

2000-2001 ACM International Collegiate Programming Contest

Sponsored by IBM

Asia Regional Contest / Shanghai



Shanghai University, Shanghai

Contest Session

October 22, 2000

This problem set should contain nine (9) problems on nineteen (19) numbered pages. Please inform a runner immediately if something is missing from your problem set.

**2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session**

General Remarks

Welcome to ACM International Collegiate Programming Contest Asia Regional Contest / Shanghai. Enjoy the game.

1. A PDF file of those problems is provided for better effect of reading. Please refer to the file if you have any questions about colors in problems. NT users may find the file "icpcasiashu2000.pdf" in the directory "C:\work\". LINUX users may find it in the directory "/usr/work/".
2. After submitting a solution for judging, you will get a message from judges. The message should be one of the following information:
correct answer
compilation error
run-time error
time-limit exceeded
wrong answer
presentation error
other - contact staff
3. Additional editors EMACS and VI have been installed for NT users, you will find shortcuts on your desktop. It depends on you whether or not to use those two editors.

Problem A

Coloration Problem

Input file: color.in

Mr. Richards, an enthusiastic mathematician, is spending his holiday in Shanghai. He is so crazy about his study that he may associate anything with mathematical problems. According to his schedule, he should visit the Museum today. But after getting up, he was in a daze in the hotel, thinking deeply about a coloration problem. Yesterday, when he took a walk in the nightfall, the slabs of stone paved on the piazza fascinated him. He has associated those stones with a combinatorial problem.

The shape of those slabs of stone is equilateral triangle whose side length is 1 meter. But if Mr. Richards arranges 4 slabs of stone, as Fig.1 below shows, those stones may form a larger equilateral triangle whose side length is 2 meters. He designates those stones with the numbers 1 to 4 first, then he tries to use three different colors —blue, green and red —to color those stones. In such a triangle, the contiguous stones are required not to be painted with the same color. He asks himself to find out the number of the coloration methods. Since he is a learned person in combinatorial theory, it's not a difficult problem for him. Soon he works out the answer, 24. You know, because of the marks on those stones, they should not be regarded as the same ones.

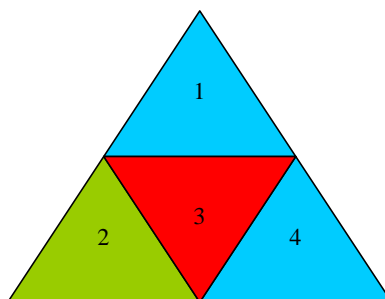


Fig.1 The colored slabs of stone

Arranging for more slabs of stone to form a larger triangle, and using more colors, the coloration problem will be much more complex. For example, arranging 9 slabs of stone, as Fig.2 below shows, Mr. Richards has to form a triangle whose side length is 3 meters. Using 5 colors to paint on those stones, the answer will be an enormous number. Without a computer, he could not tell the result exactly, although he knows clearly how to account the coloration methods. So you are required to write a program to tell him the exact answer.

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

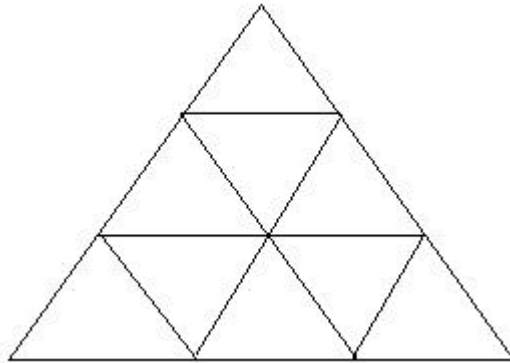


Fig.2 The triangle whose side length is 3 meters

Input

The input file may contain several data sets. Each data set consists of two integers l ($0 \leq l \leq 6$) and c ($1 \leq c \leq 4$). The first integer l represents the side length of the triangle formed by slabs of stone. The second integer c represents the number of the kinds of the colors. The input file is terminated by $l = 0$.

Output

For each data set, compute s , the number of the coloration methods without any two contiguous stones painted with the same color. If the colors given are insufficient to paint the slabs of stone, the answer should be 0. Output your answer as a single integer on a line by itself.

Note: You are required to tell him the exact answers, not the floating-point values.

Sample Input

```
2 3
6 4
0 0
```

Output for the Sample Input

```
24
3470494144278528
```

Problem B

Trip! Trip! Trip!

Input file: trip.in

In such a demanding age, after working busily day after day, undoubtedly people would expect to enjoy their leisure time. What do they want to do first? It's the trip. Nowadays, in the eyes of a traveler who hopes to visit tourist attractions all over the country, the world is wonderful. Since there are so many travel agencies, if he wants to participate in an organized trip, he may choose any of them. But for a travel agency, the competition seems to be much more ruthless, for he has so many rivals in scrambling for the tourists.

Company trip center is one of those unfortunate travel agencies. After investigating from the questionnaires, the agent has found out that there are some tourists who do not get along with each other, so he has to choose some of them to organize a trip group. But he had made many mistakes in choosing those members before, and they caused many of his customers shift to his rivals. Although he could not tell exactly how many rivals he has, he knows clearly that his profits have diminished terribly. So he decides to take action to change the situation.

Company trip center pays attention to that problem at the beginning. Initially the agent does not know what tourists do not like each other, but after discussing with those tourists, he is able to acquire the information. So he will be able to form a trip group from some of those tourists in which no one is unwilling to get along with others. If company A could make a wise decision, there will be maximum number of tourists in this group, which will undoubtedly bring the biggest profits to him. What a great idea, isn't it?

You are required to write a program for company trip center to solve the problem. The solution will exist for sure. After writing such a program, you will get an acknowledgement from him.

Input

The input file for this problem will consist of multiple test cases. The first line of each test case contains an integer n ($n \leq 50$), which specifies the number of tourists participating in the trip group. A value of 0 for the integer n indicates the end of input, and this test case should not be processed. Each line in the following n lines contains n characters without any separators. Each character should be either "1" or "0", no characters else will appear in the input file. The character "1" means that the two tourists could not get along well with each other, so you shouldn't arrange them in one trip group. And the character "0" means they will make no difficulty to join the same group. For example, if the 2nd character in the 1st line is "1", the trip group may comprise either the 1st tourist or the 2nd tourist, if the character is "0", the trip group may comprise both of them.

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

Assumption: Obviously, a tourist could not be unwilling to get along with himself. So you may assume the corresponding character will always be "0". If a tourist does not wish to get along with another one, the latter will be unwilling with the former either. You may assume the data in the input file appear conjugated.

Output

For each test case, your program should output one integer in a line, which specifies the maximum tourists in the trip group. No more white spaces or blank lines in the output are allowed.

Sample Input

```
6
010001
100100
000110
011000
001001
100010
0
```

Output for the Sample Input

```
3
```

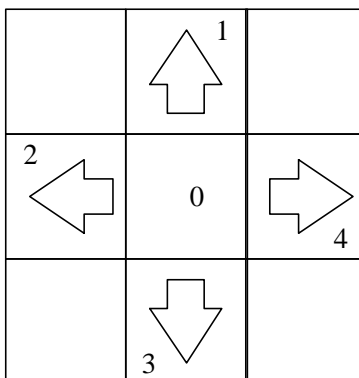
Problem C

Dance Dance Revolution

Input file: ddr.in

Mr. White, a fat man, now is crazy about a game named “Dance, Dance, Revolution”. But his dance skill is so poor that he could not dance a dance, even if he dances arduously every time. Does “DDR” just mean him a perfect method to squander his pounds? No way. He still expects that he will be regarded as “Terpsichorean White” one day. So he is considering writing a program to plan the movement sequence of his feet, so that he may save his strength on dancing. Now he looks forward to dancing easily instead of sweatily.

“DDR” is a dancing game that requires the dancer to use his feet to tread on the points according to the direction sequence in the game. There are one central point and four side points in the game. Those side points are classified as top, left, bottom and right. For the sake of explanation, we mark them integers. That is, the central point is 0, the top is 1, the left is 2, the bottom is 3, and the right is 4, as the figure below shows:



At the beginning the dancer's two feet stay on the central point. According to the direction sequence, the dancer has to move one of his feet to the special points. For example, if the sequence requires him to move to the top point at first, he may move either of his feet from point 0 to point 1 (*Note: Not both of his feet*). Also, if the sequence then requires him to move to the bottom point, he may move either of his feet to point 3, regardless whether to use the foot that stays on point 0 or the one that stays on point 1.

There is a strange rule in the game: moving both of his feet to the same point is not allowed. For instance, if the sequence requires the dancer to the bottom point and one of his feet already stays on point 3, he should stay the very foot on the same point and tread again, instead of moving the other one to point 3.

After dancing for a long time, Mr. White can calculate how much strength will be consumed when he moves from one point to another. Moving one of his feet from the central point to any side

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

points will consume 2 units of his strength. Moving from one side point to another adjacent side point will consume 3 units, such as from the top point to the left point. Moving from one side point to the opposite side point will consume 4 units, such as from the top point to the bottom point. Yet, if he stays on the same point and tread again, he will use 1 unit.

Assume that the sequence requires Mr. White to move to point $1 \rightarrow 2 \rightarrow 2 \rightarrow 4$. His feet may stay on (point 0, point 0) \rightarrow (0, 1) \rightarrow (2, 1) \rightarrow (2, 1) \rightarrow (2, 4). In this couple of integers, the former number represents the point of his left foot, and the latter represents the point of his right foot. In this way, he has to consume 8 units of his strength. If he tries another path, he will have to consume much more strength. The 8 units of strength is the least cost.

Input

The input file will consist of a series of direction sequences. Each direction sequence contains a sequence of numbers. Each number should either be 1, 2, 3, or 4, and each represents one of the four directions. A value of 0 in the direction sequence indicates the end of direction sequence. And this value should be excluded from the direction sequence. The input file ends if the sequence contains a single 0.

Output

For each direction sequence, print the least units of strength will be consumed. The result should be a single integer on a line by itself. Any more white spaces or blank lines are not allowable.

Sample Input

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

Output for the Sample Input

```
8
22
```

Problem D

Helicopter

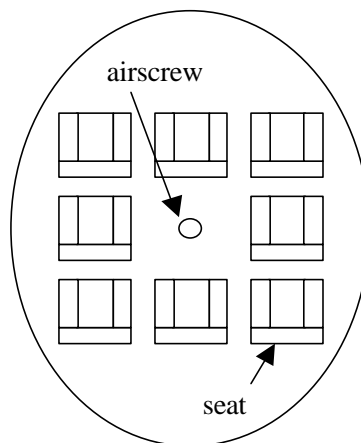
Input file: heli.in

Since ancient time, people have been dreaming of flying in the sky. Eventually, the dream was realized in the 20th century. Nowadays, the airplane becomes a useful vehicle that is used frequently in our daily life.

But before the dream came true, a large number of people had tried to design the aircrafts. One of those aircrafts, which is called “helicopter” in modern time, can be traced back to the blueprint of the aircraft designed by Leonardo da Vinci. But the helicopter was not effective enough till this century.

Since the helicopter rises through the updraft generated by the airscrew, it is very important for it to keep balance. Even for the late-model helicopters, the loads are required to be distributed evenly, so that the center of gravity of the helicopter lies just underneath the airscrew. It is one of the most important requirements for it in flight.

Now, you are requested by an airline company to write a program for a passenger transport helicopter. The program may arrange a seat for each passenger automatically so that the center of gravity of the helicopter should be located underneath the airscrew as close as possible. The seats in the helicopter and the airscrew are designed as the figure below.



You may assume the distance of the adjoining seats is 1 unit, and the airscrew occupies a seat. A seat along with a passenger on it will generate a transverse moment and a longitudinal moment. The transverse moment, Mv_i , is the weight of a passenger on the seat multiplied by the transverse distance from the seat to the airscrew. The longitudinal moment, Mh_i , is the weight of a passenger on the seat multiplied by the longitudinal distance. If the transverse moments generated by the passengers on the left are assumed to be positive, the moments by the passengers

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

on the right will be negative. Also, if the longitudinal moments generated by the passengers in front are assumed to be positive, the moments by the passengers on the back will be negative. That is, the moments may counteract with each other. You may use the formula below to figure out the composition of moments.

$$M = \sqrt{\left(\sum_{i=1}^8 Mv_i\right)^2 + \left(\sum_{i=1}^8 Mh_i\right)^2}$$

If $M = 0$, the center of gravity of the helicopter lies just underneath the airscrew.

You are required to arrange the seats of 8 passengers according to their weights to locate the center of gravity underneath the airscrew as far as possible. That is, the value of M should be minimum.

Input

The input file may contain several test cases. Each test case consists of eight integers in lines, which represent the weights of those passengers. The end of input is signified by the test case in which the weights are all 0. And this test case should not be processed.

Output

The output for each test case should include a line contains a real number which tells the composition of moments, M , in the optimal arrangement. The output of the composition of moments should be accurate to 3 decimal places. And you should not print any more white spaces or blank lines in the output.

Sample Input

```
1 2 3 4 5 6 7 8
0 0 0 0 0 0 0 0
```

Output for the Sample Input

```
0.000
```

Problem E

Card Game in FF8

Input file: cardgame.in

Introduction

The cards originated when a psychic named Orlan modified these tarot cards for games. The game became popular with soldiers passing time between battles. As its popularity spread, each region developed its own rules and picture variations.

Diagram

1. Picture;
2. Numbers: Correspond to the 4 sides of the card. (1:weak – 10:strong)
3. Element: Some rules allow the players to use its elemental property.



Battle Area

1. Battle area: Place cards here one at a time.
2. Elemental: Some rules indicate elements on the board.

Basic Rules

There are two players in the game.

One of them is the red player, and the other is the blue one. They place their own cards on the battle area in turn, the red player places red cards while the blue player places blue cards. Placing a card on the location of the area that has a card is not allowed. A Card in the battle area will be turn over if its value on side adjacent to new card is less than new card's value on side adjacent to it. That is, the card's color will be changed, and it will belong to the other player. For example, as the figure shows, assume a red card has been placed on the battle area, and a blue one is now placed on the area adjacent to the right side of the red one. Since the value of the left side of the old card is 2, and the value of the right side of the new card is 7, the red card will be turned over to be a blue one, and it belongs to the blue player. When all 9 areas are filled, the outcome of the game is judged. If the player who put first card has 5 cards in battle area, it's a tie. If he has more cards, he wins the game. But if the amount of his cards is less than 5, unfortunately, he is defeated.



Note: Only the card is being placed on the battle area may turn over those that have been already placed on the area. It will not be turned over by the old ones.

**2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session**



Same Rule

Same rule will turn over cards that have same values on 2 or more sides. As the figure shows, two red cards have already been placed on the battle area. The value of the down side of the up card is 4, and that of the right side of the left card is 1. If a blue card whose up side is 4 and right side is 1 is placed on the area, while it's adjacent to the down side of the up one and adjacent to the right side of the left one, both of the red cards will be turned over.

Note: The adjacent cards are not required to be the rival's ones. In the example above, if the up card is blue, the blue card to be placed on the battle area will turn over the left red one also.

Plus Rule

Plus Rule means that cards adding to the same value on 2 or more adjacent sides will be turned over. As the figure shows, two red cards have been placed on the area. The value of the up one's down side is 3, and that of the left one's right side is 5. If a blue card whose up side is 4 and left side is 2 is placed on the area, while it's adjacent to the down side of the up one and adjacent to the right side of the left one, both of the red cards will be turned over. Since both of the sum values are 7.



Note: The adjacent cards are not required to be the rival's ones. In the example above, if the up card is blue, the blue card to be placed on the battle area will turn over the left red one also.



Combo Rule

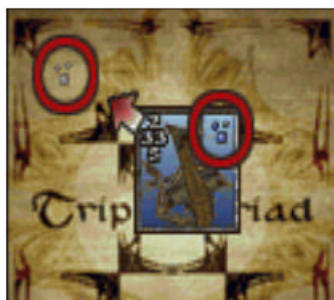
Cards turned over using *Same Rule* or *Plus Rule* turn over adjacent cards in *Combo Rule*. As the figure shows, a red card and two blue cards are on the battle area. The value of the left side of the red one is 2, and that of the down side is 4. And the red one is placed in the top left corner. A blue card whose left side is 3 is adjacent to the right side of the red one, and another blue card whose up side is 1 is adjacent to the down side of the red one. Assume those two cards are just turned over to be blue by *Same Rule* or *Plus Rule* by another card, using *Combo Rule*, the red one will be turned over, because the value of the right blue card's left side is larger than the value of the red one's right side ($3 > 2$), although the value of the down blue card's up side is less than the value of the red one's down side ($1 < 4$).

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

Note: In the example above, Combo Rule is used, not Plus Rule. If the right blue card is just turned over by Same Rule or Plus Rule, using Combo Rule, the red one will be turned over ($3 > 2$). But if the down blue card is just turned over by Same Rule or Plus Rule, using Combo Rule, the red one will not be turned over ($1 < 4$).

Same Wall Rule

Same Wall Rule uses Battle Area Wall (the frame) as one of card's 4 sides when using *Same Rule*. The value of the wall on any sides may be assumed to be 10. That is, the card that is adjacent to the frame may be turned over using *Same Wall Rule*.



Elemental Rule

Elemental Rule will change the value of a card depending on the elemental property, as the figure shows. When a card and a slot match elemental properties, the value of the four sides will be increased by 1. If elements don't match, the value of the four sides of the card will be decreased 1 from the original value.

You are required to write a program to judge the outcome of the game according to the rules above, and tell us whether the game is a tie. If it isn't, please inform us the winner and the number of cards the players have.

Input

There are data groups of several card games in one input file. Each card game will begin with a line consists of 5 integers, which tell whether the rules above will be adopted respectively. Those integers should be either 1 or 0. If an integer is 1, the corresponding rule will be adopted. Otherwise, the rule will be disabled. Those 5 integers stand for, in order, basic rules, same rule, plus rule, combo rule, and same wall rule. The end of input is marked by the card game in which basic rules are disabled. And this card game should not be processed. If the game does not indicate the end of the input, there will be 3 following lines, which tell the elemental properties of the battle area. Each line contains 3 integers from 0 to 9. The 3×3 array of integers defines the locations of the battle area. Each location's elemental property is described by the corresponding integer. A value of 0 indicates that the corresponding location has no elemental property. While all of the 9 locations do not have elemental properties, elemental rule is disabled. Otherwise, elemental rule will be adopted. The following are 9 lines of integers, each of them tells the information of a card to be placed on the battle area by one of the players. A card is described by 9 integers. The first two integers are x and y ($1 \leq x, y \leq 3$), indicating the row and column of the location on which the card will be placed. The remains are, in order, u, l, d, r , and e ($1 \leq u, l, d, r \leq 10, 1 \leq e \leq 9$). The integers u, l, d and r represent the values of the up side, left side, down side and right side of the card, and the integer e represents the elemental property of the card.

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

Assumption

1. *If the rules above are adopted, the priorities are, in order, from high to low, elemental rule, same rule, same wall rule, plus rule, combo rule, and basic rules.*
2. *If same rule and plus rule are disabled, combo rule will be useless, no matter whether it is adopted.*
3. *If same rule is disabled, same wall rule will be useless, no matter whether it is adopted.*
4. *You may assume the data groups in the input are always correct. That is, a card will not be placed on the location that has another card already, and there will not be any integers that violate the prescriptions.*
5. *The player who put the first card is regarded as player 1, and the other is regarded as player 2.*

Output

For each card game, show the outcome according to the rules above. The first line of the outcome contains 3 integers a, b, c , indicating the winner and the cards the players hold finally. The integer a represents the number of the player 1's cards, and b represent the number of the player 2's cards. If the player 1 wins the game, $c = 1$; if he loses, $c = 2$; and if it's a tie, $c = 0$. The following 3 lines contain 3 integers each, describing the results of the battle area. Those integers are from 1 to 2. If the location of the battle area has a card owned by player 1, print 1 at the corresponding location. If it has a card owned by player 2, print 2 accordingly.

Sample Input

```
1 1 1 1 0
0 0 0
3 0 0
0 0 0
1 1 1 2 3 4 1
1 2 4 5 3 3 2
2 1 3 4 5 3 3
2 2 4 3 7 8 4
3 3 1 2 3 4 5
2 3 5 8 1 6 6
1 3 2 1 1 3 7
3 1 3 3 3 3 8
3 2 8 3 2 1 9
0 0 0 0 0
```

Output for the Sample Input

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

Problem F

Hot or Cold?

Input file: hot.in

John Smith, who is a member of Academy of Cold Manager (ACM), is in charge of a large-scale cold store. For him, it's a troublesome job. Whenever the temperature in the cold store is too hot or too cold for a long time, the goods will be damaged. And poor Mr. Smith will have to compensate for the loss of the store.

Therefore, Mr. Smith has installed an automatic temperature control system in the store. The system may control the temperature according to the input polynomial and the start time. At each moment, it tries to adjust the temperature equal to the value of the polynomial function. But Mr. Smith still feels worried. Since he could not know the effect beforehand how the system will regulate. At such a worrisome moment, it's lucky for him to call to remembrance that you, an excellent programmer, are willing to offer a program to help him. Making use of this program, he may simply input the polynomial and the parameters of the start time and the end time, and then he will be aware of the average temperature during this period. Now he is relieved from such a bothersome job.

Input

The input file may contain several data sets. Each data set begins with a line containing an integer n ($n < 100$), which specifies the highest power of the polynomial. A value of 0 for the power indicates the end of input, and this data set should not be processed. In each data set, the following line contains $n+1$ real numbers, which tell the coefficients of the polynomial. The sequence of those coefficients is arranged according to the power of items from high to low in the polynomial. If an item of the polynomial does not exist, the corresponding coefficient is 0. And then follows a line consists of 2 real numbers s and e ($s < e$), indicating the start time and the end time.

Output

For each data set, compute t , the average temperature of the input polynomial from the start time to the end time. Print your answer as a single real number accurate to 3 decimal places on a line by itself. You should not print any more white spaces or blank lines in the output.

Sample Input

```
2
1.0 0.0 0.0
0.0 1.0
0
```

Output for the Sample Input

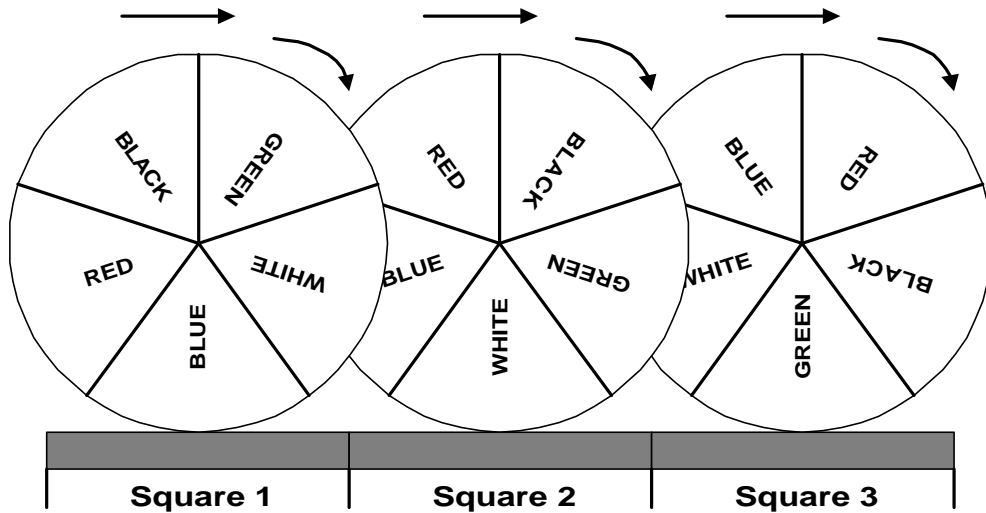
```
0.333
```

Problem G

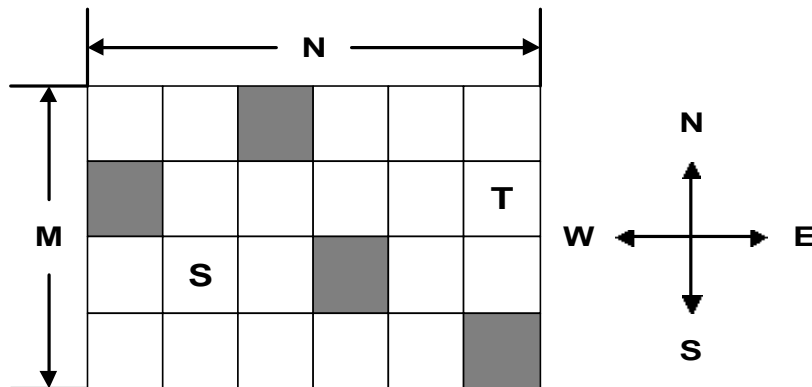
The Monocycle

Input file: cycle.in

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°) 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° around its own center. The effect is shown in the above figure. When the wheel is at the center of square 1, the mid-point 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 mid-point of its white segment touches the ground.



2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai University, Shanghai, China, October 22, 2000, Contest Session

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° left or right. Each of these actions requires exactly 1 second to execute. He always starts his ride facing north and with the mid-point 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
```

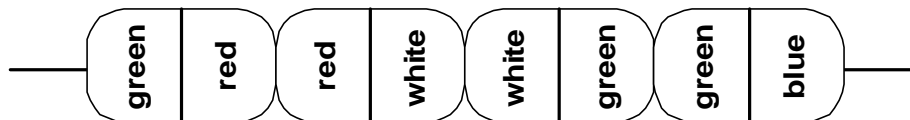
```
Case #2
minimum time = 49 sec
```

Problem H

The Necklace

Input file: necklace.in

My little sister had a beautiful necklace made of colorful beads. Two successive beads in the necklace shared a common color at their meeting point. The figure below shows a segment of the necklace:



But, alas! One day, the necklace was torn and the beads were all scattered over the floor. My sister did her best to recollect all the beads from the floor, but she is not sure whether she was able to collect all of them. Now, she has come to me for help. She wants to know whether it is possible to make a necklace using all the beads she has in the same way her original necklace was made and if so in which order the beads must be put.

Please help me write a program to solve the problem.

Input

The input contains T test cases. The first line of the input contains the integer T .

The first line of each test case contains an integer N ($5 \leq N \leq 1000$) giving the number of beads my sister was able to collect. Each of the next N lines contains two integers describing the colors of a bead. Colors are represented by integers ranging from 1 to 50.

Output

For each test case in the input first output the test case number as shown in the sample output. Then if you apprehend that some beads may be lost just print the sentence "some beads may be lost" on a line by itself. Otherwise, print N lines with a single bead description on each line. Each bead description consists of two integers giving the colors of its two ends. For $1 \leq i \leq N - 1$, the second integer on line i must be the same as the first integer on line $i + 1$. Additionally, the second integer on line N must be equal to the first integer on line 1. Since there are many solutions, any one of them is acceptable.

Print a blank line between two successive test cases.

2000-2001 ACM International Collegiate Programming Contest
Asia Regional Contest / Shanghai
Shanghai Univeristy, Shanghai, China, October 22, 2000, Contest Session

Sample Input

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

Sample Output

```
Case #1
some beads may be lost
```

```
Case #2
2 1
1 3
3 4
4 2
2 2
```

Problem I

Digital Rivers

Input file: digital.in

A digital river is a sequence of numbers where the number following n is n plus the sum of its digits. For example, 12345 is followed by 12360, since $1+2+3+4+5 = 15$. If the first number of a digital river is k we will call it river k .

For example, river 480 is the sequence beginning {480, 492, 507, 519, ...} and river 483 is the sequence beginning {483, 498, 519, ...}.

Normal streams and rivers can meet, and the same is true for digital rivers. This happens when two digital rivers share some of the same values. For example: river 480 meets river 483 at 519, meets river 507 at 507, and never meets river 481.

Every digital river will eventually meet river 1, river 3 or river 9. Write a program that can determine for a given integer n the value where river n first meets one of these three rivers.

Input

The input may contain multiple test cases. Each test case occupies a separate line and contains an integer n ($1 \leq n \leq 16384$). A test case with value of 0 for n terminates the input and this test case must not be processed.

Output

For each test case in the input first output the test case number (starting from 1) as shown in the sample output. Then on a separate line output the line "first meets river x at y ". Here y is the lowest value where river n first meets river x ($x = 1$ or 3 or 9). If river n meets river x at y for more than one value of x , output the lowest value.

Print a blank line between two consecutive test cases.

Sample Input

```
86
12345
0
```

Sample Output

```
Case #1
first meets river 1 at 101

Case #2
first meets river 3 at 12423
```