Problem A

The Chosen Sub Matrix

Time Limit: 1s

From a given *N* x *N* matrix, you should find an *M* x *M* sub matrix which has the least distinct element in it. If there are more than one sub matrixes which have the same number of distinct elements then compare each element in descending order and choose one that has the first highest element. If all of distinct elements of all sub matrixes are the same, chose one with the least row index, and then the least column index. The matrix index starts at 1.

For example, given a 4x4 matrix:

3	9	9	9
3	9	9	2
3	9	9	2
2	5	5	2

Then, the possible sub matrixes of 3x3 are:

3	9	9
3	9	9
3	9	9

S_1

9	9	9
9	9	2
9	9	2

S_2

3	9	9
3	9	9
2	5	5

S_3

9	9	2
9	9	2
5	5	2

S_4

Sub matrix S_1 has 2 distinct elements: 9 and 3;
Sub matrix S_2 has 2 distinct elements: 9 and 2;
Sub matrix S_3 has 4 distinct elements: 9, 5, 3, and 2;
Sub matrix S_4 has 3 distinct elements: 9, 5, and 2.
Sub matrixes are ranked using the rules above and give result as S_1, S_2, S_4, and then S_3.
Which means the chosen sub matrix is S_1.

Input Specification

The first line of each case contains two integers, N ($1 \le N \le 10$) the size of matrix, and M ($1 \le M \le N$) the size of sub matrix to be chosen. In the next N lines, each contains N integers (each separated by a space) that represent the matrix. Each element in the matrix should be between 0 and 9 inclusively.

Output Specification

For each case you should output in a single line, the top-left index (row and column, separated by a single space) of the chosen sub matrix.

Sample Input	Output for Sample Input
4 3 3 9 9 9 3 9 9 2 3 9 9 2 2 5 5 2 10 2 1 5 7 8 2 3 3 3 1 7 2 2 3 6 3 7 3 2 3 1 5 9 3 5 7 0 4 6 9 1 1 0 3 4 2 6 4 3 9 0 7 4 9 9 5 4 6 2 1 5 5 6 9 9 6 6 3 8 0 8 4 3 3 5 2 1 7 6 4 1 6 5 9 5 0 3 1 8 8 6 2 2 2 8 0 1 3 5 9 0 3 6 4 2 3 3 0 2 0 0	1 1 5 3

Problem B

Avoiding Financial Nightmare

Time Limit: 1s

Nowadays, getting a loan from a bank or financial company has become very popular, either it's for commercial or personal purposes. If you are a well-managed on your expenses, having a loan from a bank or using credit cards for your expenses could be a good help, otherwise it could be your worst financial nightmare. By considering the risks of paying bills, our professor has decided to get a loan to buy a new house on a pleasant city in Indonesia.

Our professor has considered 3 main variables that will affect his monthly bills:

- `Principal`, the remaining amount of the loan
- `Period`, the number of months to pay off the loan
- `Rate`, the monthly interest rate

To help their customer, bank or financial company normally offers a fixed amount payment system. Every month the customer should pay a fixed amount of money which consists of two kinds of payment:

1. Interest Payment.
   `Interest Payment = Rate x Remaining Principal`.
   The amount of interest payment should be rounded up to the higher nearest integer.

2. Principal Payment.
   `Principal Payment = Total Payment - Interest Payment`.
   The previous Principal is to be subtracted with the current Principal Payment to get the current Principal.

The total monthly payment should be calculated in some way so that the total monthly payment to be paid spread evenly each month, and at the end of the period the remaining Principal is zero or a negative amount nearest to zero (if it's not possible to reach zero).

For example, let the professor get a loan in the amount of $42,000 with 5% interest rate per month that should be paid in 5 months.

Term	Payment			Principal
	Total	Interest Payment	Principal Payment	
-	-	-	-	42,000
1	9,701	2,100	7,601	34,399
2	9,701	1,720	7,981	26,418
3	9,701	1,321	8,380	18,038
4	9,701	902	8,799	9,239
5	9,701	462	9,239	0

1st term: He should pay $9,701 ($2,100 for interest payment, and $7,601 for principal payment)
Interest Payment : 5% x $42,000 = $2,100
Principal Payment : $9,701 - $2,100 = $7,601

Current Principal : $42,000 - $7,601 = $34,399

2nd term: He pays $9,701 ($1,702 for interest payment, and $7,981 for principal payment)
Interest Payment : 5% x $34,399 = $1,720
Principal Payment : $9,701 - $1,720 = $6,902
Current Principal : $34,399 - $6,902 = $26,418, and so on.

Unfortunately, the professor is terrible with financial stuffs. We don't want him to end up broke, do we? So, let us help him with the calculation on how much money he should spend to pay his monthly bills on the loan. In that way, the professor will be able to buy his new house and who knows that someday we might be invited to visit his house in return to our help.

Input Specification

The input contains multiple cases. Each case consists of three integers respectively, n ($1 \le n \le 10{,}000{,}000$) the initial principal, m ($1 \le m \le 100$) the period, and r ($0 \le r \le 100$) the percent rate.

Output Specification

For each case, you should output in a single line the total monthly payment should be made to satisfy the condition.

Sample Input	Output for Sample Input
42000 5 5 100000 10 10	9701 16275

Problem C

No Pause Telegraph

Time Limit: 1s

Several hundreds years ago in a country, there was a war between the colonialist and the military of the country. In the war, telegraph machine was used to communicate between the country and its allied cities around the country. The Morse code was used to send messages through telegraph machine. In Morse, letters are represented only by "." (dot) and "–" (dash). Between letters there is a pause to avoid mistranslation of letters.

In that era with the popularity of telegraph machine, there was wariness that the enemy could read the messages, if the messages were accidentally sent to them. Being aware of the threat, a machine was created to encode the messages. It was called No Pause Telegraph. This new telegraph machine was similar to the regular machine, only that the machine encoded the messages with no pause between letters.

After doing some researches on old relics, our archeologist has analyzed the code being used successfully. The archeologist so far has found 7 letters that been used on the found relics, those are:

```
A      .--    dot dash dash
B      -.     dash dot
C      ---    dash dash dash
D      ..     dot dot
E      --..   dash dash dot dot
F      --.-   dash dash dot dash
G      .-.    dot dash dot
```

Your task is to translate several messages encoded by the machine on the remaining relics. You should be careful that any ambiguousity may arise when translating those messages. For example: If the code for X were `.` and the code for Y were `..`, then two dots in a row could be either XX or Y. For this particular reason, you should output only the least alphabetical solution.

Input Specification

Each line of input is a code encoded by No Pause Telegraph. The line will consists of only characters of `.` (dot(s)) and `-` (dash(es)).

Output Specification

For each line of input, print the least alphabetical possible message in one line. If there is no possible message, print "`could not be translated`" (without quotes).

Sample Input	Output for Sample Input
`.---.-` `.---.---..--..--.-.-.`	`could not be translated` `ABCDEFG`

Problem D

Burger, French Fries, Soft Drink

Time Limit: 1s

Anna works as a waitress at a popular fast-food restaurant in Indonesia, "Om Burger". Her job is much easier than any waiters/waitresses at other fast-food restaurants, because Om Burger owns an automatic fast-food machine.

Om Burger serves only a single combo-menu package named Paket Uenak, which consists of three items: 1 Burger (B), 1 French Fries (F), and 1 Soft Drink (S). Each customer would be given a card to write the number of Paket Uenak packages to be ordered. The card should be given back to Anna who will insert it into the machine. The machine will prepare all items of each package one by one according to the written number on the card, but not necessary in any particular order of item. For example, if the number 2 (means 2 Paket Uenak packages are ordered by a customer) is written on the card, the machine will prepare items in the sequence of BBFFSS, or any of its permutation (BFSBFS, BBFSFS, etc). There could be more than one card to be processed by the machine at a time. However the machine will process all cards sequentially. That means it will not proceed to the next card before finishing the current card.

One day, some customers are queuing to put their orders. While they make a quite long queue, it gives Anna an idea. Instead of inserting the card one by one, she initiatively inserts more than one card at a time. Brilliant, isn't she? Well, let's see what happens next. The machine does work well, but now she doesn't have any idea which item belongs to which card, because the machine doesn't give any separations to separate packages from different cards.

Fortunately, Anna still remembers the number of cards inserted and its sequence (which card belongs to whom). However, she doesn't remember the number of Paket Uenak packages written on each card.

For example, if there are two cards inserted into the machine:
Prepared : B B F S S F S F B B F S
Possibility-1 : (B B F S S F) (S F B B F S), the first customer ordered 2 Paket Uenak packages, and the second customer ordered 2 Paket Uenak packages.
Possibility-2 : (B B F S S F S F B) (B F S), the first customer ordered 3 Paket Uenak packages, and the second customer ordered 1 Paket Uenak package.

No need to tell, that now she's apologizing and questioning all customers for what they have been ordering again. But this problem has aroused her curiosity as she is also a student in Computer Science. Help Anna to find how many possible order arrangements could be found from any given condition(s).

Input Specification
Each line of input contains an integer N ($1 < N < 30$) denoting the number of card(s) inserted, and a string which contains character(s) of B, F, and S (no spaces) denoting the order of items prepared by the machine. You may assume that the length of each string is less than 100.

Output Specification

For each line of input, print the number of possible order arrangements could be found. If there is no possible order arrangement, output "`Impossible`" (without quotes).

Sample Input	Output for Sample Input
2 BBFSSFSFBBFS 2 BBFSSFSFBB	2 Impossible

Problem E

Taxi!

Time Limit: 1s

Taxi Driver : "Where to go, sir?"
Passenger : "Kemanggisan, Binus Campus, please."
Taxi Driver : "Okay, and which way to go?"
Passenger : "Just take the fastest one."

This kind of conversation usually happens when we take a taxi. Many people think that the fastest way would mean the cheapest one as well. But it's not always true! Sometimes the fastest way could be more expensive than the slower one and vice versa, some factors are: toll fee, longer road without traffic jam, etc.

We will model this kind of situation on this problem. The city has n intersections and m bidirectional roads connecting pairs of intersections. Each road will cost you a certain time and money (taxi fare). Write a program to find the minimum time to take you to your destination however the travel expenses must not exceed your money.

Input Specification
The first line of each case contains two integers, n ($1 \leq n \leq 50$) intersections and m roads. The intersections are numbered from 0 to n-1. The next m lines will each contains four integers: u, v, t, and c ($1 \leq t,c \leq 100$), means that there is a road from intersection u to intersection v and vice versa which will cost you t minute(s) and c Rupiah(s). The last line of each case will contains three integers: s, d, and r ($1 \leq r \leq 100$), means that you want to go to d (your destination point) from s (your starting point) while you only have r Rupiah(s).

Output Specification
For each case, output in a single line the minimum time to reach your destination while the total cost doesn't exceed your money.

Sample Input	Output for Sample Input
3 3 0 2 16 19 0 1 9 12 1 2 5 13 0 2 20 4 5 0 1 2 11 1 2 7 27 2 3 4 10 0 3 6 25 3 1 5 12 0 2 35	16 10