APP
Association of
Professional
Programmers

Expert Programming Problems

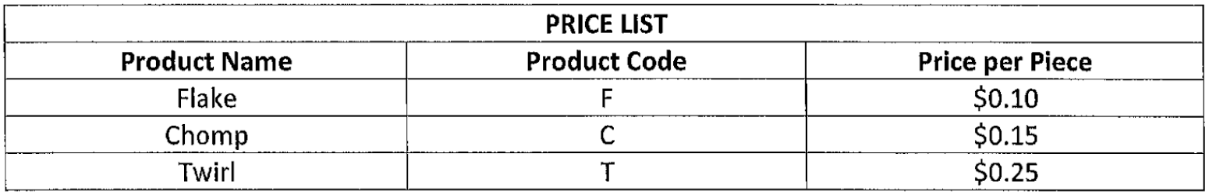
Brought to you by APP

Instructions

All programs will accept input through standard input and output to standard out only. Output must match the desired format exactly. This means there should be no debug statements that are printed and that the format must match exactly as specified. Additionally, input will be given to you exactly as expected. If the problem says input will be an integer, you do not have to check to see if a character is entered. You can solve any problem using any combination of the languages Java, C#, and/or C++. You may work in a team of up to 3 people or compete by yourself. If you need clarification on a problem you can submit a request for clarification directly through PC^2. The team who solves the most problems is the winner. In the case of a tie, the team who solved the last problem first will be the winner.

Candy Store

Your cravings for chocolate are insatiable. You finished off the candy that you found in the castle and have made your way to the local chocolate shop. The owner of the chocolate shop is tired of waiting for customers to figure out how much candy that they can buy with the money that they have. To save time, the owner has devised a system where customers enter the amount of money that they have into a secret machine and the machine prints out all possible candy orders given that amount of money. The program always ensures that the customers spend as much as they can for each possible order, therefore the amount of change left from each order should be less than the cheapest candy bar.



PRICE LIST		
Product Name	**Product Code**	**Price per Piece**
Flake	F	$0.10
Chomp	C	$0.15
Twirl	T	$0.25

Input:

Will be a monetary value with a dollar sign and two decimal points (Example: $0.75) and the program will display the specified output. The user can then enter another monetary value to run the program again. The program will stop when the user enters ($0.00).

Output:

The program will output a list of possible candy orders given the specified amount of money without repeats. Each line will represent one possible order. On each line, the program display a product code followed by an integer indicating the quantity of candy. The number of twirl bars is shown first, followed by the number of Chomp bars, and finally by the number of Flake bars.

Each candy will be separated by a comma in the output. (See example below.) The possible orders are sorted in descending order from most expensive bar to least expensive bar. If user does not have enough money to buy any candy, then the program should output “NO CANDY FOR YOU”.

Sample Input

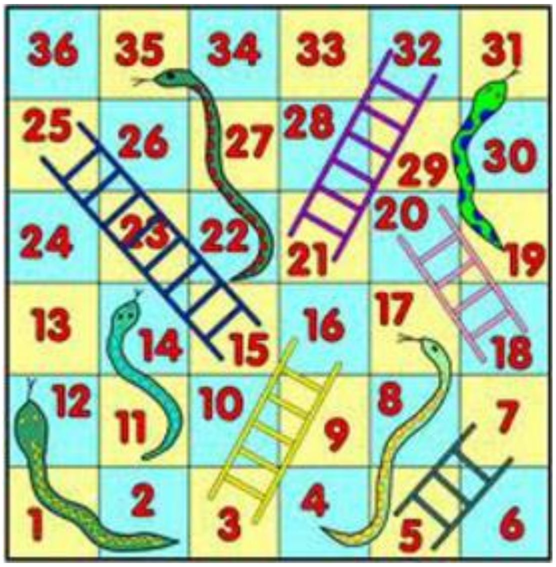
```
$0.40
$0.05
$0.00
```

Sample Output

```
T1,C1,F0
T1,C0,F1
T0,C2,F1
T0,C1,F2
T0,C0,F4
NO CANDY FOR YOU
```

Chutes and Ladders

Two players are playing Chutes and Ladders using the following game board.



Given the list of player's moves, you are to find the winner. Players may play multiple rounds. The winner is the player that reaches the last block first or happens to be ahead the other player at the end of the round. The pieces start on the square marked as 1.

Input

First line contains an integer N representing the number of rounds that the game is played. For each round there will be one line representing the moves in alternative order starting with player one's move. Moves are separated by white spaces.

Output

For each output the round number and the player name in the following format

Round <round number>: Player <one|two>

Sample input

```
2
4 3 1 6 6 3 2 5 3 6 5 4 3 6
3 5 3 6 3 2 6 5 6 3 1 3
```

Sample output

```
Round 1: Player two
Round 2: Player one
```

Darkest Spot

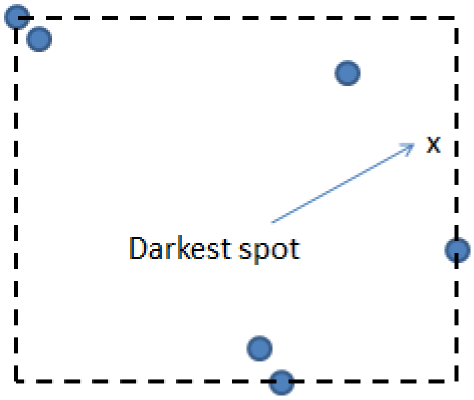
The light intensity in a given location is based on the light sources visible from that location.

The light received from a light source is calculated according to the formula

$$L = C / D^2$$

Where C is the luminosity of the source and D is the distance from the light source.

Assume that a collection of light sources and their locations in a planar grid is given. You need to find the "darkest" location on the grid within the region bounded by the light source coordinates. Assume integral coordinates only.



Input:

The first line of input consists of a single integer N, representing the number of light sources.

The remaining N lines contain three integers each representing the x-coordinate, y-coordinate, and luminosity of a light source respectively. Integral coordinates are used.

Output

Output should the coordinates of the darkest location in the grid in the form of (x, y)

Sample input

```
6
-180 90 200
-160 80 100
100 65 400
239 8 200
65 -72 100
42 -110 200
```

Sample output

```
Darkest location: (239, 90)
```

Pirate Pandemic

The dreaded pirate Yo-Ho-Ho-As has set to the seas once again to raid nearby villages of their treasure. As his first mate, it is your job to determine the best place to strike. All you have to go by is a map of all the towns Yo-Ho has pillaged in the past. Luckily for you he likes to keep tabs on his treasure and has recorded how much he obtained from each village.

Given a 2D array of random size determine the X and Y coordinates of the beast location to strike (the wealthiest area). For every unit of treasure looted for a town its wealth radius increases by 1.

Input

Input will be two comma integers representing the length and width of the array, then a blank line followed by a 2D array of the given size.

Output

Output must be the x and y coordinates given in the form x, y of the wealthiest spot on the map. If more than one location is found output must be line delimited and in numerical order starting with x.

Sample Input

```
10,10
1#########
#########1
###1######
#######3##
##########
####2#####
########2#
1#########
####2#####
```

```
###1######
```

Sample Output

```
3,2
5,7
```

Jill’s Bike

Jill Bates hates climbing hills. Jill rides a bicycle everywhere she goes, but she always wants to go the easiest and shortest way possible. The good news is that she lives in Greenhills, which has all its roads laid out in a strictly rectangular grid - east-west roads are *streets*; north-south roads are *avenues* and the distance between any two adjacent grid points is the same. The bad news is that Greenhills is very hilly and has many one-way roads.

In choosing a route between where she starts and where she ends, Jill has three rules:

1. Avoid any travel of more than 10 meters between adjacent grid points.
2. Never go the wrong way on a one-way road.
3. Always travel the shortest possible route.

Your program should help Jill find an acceptable route.

Input

The input file contains less than 50 data sets in the following form:

- The first line of each set contains two integers, separated by one or more spaces. The first integer *n* represents the number of streets, and the second integer *m* represents the number of avenues, $1 \le n \le 20$, $1 \le m \le 20$.
- The next *n* lines contain the altitudes of grid points. Each line represents a street and contains a sequence of *m* integers separated by one or more spaces. These integers represent the altitude in meters of the grid points along that street. Even if a particular street and avenue have no intersection, the altitude is still given for that grid point.
- One or more lines follow that define the one-way roads. Each road is represented by two pairs of integers, separated by one or more spaces, in the form:
- *street avenue street avenue*

- The first street and avenue define the starting point of the road and the second pair define the ending point. Since Greenhills is a strict grid, if the two points are not adjacent in the grid, the road passes through all the intervening grid points. For example,
- 5 7 5 10
- represents roads 5-7 to 5-8, 5-8 to 5-9, and 5-9 to 5-10. Road definitions are terminated by a line containing four zeroes in the above format.
- Finally, one or more lines will follow that contain pairs of grid points between which Jill wants to find an optimal path, in the form:
- *street avenue street avenue*
- As before, the integer pairs are separated by one or more spaces. The end of the input set is defined by a line containing four zeroes, formatted as before.

You may assume that all street and avenue numbers are within the bounds defined by the first line of set, and that all road definitions are strictly north-south or east-west. The end of input file is defined by EOF.

Output

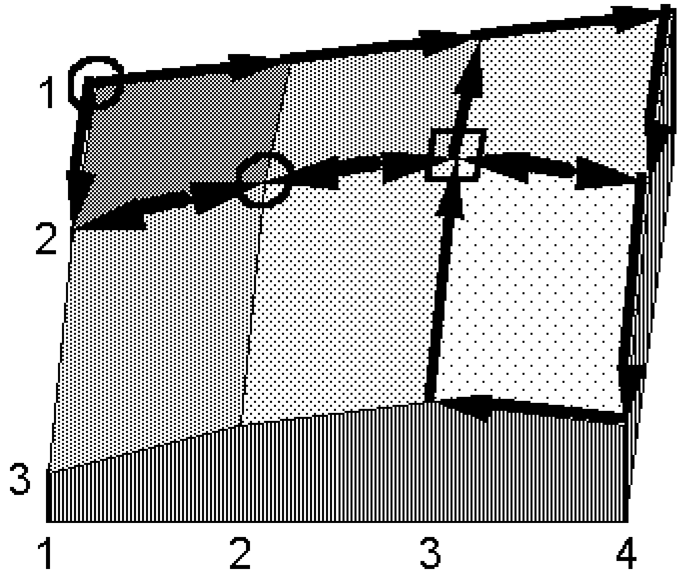
For each path query in the input file, output a sequence of grid points, from the starting grid point to the ending grid point, which meets Jill's three rules. Output grid points as \`*street-avenue*' separated by the word \`to'. If there is more than one path that meets Jill's criteria, any such path will be acceptable. If no route satisfies all the criteria, or if the starting and ending grid points are the same, output an appropriate message to that effect. Output a blank line between each output for query.

Sample Input

```
3 4
10  15  20  25
19  30  35  30
10  19  26  20
```

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

Diagram of the Sample Input



Sample Output

```
1-1 to 1-2 to 1-3 to 1-4 to 2-4 to 2-3 to 2-2

To get from 2-3 to 2-3, stay put!

There is no acceptable route from 2-2 to 1-1.
```

Too Many Rectangles

In this problem, the input will be one or more rectangles. The 1st line of input is the number of rectangles. For each rectangle, there are 4 lines of input. The 1st line is the x-coordinate of the lower left corner. The 2nd line is the y-coordinate of the lower left corner. The 3rd line is the horizontal size of the rectangle. The 4th is the vertical size. All values will be positive integers, however they may be entered in any order. The lines of the different rectangles intersect they may form new rectangles. The program must output the area of the smallest rectangle formed.

Sample input

```
3
2
1
5
8
3
2
1
9
1
7
12
3
```

Sample output

```
1
```

APP Executive Board

We need to form committees and we need a program to list all the committees we conform under some restrictions. We will tell you how many people are eligible for the committee and we will give you their names. We will tell you how big a committee we want. There are 4 possible restrictions. There are some people who must not be on the committee. There are some people who must be on the committee. There are some pairs of people who must not serve together on the committee. Finally, there are groups of people from which we must use at least one person. This information will be input using the form described below. The program prints out all possible committees that meet the restrictions. Each possible committee is on one line.

Example

Suppose we have 9 people: Al, Ben, Cary, Donna, Elmer, Gina, Lola, Ty, Xi. We want a committee of 5. We do not want Carrie and we do not want Lola. We want Gina. We do not want both Donna and Elmer. We do not want both Ty and Al. We need one or more of the Elmer, Xi, and Ben.

The answer would be:

```
Gina, Al, Ben, Donna, Xi
Gina, Al, Ben, Elmer, Xi
Gina, Ben, Donna, Ty, Xi
Gina, Ben, Elmer, Ty, Xi
```

It is possible that for some inputs, the answer is:

```
No committee is possible
```

Input form

First line is it number of people (say n)

next n lines are the names of the people, one name per line

next line is the number of people that must be on the committee

next line is the name of the committee

The preceding lines consist of the restrictions (one per line)

each restriction has the form code name_one name_two

the legal codes are N for end of restrictions, X for exclude, I for include, S for some, And C for one of. The rest of each line will have the number of names needed. The names will be separated by blank.

The will contain Y or N. If it is N, no more lines follow. If it is Y, we repeat from the line giving number of people on the committee. In other words, we are asking to form another committee. We do not change the list of people.

The restrictions should be interpreted as follows:

X name means that the person must not be on the committee

I name means this person must be on the committee

N means end of restrictions

C name1 name2 means that if one of these people is on the committee the other cannot be

S name1 name2 means at least one of these people must be on the committee

Sample input

```
9
Al
Ben
Cary
```

```
Donna
Elmer
Gina
Lola
Ty
Xi
5
planning
X Carry
I Gina
C Donna Elmer
C Ty Al
S Elmer Xi Ben
N
Y
6
hospitality
C Ty Al
C Gina Lola
C Cary Elmer
C Ben Elmer
I Ben
I Xi
I Donna
N
```

Sample output

```
planning:
Gina, Al, Ben, Donna, Xi
Gina, Al, Ben, Elmer, Xi
Gina, Ben, Donna, Ty, Xi
Gina, Ben, Elmer, Ty, Xi
hospitality:
Ben, Xi, Donna, Al, Cary, Gina
Ben, Xi, Donna, Al, Carry, Lola
Ben, Xi, Donna, Carry, Gina, Ty
Ben, Xi, Donna, Cary, Lola, Ty
```