gif http://uva.onlinejudge.org/external/104/p10405c.gif   
  
  
Given two sequences of characters, print the length of the longest common subsequence of both sequences. For example, the longest common subsequence of the following two sequences:

abcdgh

aedfhr

is adh of length 3.

Input consists of pairs of lines. The first line of a pair contains the first string and the second line contains the second string. Each string is on a separate line and consists of at most 1,000 characters

For each subsequent pair of input lines, output a line containing one integer number which satisfies the criteria stated above.

### Sample input

a1b2c3d4e

zz1yy2xx3ww4vv

abcdgh

aedfhr

abcdefghijklmnopqrstuvwxyz

a0b0c0d0e0f0g0h0i0j0k0l0m0n0o0p0q0r0s0t0u0v0w0x0y0z0

abcdefghijklmnzyxwvutsrqpo

opqrstuvwxyzabcdefghijklmn

### Output for the sample input

4

3

26

14

**Problem Setter: Piotr Rudnicki**

### 10127 - Ones

Time limit: 3.000 seconds

### http://uva.onlinejudge.org/external/101/p10127a.gifhttp://uva.onlinejudge.org/external/101/p10127b.gifProblem E - Ones

Given any integer 0 <= n <= 10000 not divisible by 2 or 5, some multiple of n is a number which in decimal notation is a sequence of 1's. How many digits are in the smallest such a multiple of n?

### Sample input

3

7

9901

### Output for sample input

3

6

12



### 10106 - Product

Time limit: 3.000 seconds

|  |
| --- |
| **Product** |

## The Problem

The problem is to multiply two integers X, Y. (*0<=X,Y<10250*)

## The Input

The input will consist of a set of pairs of lines. Each line in pair contains one multiplyer.

## The Output

For each input pair of lines the output line should consist one integer the product.

## Sample Input

12

12

2

222222222222222222222222

## Sample Output

144

444444444444444444444444

### 10003 - Cutting Sticks

### Time limit: 3.000 seconds

|  |
| --- |
| **Cutting Sticks** |

You have to cut a wood stick into pieces. The most affordable company, The Analog Cutting Machinery, Inc. (ACM), charges money according to the length of the stick being cut. Their procedure of work requires that they only make one cut at a time.

It is easy to notice that different selections in the order of cutting can led to different prices. For example, consider a stick of length 10 meters that has to be cut at 2, 4 and 7 meters from one end. There are several choices. One can be cutting first at 2, then at 4, then at 7. This leads to a price of 10 + 8 + 6 = 24 because the first stick was of 10 meters, the resulting of 8 and the last one of 6. Another choice could be cutting at 4, then at 2, then at 7. This would lead to a price of 10 + 4 + 6 = 20, which is a better price.

Your boss trusts your computer abilities to find out the minimum cost for cutting a given stick.

## Input

The input will consist of several input cases. The first line of each test case will contain a positive number *l* that represents the length of the stick to be cut. You can assume *l* < 1000. The next line will contain the number *n* (*n* < 50) of cuts to be made.

The next line consists of *n* positive numbers *ci* ( 0 < *ci* < *l*) representing the places where the cuts have to be done, given in strictly increasing order.

An input case with *l* = 0 will represent the end of the input.

## Output

You have to print the cost of the optimal solution of the cutting problem, that is the minimum cost of cutting the given stick. Format the output as shown below.

## Sample Input

100

3

25 50 75

10

4

4 5 7 8

0

## Sample Output

The minimum cutting is 200.

The minimum cutting is 22.

Miguel Revilla   
2000-08-21

### 10137 - The Trip

### Time limit: 3.000 seconds

## Problem A: The Trip

A number of students are members of a club that travels annually to exotic locations. Their destinations in the past have included Indianapolis, Phoenix, Nashville, Philadelphia, San Jose, and Atlanta. This spring they are planning a trip to Eindhoven.

The group agrees in advance to share expenses equally, but it is not practical to have them share every expense as it occurs. So individuals in the group pay for particular things, like meals, hotels, taxi rides, plane tickets, etc. After the trip, each student's expenses are tallied and money is exchanged so that the net cost to each is the same, to within one cent. In the past, this money exchange has been tedious and time consuming. Your job is to compute, from a list of expenses, the minimum amount of money that must change hands in order to equalize (within a cent) all the students' costs.

### The Input

Standard input will contain the information for several trips. The information for each trip consists of a line containing a positive integer, n, the number of students on the trip, followed by n lines of input, each containing the amount, in dollars and cents, spent by a student. There are no more than 1000 students and no student spent more than $10,000.00. A single line containing 0 follows the information for the last trip.

### The Output

For each trip, output a line stating the total amount of money, in dollars and cents, that must be exchanged to equalize the students' costs.

### Sample Input

3

10.00

20.00

30.00

4

15.00

15.01

3.00

3.01

0

### Output for Sample Input

$10.00

$11.99

### 10131 - Is Bigger Smarter?

### Time limit: 3.000 seconds

# Question 1: Is Bigger Smarter?

## The Problem

Some people think that the bigger an elephant is, the smarter it is. To disprove this, you want to take the data on a collection of elephants and put as large a subset of this data as possible into a sequence so that the weights are increasing, but the IQ's are decreasing.

The input will consist of data for a bunch of elephants, one elephant per line, terminated by the end-of-file. The data for a particular elephant will consist of a pair of integers: the first representing its size in kilograms and the second representing its IQ in hundredths of IQ points. Both integers are between 1 and 10000. The data will contain information for at most 1000 elephants. Two elephants may have the same weight, the same IQ, or even the same weight and IQ.

Say that the numbers on the i-th data line are W[i] and S[i]. Your program should output a sequence of lines of data; the first line should contain a number n; the remaining n lines should each contain a single positive integer (each one representing an elephant). If these n integers are a[1], a[2],..., a[n] then it must be the case that

W[a[1]] < W[a[2]] < ... < W[a[n]]

and

S[a[1]] > S[a[2]] > ... > S[a[n]]

In order for the answer to be correct, n should be as large as possible. All inequalities are strict: weights must be strictly increasing, and IQs must be strictly decreasing. There may be many correct outputs for a given input, your program only needs to find one.

## Sample Input

6008 1300

6000 2100

500 2000

1000 4000

1100 3000

6000 2000

8000 1400

6000 1200

2000 1900

## Sample Output

4

4

5

9

7

### 10010 - Where's Waldorf?

### Time limit: 3.000 seconds

|  |
| --- |
| **Where's Waldorf?** |

Given a *m* by *n* grid of letters, ( $1 \leq m,n \leq 20$), and a list of words, find the location in the grid at which the word can be found. A word matches a straight, uninterrupted line of letters in the grid. A word can match the letters in the grid regardless of case (i.e. upper and lower case letters are to be treated as the same). The matching can be done in any of the eight directions either horizontally, vertically or diagonally through the grid.

## Input

**The input begins with a single positive integer on a line by itself indicating the number of the cases following, each of them as described below. This line is followed by a blank line, and there is also a blank line between two consecutive inputs.**

The input begins with a pair of integers, *m* followed by *n*, $1 \leq
m,n \leq 50$ in decimal notation on a single line. The next *m* lines contain *n* letters each; this is the grid of letters in which the words of the list must be found. The letters in the grid may be in upper or lower case. Following the grid of letters, another integer *k* appears on a line by itself ($1 \leq k \leq 20$). The next *k* lines of input contain the list of words to search for, one word per line. These words may contain upper and lower case letters only (no spaces, hyphens or other non-alphabetic characters).

## Output

**For each test case, the output must follow the description below. The outputs of two consecutive cases will be separated by a blank line.**

For each word in the word list, a pair of integers representing the location of the corresponding word in the grid must be output. The integers must be separated by a single space. The first integer is the line in the grid where the first letter of the given word can be found (1 represents the topmost line in the grid, and *m* represents the bottommost line). The second integer is the column in the grid where the first letter of the given word can be found (1 represents the leftmost column in the grid, and *n*represents the rightmost column in the grid). If a word can be found more than once in the grid, then the location which is output should correspond to the uppermost occurence of the word (i.e. the occurence which places the first letter of the word closest to the top of the grid). If two or more words are uppermost, the output should correspond to the leftmost of these occurences. All words can be found at least once in the grid.

## Sample Input

1

8 11

abcDEFGhigg

hEbkWalDork

FtyAwaldORm

FtsimrLqsrc

byoArBeDeyv

Klcbqwikomk

strEBGadhrb

yUiqlxcnBjf

4

Waldorf

Bambi

Betty

Dagbert

## Sample Output

2 5

2 3

1 2

7 8

Miguel Revilla   
2000-08-22

### 10066 - The Twin Towers

### Time limit: 3.000 seconds

**Problem B**

**The Twin Towers**

**Input:**standard input

**Output:**standard output

Once upon a time, in an ancient Empire, there were two towers of dissimilar shapes in two different cities. The towers were built by putting circular tiles one upon another. Each of the tiles was of the same height and had integral radius. It is no wonder that though the two towers were of dissimilar shape, they had many tiles in common.

However, more than thousand years after they were built, the Emperor ordered his architects to remove some of the tiles from the two towers so that they have exactly the same shape and size, and at the same time remain as high as possible. The order of the tiles in the new towers must remain the same as they were in the original towers. The Emperor thought that, in this way the two towers might be able to stand as the symbol of harmony and equality between the two cities. He decided to name them the*Twin Towers*.

Now, about two thousand years later, you are challenged with an even simpler problem: given the descriptions of two dissimilar towers you are asked only to find out the number of tiles in the highest twin towers that can be built from them.

**Input**

The input file consists of several data blocks. Each data block describes a pair of towers.

The first line of a data block contains two integers N1 and N2 (1 <= N1, N2 <= 100) indicating the number of tiles respectively in the two towers. The next line contains N1 positive integers giving the radii of the tiles (from top to bottom) in the first tower. Then follows another line containing N2 integers giving the radii of the tiles (from top to bottom) in the second tower.

The input file terminates with two zeros for N1 and N2.

**Output**

For each pair of towers in the input first output the twin tower number followed by the number of tiles (in one tower) in the highest possible twin towers that can be built from them. Print a blank line after the output of each data set.

**Sample Input**

7 6  
20 15 10 15 25 20 15  
15 25 10 20 15 20  
8 9  
10 20 20 10 20 10 20 10  
20 10 20 10 10 20 10 10 20  
0 0

### Sample Output

### Twin Towers #1 Number of Tiles : 4 Twin Towers #2 Number of Tiles : 6

### 10220 - I Love Big Numbers !

### Time limit: 3.000 seconds

|  |
| --- |
| **I Love Big Numbers !** |

## The Problem

A Japanese young girl went to a Science Fair at Tokyo. There she met with a Robot named Mico-12, which had AI (You must know about AI-Artificial Intelligence). The Japanese girl thought, she can do some fun with that Robot. She asked her, "Do you have any idea about maths ?"."Yes! I love mathematics", The Robot replied.

"Okey ! Then I am giving you a number, you have to find out the Factorial of that number. Then find the sum of the digits of your result!. Suppose the number is 5.You first calculate 5!=120, then find sum of the digits 1+2+0=3.Can you do it?"

"Yes. I can do!"Robot replied.

"Suppose the number is 100, what will be the result ?".At this point the Robot started thinking and calculating. After a few minutes the Robot head burned out and it cried out loudly "Time Limit Exceeds".

The girl laughed at the Robot and said "The sum is definitely 648".

"How can you tell that ?" Robot asked the girl. "Because I am an ACM World Finalist and I can solve the Big Number problems easily." Saying this, the girl closed her laptop computer and went away.

Now, your task is to help the Robot with the similar problem.

## The Input

The input file will contain one or more test cases.

Each test case consists of one line containing an integers n (n<=1000).

## The Output

For each test case, print one line containing the required number. This number will always fit into an integer, i.e. it will be less than 2^31-1.

## Sample Input

5

60

100

## Sample Output

3

288

648

--------------------------------------------------------------------------------------------------------------------

**Ahmed Shamsul Arefin**

1. **10494 - If We Were a Child Again**

Time limit: 3.000 seconds

**Problem C**  
**If We Were a Child Again**

**Input:** standard input  
**Output:** standard output

**Time Limit:**7 seconds

|  |  |  |
| --- | --- | --- |
| “Oooooooooooooooh!  If I could do the easy mathematics like my school days!!  I can guarantee, that I’d not make any mistake this time!!”  Says a smart university student!!  But his teacher even smarter – “Ok! I’d assign you such projects in your software lab. Don’t be so sad.”  “Really!!” - the students feels happy. And he feels so happy that he cannot see the smile in his teacher’s face. | | http://uva.onlinejudge.org/external/104/p10494.gif |
| **The Problem**    The first project for the poor student was to make a calculator that can just perform the basic arithmetic operations.    But like many other university students he doesn’t like to do any project by himself. He just wants to collect programs from here and there. As you are a friend of him, he asks you to write the program. But, you are also intelligent enough to tackle this kind of people. You agreed to write only the (integer) division and mod (% in C/C++) operations for him. | | |
| **Input**  Input is a sequence of lines. Each line will contain an input number. One or more spaces. A sign (division or mod). Again spaces. And another input number. Both the input numbers are non-negative integer. The first one may be arbitrarily long. The second number **n** will be in the range (0 < n < 231). | | |
| Output A line for each input, each containing an integer. See the sample input and output. Output should not contain any extra space. | | |
| Sample Input 110 / 100  99 % 10  2147483647 / 2147483647  2147483646 % 2147483647 |  | |
| Sample Output 1  9  1  2147483646 |  | |
|  |  |  |

**Problemsetter:  S. M. Ashraful Kadir, University of Dhaka**

1. **11192 - Group Reverse**

Time limit: 3.000 seconds

**Problem G**  
**Group Reverse**  
**Input:**Standard Input

**Output:**Standard Output

Group reversing a string means reversing a string by groups. For example consider a string:

“**TOBENUMBERONEWEMEETAGAINANDAGAINUNDERBLUEICPCSKY**”

This string has length 48. We have divided into 8 groups of equal length and so the length of each group is 6. Now we can reverse each of these eight groups to get a new string:

**“UNEBOTNOREBMEEMEWENIAGATAGADNAEDNUNIIEULBRYKSCPC”**

Given the string and number of groups in it, your program will have to group reverse it.

**Input**

The input file contains at most 101 lines of inputs. Each line contains at integer G (G<10) which denotes the number of groups followed by a string whose length is a multiple of G. The length of the string is not greater than 100. The string contains only alpha numerals. Input is terminated by a line containing a single zero.

#### Output

For each line of input produce one line of output which contains the group reversed string.

# Sample Input                             Output for Sample Input

|  |  |
| --- | --- |
| **3 ABCEHSHSH**  **5 FA0ETASINAHGRI0NATWON0QA0NARI0**  **0** | **CBASHEHSH**  **ATE0AFGHANISTAN0IRAQ0NOW0IRAN0** |

**Problem-setter: Shahriar Manzoor**

**Special Thanks: Derek Kisman**

1. **11461 - Square Numbers**

Time limit: 1.000 seconds

|  |  |  |
| --- | --- | --- |
| **A** | Square Numbers  **Input:** Standard Input  **Output:** Standard Output |  |

A square number is an integer number whose square root is also an integer. For example 1, 4, 81 are some square numbers. Given two numbers a and b you will have to find out how many square numbers are there between a and b (inclusive).

##### Input

The input file contains at most 201 lines of inputs. Each line contains two integers a and b (0<a≤b≤100000). Input is terminated by a line containing two zeroes. This line should not be processed.

#### Output

For each line of input produce one line of output. This line contains an integer which denotes how many square numbers are there between a and b (inclusive).

# Sample Input                             Output for Sample Input

|  |  |
| --- | --- |
| **1 4**  **1 10**  **0 0** | **2**  **3** |

1. **11479 - Is this the easiest problem?**

Time limit: 1.000 seconds

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Problem I** | **Is this the easiest problem?** | **Time Limit : 1 second** |
|  | A triangle is a geometric shape with three positive sides. However, any given three sides won’t necessarily form a triangle. The three sides must form a closed region. Triangles are categorized depending on the values of the sides of a valid triangle. In this problem you are required to determine the type of a triangle. | | | |  |
|  | **Input** | | |  |  |
|  | The first line of input will contain a positive integer **T<20,**where **T** denotes the number of test cases. Each of the next **T** lines will contain three 32 bit signed integer. | | | |  |
|  |  | | | |  |
|  | **Output** | | | |  |
|  | For each case of  input there will be one line of output. It will be formatted as:  **Case {x}: {triangle type}**. Where **x** denotes the case number being processed and **{triangle type}** is the type of the triangle..**{triangle type}**will be one of the following, depending on the values of the three sides:   * + Invalid - The three sides can not form a triangle   + Equilateral  - All three sides of valid triangle are equal   + Isosceles  - Exactly two of the sides of a valid triangle are equal.   + Scalene - No pair of sides are equal in a valid triangle. | | | |  |
|  |  | | | |  |
|  | **Sample Input** | **Sample Output** |  |  |  |
|  | **4 1 2 5 1 1 1 4 4 2 3 4 5** | **Case 1: Invalid Case 2: Equilateral Case 3: Isosceles Case 4: Scalene** |  |  |  |
|  | **Problem Setter: Shamim Hafiz  Next Generation Contest 5** | | | |  |  |  |

### 11650 - Mirror Clock

Time limit: 1.000 seconds

|  |  |
| --- | --- |
| **E** | Mirror Clock  **Input:** Standard Input  **Output:** Standard Output |

Little Onuvob has learnt to read wall clock.Very often he loudly announces the time. He has a big clock in his room. There is also a mirror in the opposite side of the clock. Few days ago his mother noticed that sometimes Onuvob is announcing wrong time. She became worried and after some investigation she found, Onuvob has no problem in reading time from the clock, the thing is sometimes he reads from the clock in the mirror. So she told him

“Onuvob, you shouldn’t read time from the clock in the mirror.”

“But why ma?” curious Onuvob asked.

“Because it gives wrong time” mother replied.

Now Onuvob looked at the two clocks and realized his mother was telling the truth. Two clocks were not matching. He thought for some time and shouted, “But ma, we can calculate the real time from the clock in the mirror, can’t we?”

Can we? Now you have to answer.

### Input

The first line of the input contains an integer T (T < 1000), number of test cases. Each of the following T lines will contain the time Onuvob reads from the clock in the mirror. The time will be in the following format: HH:MM

Hours and minutes both are integer and the time is always valid.

## Output

For each line of Input you have to produce one line of output which will contain the real time, in the same format, namely:

HH:MM

See sample input and output for more clarification.

# Sample Input                              Output for Sample Input

|  |  |
| --- | --- |
| 2  12:00  10:09 | 12:00  01: 51 |

**Problem setter: Md. Towhidul Islam, Special Thanks: Sohel Hafiz**

### 11586 - Train Tracks

Time limit: 1.000 seconds

## Problem J: Train Tracks

Andy loves his set of wooden trains and railroad tracks. Each day, Daddy has to build a new track for him. The tracks he likes best form a simple loop with no branches or dead ends, so he can run his trains around and around for hours until it is time for the big crash that destroys the whole construction.

So here is the question: Given a set of track pieces, can you form a simple loop with them, while using up all the pieces?

Each piece of track is described by the connectors at both ends. A standard piece has one "male" and one "female" connector. But there are also track pieces with two male or two female connectors, as shown in the front right of the picture.

To fit together, each male connector must be connected to a female connector. Unlike real wooden tracks, our pieces are assumed to be flexible, so their length or shape is not an issue here. However, you may not connect the two ends of the same piece together.

Input begins with the number of test cases. Each following line contains one test case. Each test case consists of a list of between 1 and 50 (inclusive) train track pieces. A piece is described by two code letters: M for male or F for female connector. Pieces are separated by space characters.

For each test case, output a line containing either LOOP or NO LOOP to indicate whether or not all the pieces can be joined into a single loop.

### Sample input

4

MF MF

FM FF MF MM

MM FF

MF MF MF MF FF

### Sample output

LOOP

LOOP

LOOP

NO LOOP

Martin Müller

### 11689 - Soda Surpler

Time limit: 1.000 seconds

# Problem A - Soda Surpler

### Time limit: X seconds



Tim is an absolutely obsessive soda drinker, he simply cannot get enough. Most annoyingly though, he almost never has any money, so his only obvious legal way to obtain more soda is to take the money he gets when he recycles empty soda bottles to buy new ones. In addition to the empty bottles resulting from his own consumption he sometimes find empty bottles in the street. One day he was extra thirsty, so he actually drank sodas until he couldn't afford a new one.

## Input

The first line of the input file contains an integer N (N<15) which denotes the total number of test cases. The description of each test case is given below:  
  
Three non-negative integers *e,f,c*, where *e<1000* equals the number of empty soda bottles in Tim's possession at the start of the day, *f<1000* the number of empty soda bottles found during the day, and*1<c<2000* the number of empty bottles required to buy a new soda.

## Output

For each test case print how many sodas did Tim drink on his extra thirsty day? Look at the sample output for details.

## Sample Input

2

9 0 3

5 5 2

## Sample Output

4

9

The 2009 ACM Nordic Collegiate Programming Contest

### 11877 - The Coco-Cola Store

Time limit: 1.000 seconds

|  |
| --- |
| **The Coco-Cola Store** |

� Once upon a time, there is a special coco-cola store. If you return three empty bottles to the shop, you'll get a full bottle of coco-cola to drink. If you have n empty bottles right in your hand, how many full bottles of coco-cola can you drink?

## Input

There will be at most 10 test cases, each containing a single line with an integer *n* ( 1$ \le$*n*$ \le$100). The input terminates with *n* = 0, which should not be processed.

## Output

For each test case, print the number of full bottles of coco-cola that you can drink.

## Sample Input

3

10

81

0

## Sample Output

1

5

40

**Spoiler**

Let me tell you how to drink 5 full bottles with 10 empty bottles: get 3 full bottles with 9 empty bottles, drink them to get 3 empty bottles, and again get a full bottle from them. Now you have 2 empty bottles. Borrow another empty bottle from the shop, then get another full bottle. Drink it, and finally return this empty bottle to the shop!

Problemsetter: Rujia Liu, Special Thanks: Yiming Li & Sohel Hafi3z

### 12199 - Lights

Time limit: 3.000 seconds

John has *n* light bulbs and a switchboard with *n* switches; each bulb can be either on or off, and pressing the *i* -th switch changes the state of bulb *i* from on to off, and viceversa. He is using them to play a game he has made up. In each move, John selects a (possibly empty) set of switches and presses them, thus inverting the states of the corresponding bulbs. Initially all lights are off, and after exactly *m* moves John would like to have the first *v* bulbs on and the rest off; otherwise he loses the game. There is only one restriction: he is not allowed to press the same setof switches in two different moves.

This is quite an easy game, as there are lots of ways of winning. This has encouraged him to keep playing different winning games, and now he is intent on trying them all. Help him count how many ways of winning there are. Two games are considered the same if, after a reordering of the moves in one of them, at every step the same set of switches is pressed in both of them.

For example, if *n* = 4, *m* = 3, and *v* = 2, one possible winning game is obtained by pressing switches 1, 2 and 4 in the first move, 1 and 3 in the second one, and 1, 3 and 4 in the last one. This is considered equivalent to, say, first pressing 1 and 3; then 1, 2, 4; and then 1, 3, 4.

## Input

The input has at most 500 lines, one for each test case. Each line contains three integers *n* ( 1$ \le$*n*$ \le$1 000), *m* ( 1$ \le$*m*$ \le$1 000), and *v* ( 0$ \le$*v*$ \le$*n*). The last line of input will hold the values 0 0 0 and must not be processed.

## Output

Print one line for each test case containing the number of ways John can play the game, modulo the prime 10 567 201.

## Sample Input

3 3 1

6 4 0

6 4 3

0 0 0

## Sample Output

7

10416

9920

### 11783 - Nails

Time limit: 1.000 seconds

|  |
| --- |
| **Nails** |

The World Of Nails (WON) is a famous hardware store specialized in selling hundreds of different kinds of nails. Some of them very old, rare and expensive. However, the store also offers some standard hand tools and wood. Rado, the store's manager, has two kids who usually go to the store after school and spend the afternoon playing with the tools. During all this time, the kids play with hammers, nails, screws, wood, etc. (quite dangerous isn't it?).

One day, the kids were playing in the wood deposit. By accident, they hit a huge pile of wood sticks which then collapsed. A lot of wood sticks fell to the floor, but did not hit any of the kids. The kids did not get scare and so they continued playing. They took a bag of very expensive nails and a hammer and decided to hammer the nails on the wood sticks, just for fun. If a wood stick were laying in top of another, they joined these two pieces together by hammering exactly one nail where they crossed, see Figure 1. No more than two stick are crossed in the same point such as shown on third configuration in figure below. If a wood stick was laying on the floor all alone, with no other stick on top of it, they hammered the stick on to the floor using two nails, one for each end of the stick.

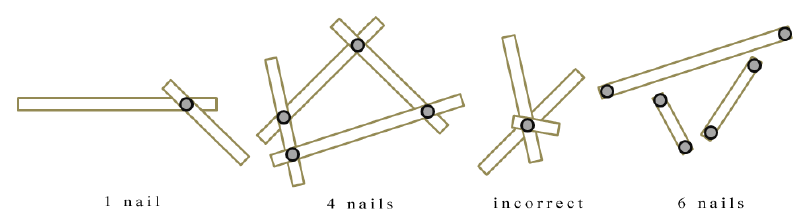


Figure 1: Different configurations

After a couple of hours, Rado discovered the disaster in the deposit. When he noticed what kind of nails were the kids using, he was shocked. A lot of money wasted on a little game. Rado wants you to help him calculate how many nails did they kids use, to calculate how much money he has lost.

## Input

The input consists in several test cases. Each case corresponds to a set of wooden sticks. The first line of a test case contains one integer ( 0$ \le$*N*$ \le$1000), indicating the number of sticks. Next, *N* lines follow, each one corresponding to a stick. Each line contains four integer values ( 0$ \le$*X*1, *Y*1, *X*2, *Y*2$ \le$1000) separated of each other by exactly one blank space. The first two integers correspond to the Cartesian coordinates of one end of the wood stick. The later two integers correspond to the Cartesian coordinate of the other end of the wood stick. Assume no stick is completely superposed by another.

The end of input is indicated by a test case with *N* = 0.

## Output

For each case specified in the input, the output is a single line containing the number of nails used by the kids.

## Sample Input

3

0 0 3 0

1 1 3 3

1 2 2 2

3

0 0 3 0

1 1 0 0

1 0 2 2

3

1 1 2 2

0 0 5 0

0 1 0 5

0

## Sample Output

3

2

6

### 11080 - Place the Guards

Time limit: 3.000 seconds

**Problem G**

**Place the Guards**  
**Input:**Standard Input

**Output:**Standard Output

In the country of Ajabdesh there are some streets and junctions. Each street connects 2 junctions. The king of Ajabdesh wants to place some guards in some junctions so that all the junctions and streets can be guarded by them. A guard in a junction can guard all the junctions and streets adjacent to it. But the guards themselves are not gentle. If a street is guarded by multiple guards then they start fighting. So the king does not want the scenario where a street may be guarded by two guards. Given the information about the streets and junctions of Ajabdesh, help the king to find the minimum number of guards needed to guard all the junctions and streets of his country.

**Input:**

The first line of the input contains a single integer T (T<80) indicating the number of test cases. Each test case begins with 2 integers v (1 **≤**v **≤**200) and e (0 **≤**e **≤**10000.). vis the number of junctions and e is the number of streets. Each of the next e line contains 2 integer f and t denoting that there is a street between f and t. All the junctions are numbered from 0 to v-1.

**Output:**

For each test case output in a single line an integer m denoting the minimum number of guards needed to guard all the junctions and streets. Set the value of m as -1 if it is impossible to place the guards without fighting.

# Sample Input                             Output for Sample Input

|  |  |
| --- | --- |
| **2**  **4 2**  **0 1**  **2 3**  **5 5**  **0 1**  **1 2**  **2 3**  **0 4**  **3 4** | **2**  **-1** |

Problemsetter: Abdullah-al-Mahmud

Special Thanks: Md. Kamruzzaman

### 10150 - Doublets

Time limit: 3.000 seconds

## Problem B: Doublets

A Doublet is a pair of words that differ in exactly one letter; for example, "booster" and "rooster" or "rooster" and "roaster" or "roaster" and "roasted".

You are given a dictionary of up to 25143 lower case words, not exceeding 16 letters each. You are then given a number of pairs of words. For each pair of words, find the shortest sequence of words that begins with the first word and ends with the second, such that each pair of adjacent words is a doublet. For example, if you were given the input pair "booster" and "roasted", a possible solution would be: ("booster", "rooster", "roaster", "roasted") provided that these words are all in the dictionary.

### The Input

Input consists of the dictionary followed by a number of word pairs. The dictionary consists of a number of words, one per line, and is terminated by an empty line. The pairs of input words follow; the words of each pair occur on a line separated by a space.

### The Output

For each input pair, print a set of lines starting with the first word and ending with the last. Each pair of adjacent lines must be a doublet. If there are several minimal solutions, any one will do. If there is no solution, print a line: "No solution." Leave a blank line between cases.

### Sample Input

booster

rooster

roaster

coasted

roasted

coastal

postal

booster roasted

coastal postal

### Output for Sample Input

booster

rooster

roaster

roasted

No solution.

### 10336 - Rank the Languages

Time limit: 3.000 seconds

**Problem A**

**Rank the Languages**

**Input:**standard input

**Output:**standard output

**Time Limit:** 2 seconds

**Memory Limit:**32 MB

You might have noticed that English and Spanish are spoken in many areas all over the world. Now it would be nice to rank all languages according to the number of states where they are spoken.

## Problem

You're given a map which shows the states and the languages where they are spoken. Look at the following map:

ttuuttdd

ttuuttdd

uuttuudd

uuttuudd

The map is read like this: Every letter stands for a language and states are defined as connected areas with the same letter. Two letters are "connected" if one is at left, at right, above or below the other one. So in the above map, there are three states where the language "t" is spoken, three where "u" is spoken and one state where people speak "d".

Your job is to determine the number of states for each language and print the results in a certain order.

## Input

The first line contains the number of test cases N. Each test case consists of a line with two numbers H and W, which are the height and the width of the map. Then follow H lines with a string of W letters. Those letters will only be lowercase letters from "a" to "z".

## Output

For each test case print "World #n", where n is the number of the test case. After that print a line for each language that appears in the test case, which contains the language, a colon, a space and the number of states, where that language is spoken. These lines have to be ordered decreasingly by the number of states. If two languages are spoken in the same number of states, they have to appear alphabetically, which means language "i" comes before language "q", for example.

## Sample Input

2

4 8

ttuuttdd

ttuuttdd

uuttuudd

uuttuudd

9 9

bbbbbbbbb

aaaaaaaab

bbbbbbbab

baaaaacab

bacccccab

bacbbbcab

bacccccab

baaaaaaab

bbbbbbbbb

## Sample Output

World #1

t: 3

u: 3

d: 1

World #2

b: 2

a: 1

c: 1

**Stefan Pochmann**

### 10946 - You want what filled?

Time limit: 3.000 seconds

## Problem D: You want what filled?

Now that Ryan and Larry infuriated the bear, they're now forced to do menial tasks for food, one such task is filling in potholes on this deserted island. But of course, it's never quite that easy, as the bear forces them to fill in the biggest hole first. Since Ryan and Larry are still lazy (very little has changed, you see), can you tell them the order to fill in the holes? 

### Input

The first line will contain two numbers **x** and **y**, followed by **x** lines of **y** characters each. (x and y are less than 50). The holes will be represented by the uppercase characters A to Z, and regular land will be represented by ".". There will be no other characters in the map. Input will be terminated by having 0 0 as input.

### Output

For each map, output the problem number (as shown below), then output the hole represented by the character, and the number of space the hole takes up, sorted by the size of the hole, break ties by sorting the characters in alphabetical order, as shown in the sample output on a separate line as shown below: 

### Sample Input

5 5

..AAA

E.BBB

..AA.

CC.DD

CC.D.

5 5

..AAA

E.BBB

..AA.

CC.DD

CC.D.

0 0

### Sample Output

Problem 1:

C 4

A 3

B 3

D 3

A 2

E 1

Problem 2:

C 4

A 3

B 3

D 3

A 2

E 1

### 11742 - Social Constraints

Time limit: 3.000 seconds

## Problem A: Social Constraints

Socializing can be a very complicated thing among teenagers. For example, finding a good seating arrangement in a movie theater can be a difficult task. Here is a list of constraints that could potentially apply to two individuals A and B in this situation:

* if A and B are dating, then they must sit beside each other
* if A and B are fighting, then they cannot sit beside each other
* if A and B have just broke up, then they must sit at opposite ends of the row

Teenage politics is a complicated thing meaning the constraints can get even more complicated than those listed above. However, we restrict this problem to a particular form of constraint that simply specifies a lower or upper bound on the number of seats separating two specific individuals.

The group arrives after everyone else watching the show has been seated. By some stroke of luck, there are exactly as many open seats as there are teenagers and all of these seats appear consecutively in the front row. How many possible seating arrangements satisfy the constraints?

### Input Format

Each test case begins with two integers n and m with 0 < n ≤ 8 and 0 ≤ m ≤ 20 where n is the size of the group. For simplicity, assume the teenagers are numbered from 0 to n-1. Then of m lines follow, each describing a constraint, where a line consists of three integers a,b,c satisfying 0 ≤ a < b < n and 0 < |c| < n. If c is positive then teenagers a and b must sit at most c seats apart. If c is negative, then a and b must sit at least -c seats apart. The end of input is signaled by a line consisting of n = m = 0.

### Output Format

The output for each test case is a single line containing the number of possible seating arrangements for the group that satisfy all of the social constraints.

### Sample Input

3 1

0 1 -2

3 0

0 0

### Sample Output

2

6

Zachary Friggstad

### 11242 - Tour de France

Time limit: 3.000 seconds

## http://uva.onlinejudge.org/external/112/p11242.gifProblem D: Tour de France

A racing bicycle is driven by a chain connecting two sprockets. Sprockets are grouped into two clusters: the front cluster (typically consisting of 2 or 3 sprockets) and the rear cluster (typically consisting of between 5 and 10 sprockets). At any time the chain connects one of the front sprockets to one of the rear sprockets. The drive ratio -- the ratio of the angular velocity of the pedals to that of the wheels -- is n:m where n is the number of teeth on the rear sprocket and m is the number of teeth on the front sprocket. Two drive ratios d1<d2 are adjacent if there is no other drive ratio d1<d3<d2. The *spread* between a pair of drive ratios d1<d2 is their quotient: d2/d1. You are to compute the maximum spread between two adjacent drive ratios achieved by a particular pair of front and rear clusters.

Input consists of several test cases, followed by a line containing 0. Each test case is specified by the following input:

* f: the number of sprockets in the front cluster;
* r: the number of sprockets in the rear cluster;
* f integers, each giving the number of teeth on one of the gears in the front cluster;
* r integers, each giving the number of teeth on one of the gears in the rear cluster.

You may assume that no cluster has more than 10 sprockets and that no gear has fewer than 10 or more than 100 teeth.

For each test case, output the maximum spread rounded to two decimal places.

### Sample Input

2 4

40 50

12 14 16 19

0

### Output for Sample Input

1.19

*Ondrej Lhotak and Gordon V. Cormack*

### 11364 - Parking

Time limit: 1.000 seconds

|  |
| --- |
| **C: Optimal Parking** |

When shopping on Long Street, Michael usually parks his car at some random location, and then walks to the stores he needs. Can you help Michael choose a place to park which minimises the distance he needs to walk on his shopping round?

Long Street is a straight line, where all positions are integer. You pay for parking in a specific slot, which is an integer position on Long Street. Michael does not want to pay for more than one parking though. He is very strong, and does not mind carrying all the bags around.

## Input

The first line of input gives the number of test cases, 1$ \le$*t*$ \le$100 . There are two lines for each test case. The first gives the number of stores Michael wants to visit, 1$ \le$*n*$ \le$20 , and the second gives their *n* integer positions on Long Street, 0$ \le$*x*i$ \le$99 .

## Output

Output for each test case a line with the minimal distance Michael must walk given optimal parking.

## Sample Input

2

4

24 13 89 37

6

7 30 41 14 39 42

## Sample Output

152

70

### 10801 - Lift Hopping

Time limit: 3.000 seconds

**Problem ?**  
**Lift Hopping**  
Time Limit: 1 second

|  |
| --- |
| Ted the bellhop: *"I'm coming up and if there isn't a dead body by the time I get there, I'll make one myself. You!"* |

Robert Rodriguez, "Four Rooms."

A skyscraper has no more than 100 floors, numbered from 0 to 99. It has **n**(1<=**n**<=5) elevators which travel up and down at (possibly) different speeds. For each **i** in {1, 2,... n}, elevator number **i** takes **Ti** (1<=**Ti**<=100) seconds to travel between any two adjacent floors (going up or down). Elevators do not necessarily stop at every floor. What's worse, not every floor is necessarily accessible by an elevator.

You are on floor 0 and would like to get to floor **k** as quickly as possible. Assume that you do not need to wait to board the first elevator you step into and (for simplicity) the operation of switching an elevator on some floor always takes exactly a minute. Of course, both elevators have to stop at that floor. You are forbiden from using the staircase. No one else is in the elevator with you, so you don't have to stop if you don't want to. Calculate the minimum number of seconds required to get from floor 0 to floor**k** (passing floor **k** while inside an elevator that does not stop there does not count as "getting to floor **k**").

**Input**  
The input will consist of a number of test cases. Each test case will begin with two numbers, **n** and **k**, on a line. The next line will contain the numbers **T1**, **T2**,... **Tn**. Finally, the next **n** lines will contain sorted lists of integers - the first line will list the floors visited by elevator number 1, the next one will list the floors visited by elevator number 2, etc.

**Output**

For each test case, output one number on a line by itself - the minimum number of seconds required to get to floor **k** from floor 0. If it is impossible to do, print "IMPOSSIBLE" instead.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 2 30  10 5  0 1 3 5 7 9 11 13 15 20 99  4 13 15 19 20 25 30  2 30  10 1  0 5 10 12 14 20 25 30  2 4 6 8 10 12 14 22 25 28 29  3 50  10 50 100  0 10 30 40  0 20 30  0 20 50  1 1  2  0 2 4 6 8 10 | 275  285  3920  IMPOSSIBLE |

**Explanation of examples**

In the first example, take elevator 1 to floor 13 (130 seconds), wait 60 seconds to switch to elevator 2 and ride it to floor 30 (85 seconds) for a total of 275 seconds.

In the second example, take elevator 1 to floor 10, switch to elevator 2 and ride it until floor 25. There, switch back to elevator 1 and get off at the 30'th floor. The total time is   
10\*10 + 60 + 15\*1 + 60 + 5\*10 = 285 seconds.

In example 3, take elevator 1 to floor 30, then elevator 2 to floor 20 and then elevator 3 to floor 50.

In the last example, the one elevator does not stop at floor 1.

**Problemsetter: Igor Naverniouk**  
**Alternate solutions: Stefan Pochmann, Frank Pok Man Chu**

### 10986 - Sending email

Time limit: 3.000 seconds

**Problem E**  
**Sending email**  
Time Limit: 3 seconds

|  |
| --- |
| *"A new internet watchdog is creating a stir in Springfield. Mr. X, if that is his real name, has come up with a sensational scoop."* |

Kent Brockman

There are **n** SMTP servers connected by network cables. Each of the **m** cables connects two computers and has a certain latency measured in milliseconds required to send an email message. What is the shortest time required to send a message from server **S** to server **T** along a sequence of cables? Assume that there is no delay incurred at any of the servers.

**Input**  
The first line of input gives the number of cases, **N**. **N** test cases follow. Each one starts with a line containing **n** (2<=**n**<20000), **m** (0<=**m**<50000), **S** (0<=**S**<**n**) and **T**(0<=**T**<**n**). **S**!=**T**. The next **m** lines will each contain 3 integers: 2 different servers (in the range [0, **n**-1]) that are connected by a bidirectional cable and the latency, **w**, along this cable (0<=**w**<=10000).

**Output**  
For each test case, output the line "Case #**x**:" followed by the number of milliseconds required to send a message from **S** to **T**. Print "unreachable" if there is no route from **S**to **T**.

|  |  |
| --- | --- |
| **Sample Input** | **Sample Output** |
| 3  2 1 0 1  0 1 100  3 3 2 0  0 1 100  0 2 200  1 2 50  2 0 0 1 | Case #1: 100  Case #2: 150  Case #3: unreachable |

**Problemsetter: Igor Naverniouk**

### 11005 - Cheapest Base

Time limit: 3.000 seconds

**Problem B**  
**Cheapest Base**  
**Input:**Standard Input

**Output:**Standard Output

When printing text on paper we need ink. But not every character needs the same amount of ink to print: letters such as 'W', 'M' and '8' are more expensive than thinner letters as 'i', 'c' and '1'. In this problem we will evaluate the cost of printing numbers in several bases.

As you know, numbers can be expressed in several different bases. Well known bases are binary (base 2; digits 0 and 1), decimal (base 10; digits 0 to 9) and hexadecimal (base 16; digits 0 to 9 and letters A to F). For the general base **n** we will use the first **n**characters of the string "0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ", which means the highest base in this problem is 36. The lowest base is of course 2.

Every character from this string has an associated cost, represented by an integer value between 1 and 128. The cost to print a number in a certain base is the sum of the costs of all characters needed to represent that number. For the numbers given in the input, you will have to calculate the cheapest base(s) to represent this number in. Numbers in any base are printed without leading zeros.

##### Input

The input has less than 25 test cases. The first line of input file denotes this number of test cases. The description of every test case is given below:

 The first 4 lines of every case contain 9 integers each: the costs of the 36 characters in the order given above. Then follows the number of queries on a line by itself. Every query appears on a line by itself and is formed by a number between 0 and 2000000000 in decimal format.

#### Output

For every case in the input, print one line "Case X:", without the quotes, where X is the case number starting from 1.

For every query within a case print one line "Cheapest base(s) for number Y: " followed by the cheapest base(s) in increasing order, separated by one space. Y is the query in decimal format. Print a blank line between cases.

# Sample Input

|  |
| --- |
| 2  10 8 12 13 15 13 13 16 9  11 18 24 21 23 23 23 13 15  17 33 21 23 27 26 27 19 4  22 18 30 30 24 16 26 21 21  5  98329921  12345  800348  14  873645  1 1 1 1 1 1 1 1 1  1 1 1 1 1 1 1 1 1  1 1 1 1 1 1 1 1 1  1 1 1 1 1 1 1 1 1  4  0  1  10  100 |

# 

# Output for Sample Input

|  |
| --- |
| Case 1:  Cheapest base(s) for number 98329921: 24  Cheapest base(s) for number 12345: 13 31  Cheapest base(s) for number 800348: 31  Cheapest base(s) for number 14: 13  Cheapest base(s) for number 873645: 22    Case 2:  Cheapest base(s) for number 0: 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36  Cheapest base(s) for number 1: 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36  Cheapest base(s) for number 10: 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36  Cheapest base(s) for number 100: 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 |

The numbers in the second sample output are actually all in one line just like the first sample output. Due to lack of horizontal space they are shown broken in two lines.

**Problem setter: Joachim Wulff, EPS**

**Special Thanks: Derek Kisman, EPS**

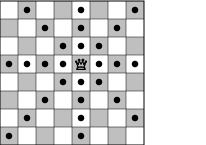
### 11494 - Queen

Time limit: 1.000 seconds

|  |
| --- |
| **Queen** |

## The Problem

The game of Chess has several pieces with curious movements. One of them is the*Queen*, which can move any number of squares in any direction: in the same line, in the same column or in any of the diagonals, as illustrated by the figure below (black dots represent positions the queen may reach in one move):



The great Chess Master Kary Gasparov invented a new type of chess problem: given the position of a queen in an empty standard chess board (that is, an 8 x 8 board) how many moves are needed so that she reaches another given square in the board?

Kary found the solution for some of those problems, but is having a difficult time to solve some others, and therefore he has asked that you write a program to solve this type of problem.

## The Input

The input contains several test cases. The only line of each test case contains four integers *X1*, *Y1*, *X2* and *Y2* (*1 ≤ X1, Y1, X2, Y2 ≤ 8*). The queen starts in the square with coordinates *(X1, Y1)*, and must finish at the square with coordinates *(X2, Y2)*. In the chessboard, columns are numbered from 1 to 8, from left ro right; lines are also numbered from 1 to 8, from top to bottom. The coordinates of a square in line *X* and column *Y* are (*X, Y*).

The end of input is indicated by a line containing four zeros, separated by spaces.

## The Output

For each test case in the input your program must print a single line, containing an integer, indicating the smallest number of moves needed for the queen to reach the new position.

## Sample Input

4 4 6 2

3 5 3 5

5 5 4 3

0 0 0 0

## Sample Output

1

0

2

### 12247 - Jollo

Time limit: 3.000 seconds

Jollo is a simple card game which the children from Logonia love to play. It is played between two players with a normal deck of 52 cards. In the game, cards are ordered according to their rank and suit, forming a sequence of 52 distinct values.

The game is composed of three rounds, played in a best-of-three series (a player must win two rounds to win the game). At the beginning of the game the deck is shuffled and each player is given a hand of three cards. In each round the players show one card to each other and the player with the highest card wins the round. The cards shown in a round are discarded (i.e., they cannot be shown again).

The King's son loves to play the game. But he is not very smart, losing frequently to his little sister. And when he loses, he cries so loud no one can stand it. The servant who deals the cards to the Prince and his sister is afraid he will be sent to prison if the Prince continues to lose. The servant is allowed to see every card he deals, and after dealing five cards (three to the Princess and two to the Prince) he wants to know which is the lowest card he should deal to the Prince so that there is no chance he will lose the game, no matter how badly he plays.

## Input

Each test case is given in a single line that contains five distinct integers *A*, *B*, *C*, *X* and*Y*, describing the cards dealt to the players. The first three cards are given to the Princess (1$ \le$*A*, *B*, *C*$ \le$52) and the last two cards are given to the Prince (1$ \le$*X*, *Y*$ \le$52). The last test case is followed by a line containing five zeros.

## Output

For each test case output a single line. If there exists a card that will make the Prince win the game no matter how badly he plays, you must print the lowest such a card. Otherwise, print -1.

## Sample Input

28 51 29 50 52

50 26 19 10 27

10 20 30 24 26

46 48 49 47 50

0 0 0 0 0

## Sample Output

30

-1

21

51

### 11389 - The Bus Driver Problem

Time limit: 1.000 seconds

|  |  |
| --- | --- |
| **I I U C   O N L I N E   C O N T E S T   2 0 0 8** | |
| **Problem E: The Bus Driver Problem** | |
| **Input: standard input**  **Output: standard output** | |
|  | |
| In a city there are **n** bus drivers. Also there are **n** morning bus routes & **n**afternoon bus routes with various lengths. Each driver is assigned one morning route & one evening route. For any driver, if his total route length for a day exceeds **d**, he has to be paid overtime for every hour after the first **d**hours at a flat **r**taka / hour. Your task is to assign one morning route & one evening route to each bus driver so that the total overtime amount that the authority has to pay is minimized. | |
| **Input** | |
| The first line of each test case has three integers **n**, **d**and **r**, as described above. In the second line, there are **n**space separated integers which are the lengths of the morning routes given in meters. Similarly the third line has **n**space separated integers denoting the evening route lengths. The lengths are positive integers less than or equal to 10000. The end of input is denoted by a case with three 0 s. | |
| **Output** | |
| For each test case, print the minimum possible overtime amount that the authority must pay. | |
| **Constraints** | |
| **-           1 ≤ n ≤ 100**  **-           1 ≤ d ≤ 10000**  **-           1 ≤ r ≤ 5** | |
| **Sample Input** | **Output for Sample Input** |
| **2 20 5**  **10 15**  **10 15**  **2 20 5**  **10 10**  **10 10**  **0 0 0** | **50**  **0** |
|  |  |
| **Problem setter: Mohammad Mahmudur Rahman** | |

### 11201 - The problem of the crazy linguist

Time limit: 3.000 seconds

|  |
| --- |
| **The Problem of the Crazy Linguist** |

## Background

Please, help the crazy linguist! He is trapped in his madness. He has developed a ``Spanish Beauty Criterion'' for words. It is defined as follows. Given a word w

$\displaystyle w = x_{1}x_{2}x_{3}\ldots x_{n}
$

(where n is the length of the word), he is interested in words which are formed by letters following pattern:

$\displaystyle x_i \in
\left\lbrace \begin{array}{ll}
\{\mathrm{bcdfghjklmnpqrst...
...{aeiou}\} & \mathrm{if} i \mathrm{is} \mathrm{even} \\
\end{array} \right.
$

and, also, in his madness, he won't allow a word that actually has i, j, and k, so that $ x_i = x_j = x_k$ (that is, any letter can appear in the word at most two times).

Then, the ``Spanish Beauty Criterion'' (SBC) is defined as:

$\displaystyle \mathrm{SBC}(w) = \sum_{i \in 1\ldots n} i*P(x_i)
$

where P is defined as the probability of appearance of a letter in Spanish, defined by the following table:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| a | b | c | d | e | f | g | h | i | j | k | l | m |
| 12.53 | 1.42 | 4.68 | 5.86 | 13.68 | 0.69 | 1.01 | 0.70 | 6.25 | 0.44 | 0.00 | 4.97 | 3.15 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| n | o | p | q | r | s | t | u | v | w | x | y | z |
| 6.71 | 8.68 | 2.51 | 0.88 | 6.87 | 7.98 | 4.63 | 3.93 | 0.90 | 0.02 | 0.22 | 0.90 | 0.52 |

So, given a word w, of size n, our poor linguist wants to know **if this word is above or below the average of the SBC of all the words of size n that can be constructed following the above pattern and start with the same letter than w**.

## The Input

The input will have a first line with a number, N, the number of samples that will be entered to know if they are above or below the average. Following this first line, Nlines with a word in each one (of at most seven characters each). All the input words follow the given pattern.

## The Output

The output will have N lines, each line corresponding to one input word in order, showing just ``above or equal'' or ``below'', depending on the value of the SBC of that word relative to the average of those of the same size.

## Sample Input

5

bubu

terabit

hacer

qed

loco

## Sample Output

below

above or equal

above or equal

above or equal

above or equal

### 10718 - Bit Mask

Time limit: 3.000 seconds

|  |  |
| --- | --- |
| **Problem A** | **Bit Mask** |
| **Time Limit** | **1 Second** |

In bit-wise expression, mask is a common term. You can get a certain bit-pattern using mask. For example, if you want to make first 4 bits of a 32-bit number zero, you can use 0xFFFFFFF0 as mask and perform a bit-wise AND operation. Here you have to find such a bit-mask.

Consider you are given a 32-bit unsigned integer **N**. You have to find a mask **M** such that **L ≤ M ≤ U**and **N OR M**is maximum. For example, if **N**is 100 and **L** = 50, **U** = 60 then **M** will be 59 and **N OR M**will be 127 which is maximum**.**If several value of**M** satisfies the same criteria then you have to print the minimum value of **M**.

**Input**Each input starts with 3 unsigned integers **N**, **L**, **U** where **L ≤ U**. Input is terminated by EOF.

**Output**For each input, print in a line the minimum value of **M**, which makes **N OR M**maximum.

Look, a brute force solution may not end within the time limit.

|  |  |
| --- | --- |
| **Sample Input** | **Output for Sample Input** |
| 100 50 60 100 50 50 100 0 100 1 0 100 15 1 15 | 59 50 27 100 1 |

**Problem setter: Md. Kamruzzaman  
Member of Elite Problemsetters' Panel**

### 10503 - The dominoes solitaire

Time limit: 3.000 seconds

|  |
| --- |
| **Problem D The dominoes solitaire** |

A man used to play dominoes with some friends in a small town. Some families have left the town and at present the only residents in the town are the man and his wife. This man would like to continue playing dominoes, but his wife does not like plaing. He has invented a game to play alone. Two pieces are picked out and are put at the two extremes of a row with a number (n) of spaces. After that, other pieces (m) are picked out to fill the spaces. The number of pieces is greater than or equal to the number of spaces (m>=n), and the number of pieces is less than or equal to 14 (m<=14). The spaces are filled by putting one piece in each space, accoding to the rules of dominoes: the number of adjacent dots on two different dominoes must coincide. Pieces with repeated values are placed in the same way as the other pieces, and not at right angels.

The problem consists in the design of a program which, given a number of spaces (n), a number of pieces (m), the two initial pieces (i1,i2) and (d1,d2), and the m pieces (p1,q1), (p2,q2), ..., (pm,qm), decides if it is possible to fill the n spaces between the two initial pieces using the m pieces and with the rules of dominoes. For example, with n=3, m=4, initial pieces (0,1) and (3,4), and pieces (2,1), (5,6), (2,2) and (3,2), the answer is YES, because there is a solution: (0,1), (1,2), (2,2), (2,3), (3,4). With n=2, m=4, pieces in the extremes (0,1) and (3,4), and (1,4), (4,4), (3,2) and (5,6), the answer is NO.

## The Input

The input will consist of a series of problems, with each problem described in a series of lines: in the first line the number of spaces (n) is indicated, in the second line the number of pieces (m) used to fill the spaces, in the next line the piece to be placed on the left, with the two values in the piece separated by a space and in the same way that the numbers appear in the row, in the following lines the piece to be placed on the right, with the two values in the piece separated by a space and in the same way that the numbers appear in the row, and the other m pieces appear in  consecutive lines, one in each line, with the two values separated by a space. Different problems appear in the input successively without separation, and the input finishes when 0 appears as the number of spaces.

## The Output

For each problem in the input a line is written, with YES if the problem has a solution, and NO if it has no solution.

## Sample Input

3

4

0 1

3 4

2 1

5 6

2 2

3 2

2

4

0 1

3 4

1 4

4 4

3 2

5 6

0

## Sample Output

YES

NO

### 11108 - Tautology

Time limit: 3.000 seconds

## http://uva.onlinejudge.org/external/111/p11108.jpgProblem D: Tautology

WFF 'N PROOF is a logic game played with dice. Each die has six faces representing some subset of the possible symbols K, A, N, C, E, p, q, r, s, t. A Well-formed formula (WFF) is any string of these symbols obeying the following rules:

* p, q, r, s, and t are WFFs
* if *w* is a WFF, N*w* is a WFF
* if *w* and *x* are WFFs, K*wx*, A*wx*, C*wx*, and E*wx* are WFFs.

The meaning of a WFF is defined as follows:

* p, q, r, s, and t are logical variables that may take on the value 0 (false) or 1 (true).
* K, A, N, C, E mean *and, or, not, implies,* and *equals* as defined in the truth table below.

|  |
| --- |
| **Definitions of K, A, N, C, and E** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ***w*  *x*** | **K*wx*** | **A*wx*** | **N*w*** | **C*wx*** | **E*wx*** |
| **1  1** | **1** | **1** | **0** | **1** | **1** |
| **1  0** | **0** | **1** | **0** | **0** | **0** |
| **0  1** | **0** | **1** | **1** | **1** | **0** |
| **0  0** | **0** | **0** | **1** | **1** | **1** |

A *tautology* is a WFF that has value 1 (true) regardless of the values of its variables. For example, *ApNp* is a tautology because it is true regardless of the value of *p*. On the other hand, *ApNq* is not, because it has the value 0 for *p=0, q=1*.

You must determine whether or not a WFF is a tautology.

Input consists of several test cases. Each test case is a single line containing a WFF with no more than 100 symbols. A line containing 0 follows the last case. For each test case, output a line containing *tautology* or *not* as appropriate.

### Sample Input

ApNp

ApNq

0

### Possible Output for Sample Input

tautology

not

*Gordon V. Cormack*