Easy

Sentence casing

Many people occasionally mis-time holding the shift key while typing, thus causing the capitalization at the beginning of their sentences to be incorrect. Thankfully, Microsoft Word engineers created a feature many years ago to automatically fix this for users. To pay homage to this feature, which is now often taken for granted, this problem focuses on a simple version of the feature: where the only time a capital should be used is at the beginning of a sentence (we'll ignore things like proper nouns and abbreviations).

Input definition

The input will be a series of lines, each of which contains one or more sentences. The beginning of a sentence is the first letter of the line or the first letter following a previous sentence. The end of a sentence is marked by a period, question mark, or exclamation point. The input will be terminated with an empty line.

Output definition

The same format as the input, only the first letter of each sentence must be capitalized and no other letters may be capitalized.

Example input

```
This is a well-capitalized sentence. this is not.
Neither is a terribly Interesting sentence.
this is a second line. Why are these sample sentences
so blah?
```

Example output

```
This is a well-capitalized sentence. This is not.
Neither is a terribly interesting sentence.
This is a second line. Why are these sample sentences
so blah?
```

Sorting using directives

If you're a C# user in Visual Studio, you've probably noticed how quickly the using directives in a file can seem to get out of control. For those who may be unfamiliar with C#, more info on the directive can be found here: using

Directive. For most users, using directives are added piecemeal, as needed, and generally have no real organization. To help manage this mess, Visual Studio has a set of features to organize usings. This problem focuses on a part of that featureset: "remove and sort". This feature both sorts the using directives alphabetically and removes any using directives that are not being used. To simplify this particular problem, the input will include a comment for each using directive indicating whether or not it is being used.

Input definition

The input will contain multiple sets of using directives, with each set terminated by the line "//----". Each using directive will be in the form "using <namespace>; //<comment>" where <namespace> will be a C# namespace and <comment> will be either "Used" or "Unused" to indicate if the using directive should be included.

Output definition

The output should contain the sorted and filtered sets of using directives, with each set terminated by the line "//----".

Example input

```
using System; //Used
using System.Text; //Unused
//----
using System.Collections.Generic; //Used
using System; //Unused
using System.Text; //Unused
using System.Threading.Tasks; //Unused
using System.Linq; //Used
//----
```

Example output

```
using System; //Used
//----
using System.Collections.Generic; //Used
using System.Linq; //Used
//----
```

/ko

Rock paper scissors

The World Rock Paper Scissors Society has decided to create a new tournament for robotic players only. The robots must enter pre-programmed with the series of moves that they will make.

The rules are simple: each opponent shows the symbol of rock, paper, or

scissors at the same time. Rock beats scissors. Scissors beats paper. Paper beats rock. The losing player takes one robotic step backward and the winner takes one step forward. This repeats until one player has traveled a net five steps forward and is declared the winner.
You will be given the programmed series of moves for each robot. These moves will be used in order (when you get to the end of the series, start again at the beginning). Your task is to determine the winner. Note that a match may have at a maximum 100,000 plays of rock, paper, scissors. If that number has been reached and no winner found, a draw is declared.

Input definition

Input consists of two lines. The first line is for robot player one. The second is for robot player 2.
The lines start with the name of the robot player. This is followed by a : character; robot names are not allowed to have : characters in them. This is then followed by the series of moves the robot is to make in single characters. R stands for rock. P stands for paper. S stands for scissors.

Output definition

The output consists of two items, the number of rounds played in the match (remember that 100,000 is the maximum). This number should be represented as an integer with only numeric characters.
The second line should contain either the name of the winning robot or the text ":DRAW:".

Example input

```
Johnny 5:RPSRPSR
D.A.R.Y.L.:PPSSPPSSRRPPSS
```

Example output

```
21
Johnny 5
```

Medium

Cyber security

With all of the recent high profile cyber attacks on large companies, your company has decided to take proactive action to strengthen security measures. You have been tasked with devising a system that will limit access to sensitive documents, such as employee salaries, expense reports, and so on. A resource can be limited to only certain employees or to a group that contains

employees. Groups can contain other groups and permissions should apply to all members; in particular, permission to a resource should be granted to any member of a group that has permission to the resource, regardless of whether the permission is direct or via nested group membership.

Input definition

The input will begin with two numbers: **n** (1 < n <= 1000) and **m** (1 < m < 100), where **n** is the number of groups in the system and **m** is the number of access checks to be made.

The first **n** lines after the first line will be the group definitions; they will be of the following format:

```
<name of group>:<name of member>;<name of second member>;<third & etc>
```

The following **m** lines will be the access checks and will be of the following format:

```
<name of user requesting access>;<group the member needs to be a part of>
```

Consider these group definitions:

```
Electronic Artists:Daft Punk;House DJs;Trance Kings
House DJs:Don Diablo;Tritonal;Hardwell
Trance Kings:Above & Beyond;Super8 & Tab
```

`Daft Punk` is a direct member of the group `Electronic Artists` and `Hardwell` is an indirect member. Any resource that permitted access to members of `Electronic Artists` would permit access to those two users (along with the group's other direct and indirect members). A resource that restricted access only to members of `House DJs`, however, would not include `Daft Punk` or `Above & Beyond`.

Consider these access checks:

```
Tritonal:Electronic Artists
Above & Beyond:House DJs
Super8 & Tab:Trance Kings
```

The first access check would be granted, although the membership is indirect. The second access check would be denied because `Above & Beyond` is not a direct or indirect member of `House DJs`. The third access check would be granted with a direct membership.

The system dictates that group memberships are not allowed to form a cycle.

Output definition

The output will contain **m** lines. The output is one of three possible values: `Yes (direct)`, `Yes (indirect)` or `No`. The value to use depends on the result of the access check. If the access check was granted because the user was a direct member of the group, the output value should be `Yes (direct)`. If the access check was granted because the user was an indirect

member, the value should be `Yes (indirect)`. If the access check was denied, the value should be `No`.
Continuing with the previous example, the output would be:

```
Yes (indirect)
No
Yes (direct)
```

Example input

```
6 6
Seahawks:Players;Coaching Staff;Owners;Paul Allen
Players:Russell Wilson;Marshawn Lynch (BEAST)
Coaching Staff:Pete Carroll
Owners:Paul Allen
Backup dancers:Left Shark;Right Shark
Halftime Show:Backup dancers;Katy Perry
Tom Brady;Seahawks
Left Shark;Halftime Show
Marshawn Lynch (BEAST);Seahawks
Pete Carroll;Coaching Staff
Katy Perry;Seahawks
Paul Allen;Seahawks
```

Example output

```
No
Yes (indirect)
Yes (indirect)
Yes (direct)
No
Yes (direct)
```

Palindromic anagrams

A palindrome is a word, phrase, number, or other sequence of characters which reads the same backward or forward. An anagram is the result of rearranging the letters of a word or phrase to produce a new word or phrase, using all of the original letters exactly once.
This problem is a combination of both. You will be given a string and your program needs to check whether or not any of its anagrams contain a palindrome. If none of the anagrams is a palindrome, then you need to find the minimum number of characters to eliminate from the string in order for at least one of its anagrams to be a palindrome. In addition, your program needs to figure out the number of anagrams that are palindromes.
Consider this example: You are given the string "aabd". None of the anagrams

of this string are palindrome. If you remove the letter "d", however, then "aba" is an anagram of the string that is a palindrome; if you remove the letter "b", then "ada" is an anagram that is a palindrome. In either case, there is only one anagram that is a palindrome.

Input definition

The input file will contain multiple lines. Each line will have a string of lowercase alphabetic characters ('a'-'z'). The length of the string will vary from 1 to 20.

Output definition

The output file should contain exactly the same number of lines as the input. Each line of the output should be of the following format: X,Y where X is the minimum number of characters you need to remove from the original string for any of its anagrams to be a palindrome and Y is the total number of anagrams that are palindromic after you remove X characters. Permutations of the same letter within a palindrome do not count; in particular, 'aaa' is 1 palindrome, not 6.

Example input

```
aba
acad
agabgb
agqagdt
```

Example output

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

Hard

Minecraft tunneling

You are to assume the role of Steve in the Minecraft world. He is looking to dig a path from a given point to his destination, using a pickaxe of limited durability.

Since the durability of the pickaxe is limited, he is looking to take the least number of swings at the rocks/minerals/dirt to arrive at his destination.

Your task is to calculate the least number of swings that need to be made to create a tunnel through the ground (a cube in this scenario) from one point to another. Moves must be in only one dimension at a time, i.e. the path cannot

contain diagonal steps.

Input definition

Input Format

The first line of the input contains the cube size N.
The second line of the input contains the hardness(H) (number of swings required to break) of each block of the cube (N^3 total).
The third line contains the target start block, which consists of 3 numbers representing x,y,z coordinates, in that order.
The forth line contains the target end block, which consists of 3 numbers representing x,y,z coordinates, in that order.

Constraints

2 <= N <= 25
0 <= H <= 10
start block != end block

Detailed explanation of line 2 format

Consider the following line 2 input. 3 4 7 3 4 1 0 2
The corresponding cube should be laid out like this.

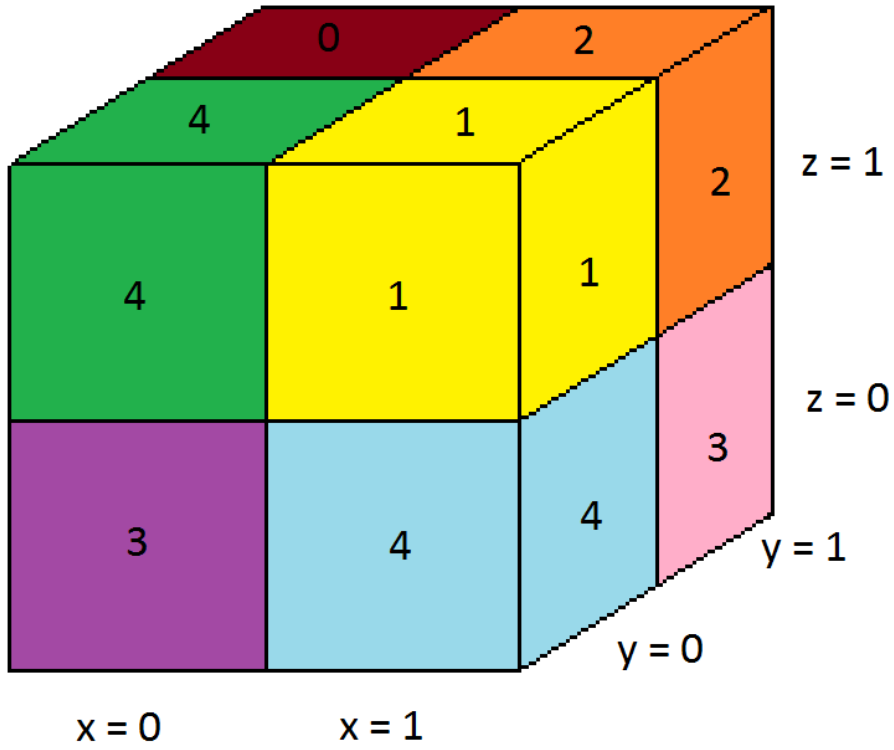
Z Position 0

```
y
1 |7 3
0 |3 4
   ------x
   0 1
```

Z Position 1

```
y
1 |0 2
0 |4 1
  ------x
   0 1
```

If a target start block of 0 1 1 was provided, we could see the hardness of this block would be 0.



Output definition

Output a single integer, representing the smallest number of pickaxe swings required to create a path from the start block to the end block.
Note: Since hardness values are provided for the starting and ending coordinates, those blocks must be broken and those swings are to included in the total count.

Example input

```
2
3 4 7 3 4 1 0 2
0 1 0
1 0 1
```

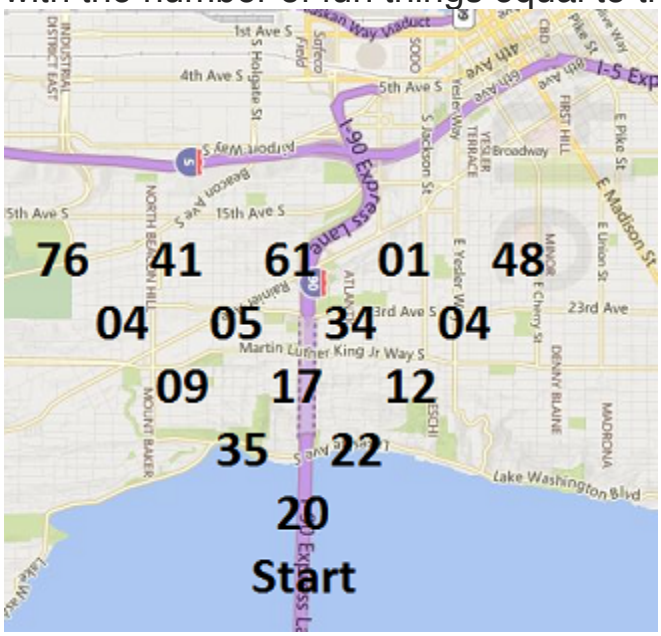
Example output

```
10
```

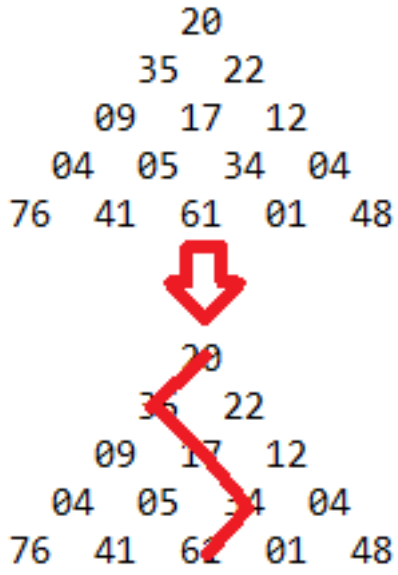
The curious intern

Samantha the Intern is leaving Microsoft soon, and wants to visit the fun places around the Puget Sound before she goes. She doesn't have much time so she decides to rank each spot on a fun scale from 1 to 99, inclusive.
As she moves farther west into Seattle, there are more fun things to do,

with the number of fun things equal to the length of her trip.



Samantha doesn't like travelling far between fun spots, so after visiting a spot she can only move to an adjacent spot.



Every time she moves, she wants to go deeper into the city. Can you help Samantha by finding the most possible fun she can have by the end of her day? There are 2 to the power of L possible paths, so Samantha hopes you don't try to brute force the solution.

Limits:

L layers, where 45 <= L <= 55. Each layer will consist of fun scores F, where F is from [0, 100).

Input:

L, which is the number of layers to follow. Followed by L lines of space-delimited fun scores, where the number of scores is equal to the depth (1

score at a depth of 1, 2 scores at a depth of 2, etc.).

Output:

The number corresponding to the sum of scores in the greatest possible path.

Input definition

The input will be a number, followed by that many lines of space-delimited numbers. Each subsequent line will have one more number/score than the line before. Extra padding included for readability should be ignored.

Output definition

A single number corresponding to the greatest path through the graph. When moving down through the graph, only adjacent numbers can be considered.

Example input

```
5
         20
       35  22
     09  17  12
  04  05  34  04
76  41  61  01  48
```

Example output

```
167
```