Problem A : Sim Card

There are three mobile operators in Iran. Each operator has different prices for call and data usage, given in the table below. All prices are in Rials:

#	Name	Call (per minute)	Data (per megabyte)
1	ParsTel	30	40
2	ParsCell	35	30
3	ParsPhone	40	20

Some foreign students have arrived Iran to participate in the ACM-ICPC, Tehran Site. They already know how many minutes they will call, and how much Internet they will use. For each student, you want to recommend an operator to minimize the total cost of call usage and data usage for that student.

Input (Standard Input)

Each line of the input contains the information of one student. For each student, there are two positive integers c and d $(1 \leqslant c, d \leqslant 1000)$ that show the amount of call (in minutes) and data usage (in megabytes) for the student, respectively. The input terminates with "`0 0`" that should not be processed.

Output (Standard Output)

For each student, print a line containing the minimum total cost of call usage and data usage.

Sample Input and Output

Standard Input	Standard Output
10 60	1600
100 20	3800
24 12	1200
900 400	43000
50 50	3000
0 0	

Problem B : Bank Card Verifier

Cafe Bazaar, the famous Iranian android market, is looking for creative software developers. A group of applicants are attending an interview, and the company wants to select the fastest developer who can code simple rules accurately. As a test, all applicants should quickly develop a bank card verifier that determines whether a payment card number is valid or not.

All payment card numbers are 16 digits long. The leftmost 6 digits represent a unique identification number for the bank who has issued the card. The next 2 digits determine the type of the card (e.g., debit, credit, gift). Digits 9 to 15 are the serial number of the card, and the last digit is used as a control digit to verify whether the card number is valid. Hence, if somebody enters the card number incorrectly, there is a high chance that a payment software can easily determine it.

For a valid card number, the last digit is selected in such a way that the following algorithm passes:

1. Label all digits from left to right by 1 to 16.
2. Multiply each odd-labeled digit by 2.
3. If the result for any digit is greater than 9, subtract 9 from it.
4. Sum the results of the previous step, and add to it the sum of all even-labeled digits.
5. If the result is a multiple of 10, the card number is valid; otherwise, it is invalid.

Your task is to read several card numbers from the input, and determine whether each one is a valid card number or not.

Input (Standard Input)

There are multiple test cases in the input. Each test is given in one line consisting of four space-separated 4-digit strings. The leftmost digit of the given card number is guaranteed to be non-zero. The input terminates with a line containing “`0000 0000 0000 0000`” that should not be processed.

Output (Standard Output)

For each test case, output a line containing “`Yes`” or “`No`” depending on whether the card number is valid or not, respectively.

Sample Input and Output

Standard Input	Standard Output

`6104 3376 7866 1545`
`6104 3376 7866 1546`
`5022 2910 0140 7954`
`0000 0000 0000 0000`
	`Yes`
	`No`
	`Yes`

Problem C : Border Wall

Through one of the most controversial presidential elections in the world, Mr. Trunk was finally elected as the president of Demoland. During his long presidential campaign, Mr. Trunk made a series of strange promises, one of which was to build a wall along the border of Demoland with its neighboring country, Indoland.

Now, months after his election, Mr. Trunk has started thinking on how to really build the wall. To keep things simple, he has decided to build the wall along a straight line. However, there are still several difficulties. The main issue is that the houses owned by the citizens of these two neighboring countries are so mixed that it is almost impossible to separate them by a straight wall. Wherever the wall is built, some houses must be evacuated and their residents must be moved to the other side of the wall in order to keep all citizens of each county on one side of the wall. The second issue is that the wall itself has some width, meaning that all houses intersected by the wall must be destructed. Of course, evacuating or destructing each house has a cost, and if the total number of such houses is high, it would raise a vigorous public protest. To make his final decision, Mr. Trunk is looking for a keen advisor to tell him how many houses must be evacuated/destructed at the minimum in order to build a straight border wall of a desired width.

Being active in ACM ICPC, you have been selected as an advisor to Mr. President. The locations of the houses owned by the citizens of each country is given to you as a set of points in the plane. The desired width of the border wall is also given to you by the president. Your task is to compute the minimum number of points (houses) that must be removed (destructed or evacuated) so that the remaining point sets can be separated by a wall of the given width. Two point sets are called separated by a wall if (i) no two points from different sets lie in the same side of the wall, and (ii) none of the points intersects the wall except on its boundary lines.

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains three positive integers n, k, d, where n and k indicate the number of houses owned by the citizens of Demoland and Indoland, respectively, and d indicates the width of the border wall ($1 \leqslant n, k \leqslant 100$, and $0 \leqslant d < 1,000$). The next n lines, each contains two integers x and y ($0 \leqslant x, y \leqslant 10,000$), where (x, y) represents the location of a house owned by a citizen of Demoland. Analogously, the next k lines represent the location of the houses owned by the citizens of Indoland. Note that no two houses are located in the same position. The input terminates with a line containing "`0 0 0`" which should not be processed.

Output (Standard Output)

For each test case, output a line containing the minimum number of the points that must be removed to construct the wall. Note that all houses from a country may be destructed/evacuated in the optimal solution.

Sample Input and Output

Standard Input	Standard Output

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

Problem D : World Cup Draw

The draw for the 2018 FIFA World Cup took place on Friday, December 1 at the State Kremlin Palace in Moscow. The FIFA released the draw procedure a few months ago on the official FIFA website, explained below.

The 32 qualified finalists are first distributed into four seeding pots based on the FIFA ranking for October 2017. Pot 1 contains the host Russia and the seven highest ranking teams; the next eight highest ranked teams in Pot 2, and so on. Then, teams are drawn into eight groups of four, which are labeled from A to H. The pots are emptied into groups in order from Pot 1 through Pot 4. The draw must satisfy the following two rules:

(i) no teams from the same pot can be drawn into the same group.

(ii) with the exception of UEFA, which has more qualified teams (14) than the groups (8), no teams from the same confederation can be drawn in the same group. Moreover, at most two teams from UEFA can be drawn in the same group.

Order groups alphabetically from A to H with A and H respectively being the leftmost and rightmost groups. At each step of the draw, the drawn team x from Pot i is placed in the first group from left (starting from group A) not violating rules (i) and (ii) and being possible to distribute the remaining teams (not drawn teams so far) of Pot i in the next steps without violating rules (i) and (ii). Computer scientists have assured FIFA that it is always possible to distribute teams of Pot i into groups satisfying rules (i) and (ii), regardless of how the other teams in Pot 1 through Pot $i-1$ are distributed into groups.

The table below shows the pots. In this table, $x\ (r,\ c)$ means that team x belongs to confederation c, and its rank in the FIFA ranking is r. You are to write a program to simulate the draw and report the groups for the given draw order for each pot.

Seeding Pot 1	Seeding Pot 2	Seeding Pot 3	Seeding Pot 4
Russia (65, UEFA)	Spain (8, UEFA)	Denmark (19, UEFA)	Serbia (38, UEFA)
Germany (1, UEFA)	Peru (10, CONMEBOL)	Iceland (21, UEFA)	Nigeria (41, CAF)
Brazil (2, CONMEBOL)	Switzerland (11, UEFA)	Costa Rica (22, CONCACAF)	Australia (43, AFC)
Portugal (3, UEFA)	England (12, UEFA)	Sweden (25, UEFA)	Japan (44, AFC)
Argentina (4, CONMEBOL)	Colombia (13, CONMEBOL)	Tunisia (28, CAF)	Morocco (48, CAF)
Belgium (5, UEFA)	Mexico (16, CONCACAF)	Egypt (30, CAF)	Panama (49, CONCACAF)
Poland (6, UEFA)	Uruguay (17, CONMEBOL)	Senegal (32, CAF)	South Korea (62, AFC)
France (7, UEFA)	Croatia (18, UEFA)	Iran (34, AFC)	Saudi Arabia (63, AFC)

Input (Standard Input)

There are multiple test cases in the input. Each test case consists of 4 lines. The ith line presents all 8 team names (as written in the table) in Pot i in the draw order (from left to right). Team names are separated by ”,” and there may exist a space before or after team names. In all test cases, Russia is the first drawn country in Pot 1 due to the old tradition of placing the host country in group A. The input terminates with “`End`” that should not be processed.

Output (Standard Output)

For each test case, simulate the draw based on the given draw order, and compute the weight of each group which is the summation of team ranks in that group. For each group, print the group name (an uppercase letter) and its weight separated by a space in one line. This must be done in the increasing order of the weights from the strongest group (the group with the minimum weight) to the weakest group (the group with the maximum weight). In the case of a tie, print based on the alphabetical order of group names.

Sample Input and Output

Standard Input	Standard Output
Russia, Germany, Brazil, Portugal, Argentina, Belgium, Poland, France	B 73
Spain, Peru, Switzerland, England, Colombia, Mexico, Uruguay, Croatia	C 78
Denmark, Iceland, Costa Rica, Sweden, Tunisia, Egypt, Senegal, Iran	E 83
Serbia, Nigeria, Australia, Japan, Morocco, Panama, South Korea, Saudi Arabia	D 92
Russia, Germany, Brazil, Portugal, Argentina, Belgium, Poland, France	H 107
Spain, Peru, Switzerland, England, Colombia, Mexico, Uruguay, Croatia	F 110
Denmark, Iceland, Costa Rica, Sweden, Tunisia, Egypt, Senegal, Iran	G 118
Serbia, Nigeria, Morocco, Panama, Australia, Japan, South Korea, Saudi Arabia	A 136
End	C 77
	B 78
	E 83
	D 87
	H 108
	F 110
	G 118
	A 136

Problem E : Barareh on Fire

The Barareh village is on fire due to the attack of the virtual enemy. Several places are already on fire and the fire is spreading fast to other places. Khorzookhan who is the only person remaining alive in the war with the virtual enemy, tries to rescue himself by reaching to the only helicopter in the Barareh villiage.

Suppose the Barareh village is represented by an $n \times m$ grid. At the initial time, some grid cells are on fire. If a cell catches fire at time x, all its 8 vertex-neighboring cells will catch fire at time $x + k$. If a cell catches fire, it will be on fire forever. At the initial time, Khorzookhan stands at cell s and the helicopter is located at cell t. At any time x, Khorzookhan can move from its current cell to one of four edge-neighboring cells, located at the left, right, top, or bottom of its current cell if that cell is not on fire at time $x + 1$. Note that each move takes one second.

Your task is to write a program to find the shortest path from s to t avoiding fire.

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains three positive integers n, m and k ($1 \leqslant n, m, k \leqslant 100$), where n and m indicate the size of the test case grid $n \times m$, and k denotes the growth rate of fire. The next n lines, each contains a string of length m, where the jth character of the ith line represents the cell (i, j) of the grid. Cells which are on fire at time 0, are presented by character “`f`”. There may exist no “`f`” in the test case. The helicopter and Khorzookhan are located at cells presented by “`t`” and “`s`”, respectively. Other cells are filled by “`-`” characters. The input terminates with a line containing “`0 0 0`” which should not be processed.

Output (Standard Output)

For each test case, output a line containing the shortest time to reach t from s avoiding fire. If it is impossible to reach t from s, write “`Impossible`” in the output.

Sample Input and Output

Standard Input	Standard Output
7 7 2 f------ -f---f- ----f-- ------- ------f ---s--- t----f- 3 4 1 t--f --s- ---- 2 2 1 st f- 2 2 2 st f- 0 0 0	4 Impossible Impossible 1

Problem F : Homotopic Paths

Soroush plans to walk through a predefined path in the birds garden from the entrance point to the exit point to enjoy watching birds on trees. His playful dog on a leash does not necessarily follow the same path but he finally reaches the exit point. The garden is full of tall trees, and there is no other obstacle. The leash is constructed in such a way that its length can vary from 0 to any length but it tends to have the smallest length at any time. Soroush and his dog start their journey from the entrance point with the leash length to be zero. Soroush is wondering by knowing his dog's path in advance, if it is possible for the leash length to be zero when they both arrive at the exit point. Note that the leash can not be paased over the trees. If the leash length is zero at the exit point, the Soroush's path and his dog's path are called homotopic. Your task is to write a program to determine whether two given paths are homotopic.

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains three positive integers n, m and k ($0 \leqslant m, k \leqslant 1,000$ and $1 \leqslant n \leqslant 1,000$) where n is the number of trees, and m and k are the the number of the intermediate vertices of the Soroush's path and his dog's path, respectively. The next two lines contain the x and y coordinates of the entrance point s and the exit point t ($s \neq t$) in that order. The next n lines present the x and y coordinates of the trees. Let Soroush's path and his dog's path be $s, v_1, \ldots, v_m, t$ and $s, u_1, \ldots, u_k, t$, respectively. At the end, the x and y coordinates of $v_1, \cdots, v_m$ come in m lines in order and then the x and y coordinates of $u_1, \cdots, u_k$ come in k lines in order. Note that the paths may have self-intersections or intersect each other. There is no tree on the Soroush's path or his dog's path including s and t. You can assume all input coordinates are integers whose absolute values are at most 10^6. The input terminates with a line containing "`0 0 0`" which should not be processed.

Output (Standard Output)

For each test case, output a line containing "`Yes`" if the two given paths are homotopic. Output "`No`", otherwise.

Sample Input and Output

Standard Input	Standard Output

```
2 4 8
0 2
7 2
2 1
5 1
6 2
6 0
1 0
1 3
3 2
3 0
1 0
1 3
6 3
6 0
4 0
4 2
1 2 0
0 0
10 0
3 1
5 1
1 0
0 0 0
```
	```
	No
	Yes
	```

Problem G : New Country Division

Tehran province is decided to be divided into two provinces: Alborz and New Tehran. There are lots of cities in the old Tehran province that should be distributed between these two new provinces. The criteria for distribution is the safety of inter-province transportation.

Amir is given the map of old Tehran province, including cities and roads and a safety measure for each road. He is supposed to decide for each city to be in either New Tehran or Alborz province. There are two special cities, Tehran and Karaj that should be in New Tehran and Alborz provinces, respectively. Note that it is possible that a city in a province will not be reachable by another city in the same province.

Amir needs to divide the cities in the safest possible manner. The safety of a province division is calculated using a drone. The drone flies between two provinces and crosses all bidirectioanl roads connecting two cities in different provinces exactly once. Each time it passes over a road, the drone calculates the safety of that road and updates the overall safety of the roads it has passed so far. The safety of a road is calculated using a sensor that stores it as a 60-bit binary number. The drone also stores the overall safety in a 60-bit binary number. After passing a road with safety $x_{60}x_{59}\ldots x_1$, the i-th bit of the overall safety is flipped if x_i equals 1 and is left untouched otherwise. The safety of the province division is the overall safety after the drone finishes its journey. Note that initially the overall safety is zero, so if there is no road between two provinces the overall safety does not change during drone's journey which means the safety of that province division is zero.

There are n cities in the old Tehran numbered from 1 to n. Tehran is city number 1 and Karaj is city number n. Your task is to help Amir by writing a program that finds a division of cities with the maximum overall safety.

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains two integers n and m ($2 \leqslant n \leqslant 100$, and $1 \leqslant m \leqslant \frac{n(n-1)}{2}$), which indicate the number of cities and the total number of roads, respectively. The next m lines, each contain three positive integers u, v and r ($1 \leqslant u, v \leqslant n$, $u \neq v$, and $0 \leqslant r < 2^{60}$) which indicate that there is a road between cities u and v with safety r (assume that the safty of all roads has been already calculated by the drone's sensor). It is a guaranteed that there is at most one road between each pair of cities. Input terminates with a line containing "`0 0`" which should not be processed.

Output (Standard Output)

For each test case, output a line containing the overall safety of the division which maximizes it.

Sample Input and Output

Standard Input	Standard Output

3 3
1 2 1
2 3 10
1 3 2
4 1
2 3 47
0 0
	8
	47

Problem H : Column Addition

A multi-digit column addition is a formula on adding two integers written like this:

$$\begin{array}{r} 123 \\ +\quad 456 \\ \hline 579 \end{array}$$

A multi-digit column addition is written on the blackboard, but the sum is not necessarily correct. We can erase any number of the columns so that the addition becomes correct. For example, in the following addition, we can obtain a correct addition by erasing the second and the forth columns.

$$\begin{array}{r} 12127 \\ +\quad 45618 \\ \hline 51825 \end{array} \Rightarrow \begin{array}{r} 117 \\ +\quad 468 \\ \hline 585 \end{array}$$

Your task is to find the minimum number of columns needed to be erased such that the remaining formula becomes a correct addition.

Input (Standard Input)

There are multiple test cases in the input. Each test case starts with a line containing the single integer n, the number of digit columns in the addition ($1 \leqslant n \leqslant 1000$). Each of the next 3 lines contain a string of n digits. The number on the third line is presenting the (not necessarily correct) sum of the numbers in the first and the second line. The input terminates with a line containing "0" which should not be processed.

Output (Standard Output)

For each test case, print a single line containing the minimum number of columns needed to be erased.

Sample Input and Output

Standard Input	Standard Output
3 123 456 579 5 12127 45618 51825 2 24 32 32 5 12299 12299 25598 0	0 2 2 1

Problem I : Cafe Bazaar

Having a heavy load on its servers, Cafe Bazaar keeps a daily log of the traffic to its server for performance optimization and statistical purposes. The IP addresses in the log for each day is stored in a database either in the IP-range format or CIDR format, both explained below.

The IP database software uses a new version of the Internet Protocol (IP) called IPv5. IPv5 provides more IP addresses by extending the number of address bits from 32 in the IPv4 to 40. Precisely, each IP address in the IPv5 is a 40-bits numerical label assigned to each device connected to the Internet. Each IP address can be represented as a sequence of 5 numbers, called bytes, each having a decimal value ranging from 0 to 255. IP addresses are often written in the dot-decimal notation, for example "134.172.16.254.1". The notation consists of five bytes of the IP address expressed in decimal and separated by periods. Note that in this notation the leading zero bytes are not removed. For example, the correct representation of an address with numerical value 0 is "0.0.0.0.0".

Classless Inter-Domain Routing (CIDR) is a way of specifying a range of IP addresses. A CIDR looks like a normal IP address except that it ends with a slash followed by a number. The CIDR y/x in which y is an IP address and x is an integer from 0 to 40 (inclusive), shows a range of addresses whose x leftmost bits are equal to the x leftmost bits of y. The remaining bits are free to be either `0` or `1`. To guarantee a unique representation for each range, the $40 - x$ rightmost bits of y should be equal to `0`.

The IP-range format is another way of specifying a range of IP addresses. A range in this format is represented by its first and last address in dot decimal notation, separated by a dash (-). The first address is not larger than the last address assuming IP addresses are 5-digit numbers in the base 256.

For example, "128.192.168.200.0/32" is a CIDR in which the first 32 bits (4 bytes, "128.192.168.200") are the same for all addresses in the range, and only the last byte can be different. The same range can be represented in the IP-range format by "128.192.168.200.0-128.192.168.200.255".

Due to attracting many customers from all around the world and continuous service for many years, the IP database is getting larger and larger. Thus, Cafe Bazaar plans to reduce the number of entries in the database by representing the existing ranges using the minimum number of CIDRs. The new CIDRs should not include any IP address that does not belong to the IP database. Your task is to solve this challenging problem.

Input (Standard Input)

There are multiple test cases in the input. For each test case, the first line contains an integer n, the number of IP ranges ($1 \leqslant n \leqslant 100$) in the IP database. Each of the next n lines contains an IP range, either in the CIDR format or in the IP-range format. The input terminates with a line containing "`0`" which should not be processed.

Output (Standard Output)

For each test case, print the minimum number of CIDRs which represent the whole IP database; followed by the list of CIDRs in an increasing order of IP values.

Sample Input and Output

Standard Input	Standard Output
2 128.192.168.10.0/32 128.192.168.11.0-128.192.168.11.255 1 128.192.168.200.0-128.192.168.200.10 0	1 128.192.168.10.0/31 3 128.192.168.200.0/37 128.192.168.200.8/39 128.192.168.200.10/40

Problem J : Getting Back Home

Arman has recently moved to a remote village in the country-side. The map of the village is in the form of a tree, i.e. there are exactly $n - 1$ roads connecting n intersections of the village such that for every pair of intersections there is a sequence of roads connecting them.

Every morning Arman goes to his office and late at night he gets back home. It is very dark at night and the village roads do not have any lights so Arman is starting to have problems finding his way back home. There are no signs at intersections and he cannot distinguish them from each other. Even his phone is not signalling on the way, and using GPS is not possible. To solve the problem, he has decided to buy a flashlight. Flashlights have different integer beam distances, and a flashlight with a higher beam range costs more. A flashlight with range d reveals the intersections within a distance at most d around the current intersection. All roads of the village have an equal length of 1.

When Arman leaves office towards home, in every intersection of the path he traverses, he makes a decision as explained below.

1. If he sees home, he just moves towards it.
2. Assume he is at intersection u. Let A be the set of all roads incident to u. Let B be the empty set at the initial time, and $\{e\}$ otherwise where e is the road he just arrived at u through it. Moreover, let C be the set of useless roads incident to u. A road e' incident to u is called useless if all simple paths starting at u and passing through e' have length less than d. If $A - (B \cup C)$ is not empty, one road of this set is chosen randomly. Otherwise, the road e is chosen again.

Arman doesn't care about walking a little bit longer as long as he knows regardless of his random choices, he will eventually reach home after passing through at most 10^9 roads. He wants to purchase the cheapest flashlight for that purpose. Please help him find the appropriate flashlight by which he would be able to reach home.

Input (Standard Input)

There are multiple test cases in the input. For each input the first line contains an integer n, the number of intersections in the village ($2 \leqslant n \leqslant 30,000$). Next $n - 1$ lines each contains two integers a, b meaning that there is a road between intersections a and b ($1 \leqslant a, b \leqslant n$). Arman's home is at intersection 1 and his office is at intersection n. The input terminates with a line containing "0" which should not be processed.

Output (Standard Output)

For each test case, output a single line containing the minimum range d of a flashlight that guarantees Arman can always reach home after passing through at most 10^9 roads regardless of his random choices. If Arman does not need any flashlight, print "0".

Sample Input and Output

Standard Input	Standard Output

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

Problem K : Mars

A new form of life is recently discovered on Mars. Every alien has a DNA, that is a string with an alphabet of only two, rather than four, letters. Hence we can show the DNA of a Mars alien by a binary string. Let s be an alien DNA of length n. There are q regions in the DNA specified as *genes*. A gene located at $[a, b]$ is a substring of the DNA, containing characters from position a to position b, inclusive ($1 \leqslant a \leqslant b \leqslant n$). A gene might overlap with or be inside the other genes.

During the life of a Mars alien, each gene is copied billions of times: a protein binds to the start of the gene, and copies the gene from start to the end. But this process is not error-free, and might produce mutations. In each mutation, a `0` in the gene is copied as `1`, or vice-versa. A mutated copy does not match the gene, but might match a (possibly overlapping) substring in another position of the DNA. For instance, assume that $s = 001011111$ and a gene is located at $[3, 6]$. Hence, the gene string is 1011. A copy of this gene can be 1111, which is mutated at the second letter. Although 1111 does not match the original $[3, 6]$ substring in the DNA, it matches $[5, 8]$. A mutated copy of a gene is called *degenerate* if it does not appear in any place of the whole DNA. Hence, 1010, a copy of the same gene having one mutation at the fourth letter, is degenerate, but 1111 is not.

Your task is to find, for each gene, the minimum number of mutations that can result in a degenerate copy of that gene.

Input (Standard Input)

There are multiple test cases in the input. The first line of each test case contains two integers n and q ($2 \leqslant n \leqslant 10,000$ and $1 \leqslant q \leqslant 1000$). The next line contains a binary string s of length n. Each of the next q lines contains the location $[a, b]$ of a gene, in the form of two space-separated integers a and b ($1 \leqslant a \leqslant b \leqslant n$). The input terminates with a line containing "`0 0`" that should not be processed.

Output (Standard Output)

For each gene, print the minimum number of mutations that can result in a degenerate copy. If no set of mutations applied on a gene can result in a degenerate copy, print "`Impossible`" instead.

Sample Input and Output

Standard Input	Standard Output

4 3
0110
1 2
2 3
1 1
0 0
	1
	2
	Impossible