Root :: Programming Challenges (Skiena & Revilla) :: Chapter 12

Link: https://onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=40

| **Discentes** | **Problemas** |
|---|---|
| BRIHAN | Ant on a Chessboard |
| CARLOS | Robbery |
| DANIEL | (2/3/4)-D Sqr/Rects/Cubes/Boxes |
| JOÃO GABRIEL | Bee Maja |
| JOÃO HENRIQUE | Star |
| MATEUS | Dermuba Triangle |
| PEDRO | Airlines |
| VINICIUS | The Monocycle |

# 707 Robbery

Inspector Robstop is very angry. Last night, a bank has been robbed and the robber has not been caught. And this happened already for the third time this year, even though he did everything in his power to stop the robber: as quickly as possible, all roads leading out of the city were blocked, making it impossible for the robber to escape. Then, the inspector asked all the people in the city to watch out for the robber, but the only messages he got were of the form "We don't see him."

But this time, he has had enough! Inspector Robstop decides to analyze how the robber could have escaped. To do that, he asks you to write a program which takes all the information the inspector could get about the robber in order to find out where the robber has been at which time.

Coincidentally, the city in which the bank was robbed has a rectangular shape. The roads leaving the city are blocked for a certain period of time $t$, and during that time, several observations of the form "The robber isn't in the rectangle $R_i$ at time $t_i$" are reported. Assuming that the robber can move at most one unit per time step, your program must try to find the exact position of the robber at each time step.

## Input

The input file contains the description of several robberies. The first line of each description consists of three numbers $W, H, t$ ($1 \leq W, H, t \leq 100$) where $W$ is the width, $H$ the height of the city and $t$ is the time during which the city is locked.

The next contains a single integer $n$ ($0 \leq n \leq 100$), the number of messages the inspector received. The next $n$ lines (one for each of the messages) consist of five integers $t_i, L_i, T_i, R_i, B_i$ each. The integer $t_i$ is the time at which the observation has been made ($1 \leq t_i \leq t$), and $L_i, T_i, R_i, B_i$ are the left, top, right and bottom respectively of the (rectangular) area which has been observed. ($1 \leq L_i \leq R_i \leq W$, $1 \leq T_i \leq B_i \leq H$; the point $(1, 1)$ is the upper left hand corner, and $(W, H)$ is the lower right hand corner of the city.) The messages mean that the robber was not in the given rectangle at time $t_i$.

The input is terminated by a test case starting with $W = H = t = 0$. This case should not be processed.

## Output

For each robbery, first output the line '`Robbery #`$k$`:`', where $k$ is the number of the robbery. Then, there are three possibilities:

If it is impossible that the robber is still in the city considering the messages, output the line '`The robber has escaped.`'

In all other cases, assume that the robber really is in the city. Output one line of the form '`Time step` $\tau$`: The robber has been at` $x$`,`$y$`.`' for each time step, in which the exact location can be deduced. ($x$ and $y$ are the column resp. row of the robber in time step $\tau$.) Output these lines ordered by time $\tau$.

If nothing can be deduced, output the line '`Nothing known.`' and hope that the inspector will not get even more angry.

Output a blank line after each processed case.

## Sample Input

```
4 4 5
4
```

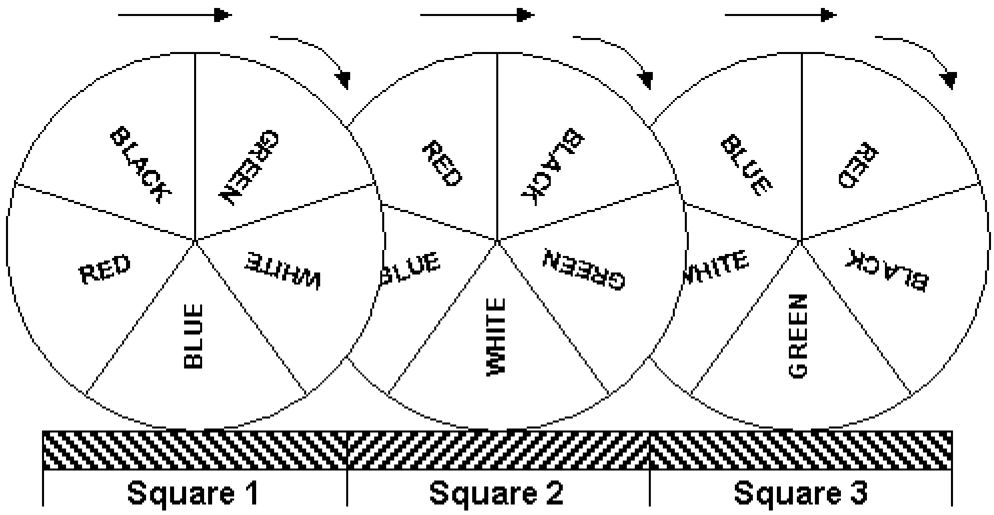
```
1 1 1 4 3
1 1 1 3 4
4 1 1 3 4
4 4 2 4 4
10 10 3
1
2 1 1 10 10
0 0 0
```

## Sample Output

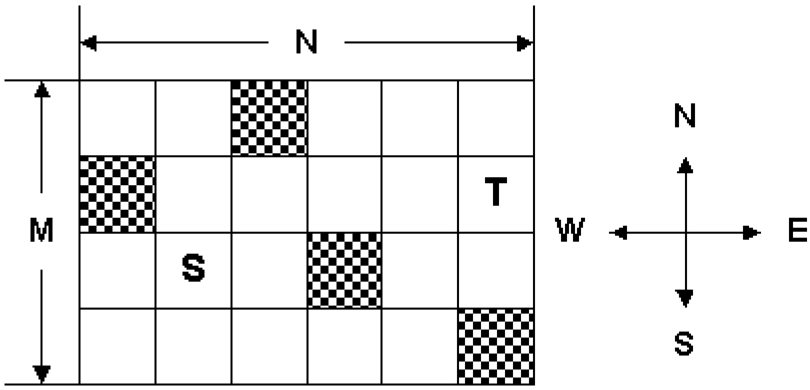
```
Robbery #1:
Time step 1: The robber has been at 4,4.
Time step 2: The robber has been at 4,3.
Time step 3: The robber has been at 4,2.
Time step 4: The robber has been at 4,1.

Robbery #2:
The robber has escaped.
```

# 10047 The Monocycle

A monocycle is a cycle that runs on one wheel and the one we will be considering is a bit more special. It has a solid wheel colored with five different colors as shown in the figure:




The colored segments make equal angles ($72^o$) at the center. A monocyclist rides this cycle on an $M \times N$ grid of square tiles. The tiles have such size that moving forward from the center of one tile to that of the next one makes the wheel rotate exactly $72^o$ around its own center. The effect is shown in the above figure. When the wheel is at the center of square 1, the midpoint of the periphery of its blue segment is in touch with the ground. But when the wheel moves forward to the center of the next square (square 2) the midpoint of its white segment touches the ground.




Some of the squares of the grid are blocked and hence the cyclist cannot move to them. The cyclist starts from some square and tries to move to a target square in minimum amount of time. From any square either he moves forward to the next square or he remains in the same square but turns $90^o$ left or right. Each of these actions requires exactly 1 second to execute. He always starts his ride facing north and with the midpoint of the green segment of his wheel touching the ground. In the target square, too, the green segment must be touching the ground but he does not care about the direction he will be facing.

Before he starts his ride, please help him find out whether the destination is reachable and if so the minimum amount of time he will require to reach it.

## Input

The input may contain multiple test cases.

The first line of each test case contains two integers $M$ and $N$ ($1 \leq M, N \leq 25$) giving the dimensions of the grid. Then follows the description of the grid in $M$ lines of $N$ characters each. The character '`#`' will indicate a blocked square, all other squares are free. The starting location of the cyclist is marked by '`S`' and the target is marked by '`T`'.

The input terminates with two zeros for $M$ and $N$.

## Output

For each test case in the input first print the test case number on a separate line as shown in the sample output. If the target location can be reached by the cyclist print the minimum amount of time (in seconds) required to reach it exactly in the format shown in the sample output, otherwise, print "`destination not reachable`".

Print a blank line between two successive test cases.

## Sample Input

```
1 3
S#T
10 10
#S.......#
#..#.##.##
#.##.##.##
.#....##.#
##.##..#.#
#..#.##...
#......##.
..##.##...
#.###...#.
#.....###T
0 0
```

## Sample Output

```
Case #1
destination not reachable

Case #2
minimum time = 49 sec
```

# 10075 Airlines

A leading airlines company has hired you to write a program that answers the following query: given a list of city locations (latitudes and longitudes) and a list of direct flights what is the minimum distance a passenger needs to fly to get from a given city to another?

To get from a city to another a passenger may either take a direct flight (if exists) or take a sequence of connecting flights (if there exists such a route).

Assume that if a passenger takes a direct flight from X to Y he never flies more than the geographical distance between X and Y. The geographical distance between two locations X and Y is the length of the geodetic line segment connecting X and Y. The geodetic line segment between two points on a sphere is the shortest connecting curve lying entirely in the surface of the sphere. Assume that the Earth is a perfect sphere with a radius of exactly 6378-km and the value of $\pi$ is approximately 3.141592653589793. Round the geographical distance between every pair of cities to the nearest integer.

## Input

The input may contain multiple test cases.

The first line of each test case contains three integers $N$ ($N \leq 100$), $M$ ($M \leq 300$) and $Q$ ($Q \leq 10000$) where $N$ indicates the number of cities, $M$ represents the number of direct flights and $Q$ is the number of queries.

The next $N$ lines contain the city list. The $i$-th of these $N$ lines will contain a string $c_i$ followed by two real numbers $lt_i$ and $ln_i$, representing the city name, its latitude and longitude respectively. The city name will be no longer than 20 characters and will not contain white-space characters. The latitude will be between -90 (South Pole) and +90 (North Pole). The longitude will be between -180 and +180 where negative numbers denote locations west of the meridian and positive numbers denote locations east of the meridian. (The meridian passes through Greenwich, London.)

The next $M$ lines contain the direct flight list. The $i$-th of these $M$ lines will contain two city names $a_i$ and $b_i$ indicating that there exists a direct flight from city $a_i$ to city $b_i$. Be assured that both city names will occur in the city list.

The next $Q$ lines contain the query list. The $i$-th of these $Q$ lines will contain two city names $a_i$ and $b_i$ asking for the minimum distance a passenger needs to fly in order to get from city ai to city bi. Be assured that $a_i b_i$ are not equal and both city names will occur in the city list.

The input will terminate with three zeros form $N$, $M$ and $Q$.

## Output

For each test case in the input first output the test case number (starting from 1) as shown in the sample output. Then for each query in the input print a line giving the shortest distance (in km) a passenger needs to fly to get from the first city ($a_i$) in the query to the second one ($b_i$). If there exists no route form $a_i$ to $b_i$, just print the line '`no route exists`'.

Print a blank line between two consecutive test cases.

## Sample Input

```
3 4 2
Dhaka 23.8500 90.4000
Chittagong 22.2500 91.8333
Calcutta 22.5333 88.3667
```

```
Dhaka Calcutta
Calcutta Dhaka
Dhaka Chittagong
Chittagong Dhaka
Chittagong Calcutta
Dhaka Chittagong
5 6 3
Baghdad 33.2333 44.3667
Dhaka 23.8500 90.4000
Frankfurt 50.0330 8.5670
Hong_Kong 21.7500 115.0000
Tokyo 35.6833 139.7333
Baghdad Dhaka
Dhaka Frankfurt
Tokyo Hong_Kong
Hong_Kong Dhaka
Baghdad Tokyo
Frankfurt Tokyo
Dhaka Hong_Kong
Frankfurt Baghdad
Baghdad Frankfurt
0 0 0
```

## Sample Output

```
Case #1
485 km
231 km

Case #2
19654 km
no route exists
12023 km
```

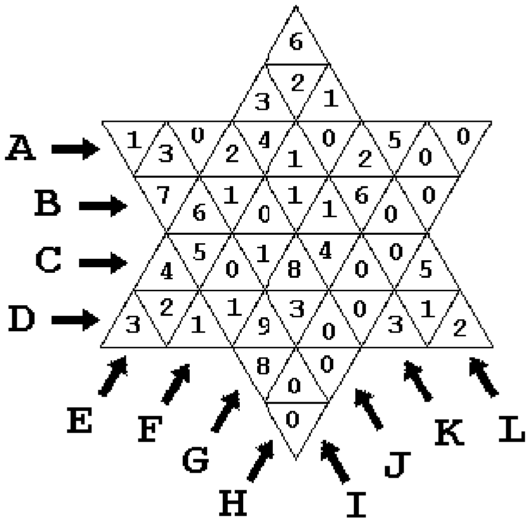
# 10159 Star

A board contains 48 triangular cells. In each cell, there is written a digit (in a range from 0 through 9). Every cell belongs to two or three lines. These lines are marked by letters from A through L.

An example is depicted on the Figure. There, the cell containing digit 9, belongs to lines D, G and I. The cell containing digit 7, belongs to lines B and I.

For each line, A, B, C, ..., L, we consider the largest digit lying on it. In the above example, the largest digit for line A is 5, for line B is 7, for line E is 6, for line H is 0, for line J is 8 and so on.

Write a program, that inputs the largest digit for any one of the depicted 12 lines. The program should find out the smallest and the largest possible sum of digits located in all the cells of the board.




## Input

Every line in the input contains 12 digits, each two of them separated by a space. The first of these digits means the largest one in line A, the second means the largest one in line B, and so on, the last digit means the largest one in line L.

## Output

For every line in the input file write the value of the smallest and of the largest possible sum of digits located in the cells of the board, on a single line. These two values should be separated by one space exactly. If there does not exists a solution, your program must outputs the words '`NO SOLUTION`' instead of the above two values.

## Sample Input

```
5 7 8 9 6 1 9 0 9 8 4 6
```

## Sample Output

```
40 172
```

UVa Online Judge

# 10161 Ant on a Chessboard

One day, an ant called Alice came to an M*M chessboard. She wanted to go around all the grids. So she began to walk along the chessboard according to this way: (you can assume that her speed is one grid per second)

At the first second, Alice was standing at (1,1). Firstly she went up for a grid, then a grid to the right, a grid downward. After that, she went a grid to the right, then two grids upward, and then two grids to the left¡in a word, the path was like a snake.

For example, her first 25 seconds went like this:

( the numbers in the grids stands for the time when she went into the grids)

| | | | | | |
|---|---|---|---|---|---|
| 25 | 24 | 23 | 22 | 21 | **5** |
| 10 | 11 | 12 | 13 | 20 | **4** |
| 9 | 8 | 7 | 14 | 19 | **3** |
| 2 | 3 | 6 | 15 | 18 | **2** |
| 1 | 4 | 5 | 16 | 17 | **1** |
| **1** | **2** | **3** | **4** | **5** | |

At the 8-th second , she was at (2,3), and at 20-th second, she was at (5,4).

Your task is to decide where she was at a given time (you can assume that $M$ is large enough).

## Input

Input file will contain several lines, and each line contains a number $N$ ($1 \leq N \leq 2*10^9$), which stands for the time. The file will be ended with a line that contains a number '0'.

## Output

For each input situation you should print a line with two numbers ($x$,$y$), the column and the row number, there must be only a space between them.

## Sample Input

```
8
20
25
0
```

## Sample Output

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

# 10177 (2/3/4)-D Sqr/Rects/Cubes/Boxes?

You can see a $(4 \times 4)$ grid below. Can you tell me how many squares and rectangles are hidden there? You can assume that squares are not rectangles. Perhaps one can count it by hand but can you count it for a $(100 \times 100)$ grid or a $(10000 \times 10000)$ grid. Can you do it for higher dimensions? That is can you count how many cubes or boxes of different size are there in a $(10 \times 10 \times 10)$ sized cube or how many hyper-cubes or hyper-boxes of different size are there in a four-dimensional $(5 \times 5 \times 5 \times 5)$ sized hypercube. Remember that your program needs to be very efficient. You can assume that squares are not rectangles, cubes are not boxes and hyper-cubes are not hyper-boxes.

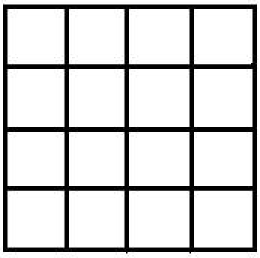


Fig: A $4 \times 4$ Grid

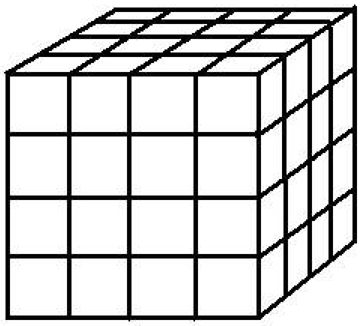


Fig: A $4 \times 4 \times 4$ Cube

## Input

The input contains one integer $N$ $(0 \leq N \leq 100)$ in each line, which is the length of one side of the grid or cube or hypercube. As for the example above the value of $N$ is 4. There may be as many as 100 lines of input.

## Output

For each line of input, output six integers $S_2, R_2, S_3, R_3, S_4, R_4$ in a single line where $S_2$ means no of squares of different size in $(N \times N)$ two-dimensional grid, $R_2$ means no of rectangles of different size in $(N \times N)$ two-dimensional grid. $S_3, R_3, S_4, R_4$ means similar cases in higher dimensions as described before.

## Sample Input

```
1
2
3
```

## Sample Output

```
1 0 1 0 1 0
5 4 9 18 17 64
14 22 36 180 98 1198
```

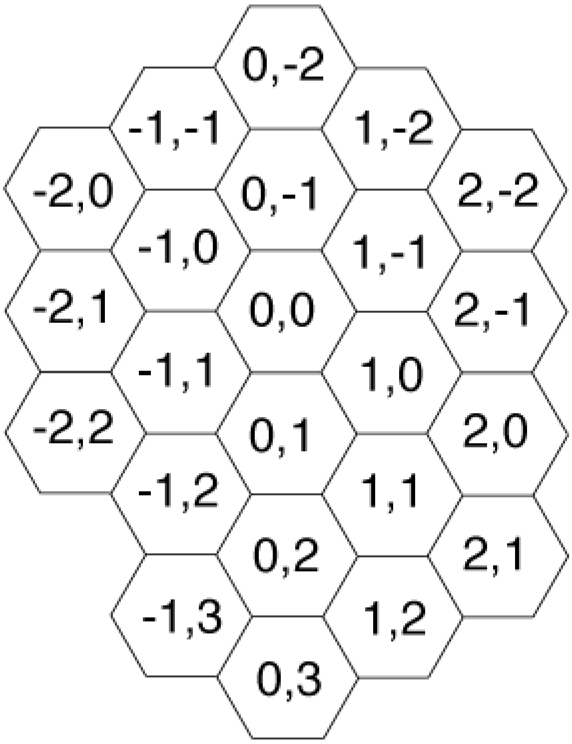
# 10182 Bee Maja

Maja is a bee. She lives in a bee hive with thousands of other bees. This bee hive consists of many hexagonal honey combs where the honey is stored in.

But bee Maja has a problem. Willi told her where she can meet him, but because Willi is a male drone and Maja is a female worker they have different coordinate systems.

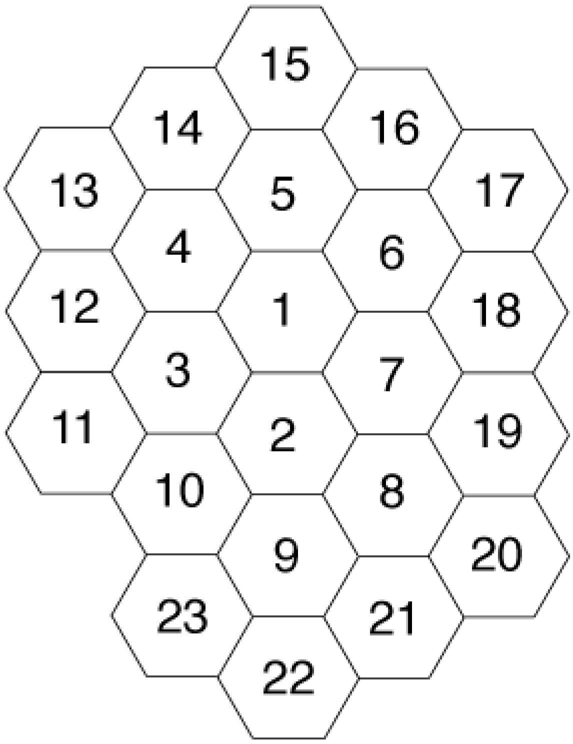
**Maja's Coordinate System**

Maja who often flies directly to a special honey comb has laid an advanced two dimensional grid over the whole hive.




**Willi's Coordinate System**

Willi who is more lazy and often walks around just numbered the cells clockwise starting from 1 in the middle of the hive.




Help Maja to convert Willi's system to hers. Write a program which for a given honey comb number gives the coordinates in Maja's system.

## Input

The input file contains one or more integers which represent Willi's numbers. Each number stands on its own in a separate line, directly followed by a newline. The honey comb numbers are all less than 100 000.

## Output

You should output the corresponding Maja coordinates to Willi's numbers, each coordinate pair on a separate line.

## Sample Input

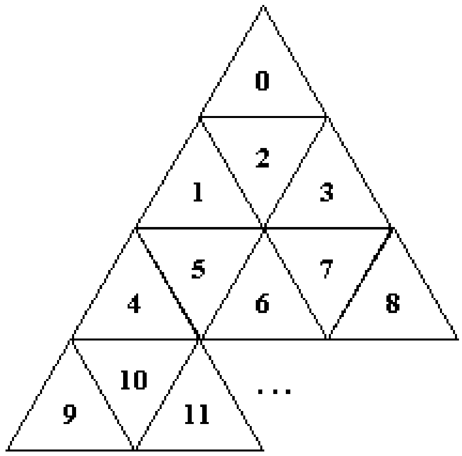
```
1
```

```
2
3
4
5
```

## Sample Output

```
0 0
0 1
-1 1
-1 0
0 -1
```

# 10233 Dermuba Triangle




Dermuba Triangle is the universe-famous flat and triangular region in the L-PAX planet in the Geometria galaxy. Actually nobody knows whether the region is triangular or how it came into existence or how big it is. But the people of Dermuba firmly believe that the region extends to infinity. They live in equilateral triangular fields with sides exactly equal to 1km. Houses are always built at the circumcentres of the triangular fields. Their houses are numbered as shown in the figure on the right.

When Dermubian people wishes to visit each other, they follow the shortest path from their house to the destination house. This shortest path is obviously the straight line distance that connects these two houses. Moreover, they also visit all the houses that lie in the straight line they travel. Now, comes your task. You have to write a program which computes the length of the shortest path between two houses given their house numbers.

## Input

Input consists of several lines with two non-negative integer values $n$ and $m$ which specify the start and destination house numbers. $0 \leq n$, $m \leq 2147483647$. Actually, there are houses beyond this region, but some seventh-sense people in Dermuba say that these houses are left for the dead people.

## Output

For each line in the input, print the shortest distance between the given houses in kilometers rounded off to three decimal places.

## Sample Input

```
0 7
2 8
9 10
10 11
```

## Sample Output

```
1.528
1.528
0.577
0.577
```