Problem C

Optimal Parking

When shopping on Long Street, Michael usually parks his car at some random location, and then walks to the stores he needs. Can you help Michael choose a place to park which minimises the distance he needs to walk on his shopping round?

Long Street is a straight line, where all positions are integer. You pay for parking in a specific slot, which is an integer position on Long Street. Michael does not want to pay for more than one parking though. He is very strong, and does not mind carrying all the bags around.

Input specifications

The first line of input gives the number of test cases, $1 \leq t \leq 100$. There are two lines for each test case. The first gives the number of stores Michael wants to visit, $1 \leq n \leq 20$, and the second gives their n integer positions on Long Street, $0 \leq x_i \leq 99$.

Output specifications

Output for each test case a line with the minimal distance Michael must walk given optimal parking.

Sample input

```
2
4
24 13 89 37
6
7 30 41 14 39 42
```

Output for sample input

```
152
70
```

A

Problem E

Circle of Debt

The three friends Alice, Bob, and Cynthia always seem to get in situations where there are debts to be cleared among themselves. Of course, this is the "price" of hanging out a lot: it only takes a few resturant visits, movies, and drink rounds to get an unsettled balance. So when they meet as usual every Friday afternoon they begin their evening by clearing last week's debts. To satisfy their mathematically inclined minds they prefer clearing their debts using as little money transaction as possible, i.e. by exchanging as few bank notes and coins as necessary. To their surprise, this can sometimes by harder than it sounds. Suppose that Alice owes Bob 10 crowns and this is the three friends' only uncleared debt, and Alice has a 50 crown note but nothing smaller, Bob has three 10 crown coins and ten 1 crown coins, and Cynthia has three 20 crown notes. The best way to clear the debt is for Alice to give her 50 crown note to Cynthia, Cynthia to give two 20 crown notes to Alice and one to Bob, and Bob to give one 10 crown coin to Cynthia, involving a total of only five notes/coins changing owners. Compare this to the straightforward solution of Alice giving her 50 crown note to Bob and getting Bob's three 10 crown notes and all his 1 crown coins for a total of fourteen notes/coins being exchanged!

Input specifications

On the first line of input is a single positive integer, $1 \leq t \leq 50$, specifying the number of test cases to follow. Each test case begins with three integers $ab, bc, ca \leq 1000$ on a line of itself. ab is the amount Alice owes Bob (negative if it is Bob who owes Alice money), bc the amount Bob owes Cynthia (negative if it is Cynthia who is in debt to Bob), and ca the amount Cynthia owes Alice (negative if it is Alice who owes Cynthia).

Next follow three lines each with six non-negative integers $a_{100}, a_{50}, a_{20}, a_{10}, a_5, a_1$, $b_{100}, \ldots, b_1$, and $c_{100}, \ldots c_1$, respectively, where a_{100} is the number of 100 crown notes Alice got, a_{50} is the number of her 50 crown notes, and so on. Likewise, $b_{100}, \ldots, b_1$ is the amount of notes/coins of different value Bob got, and $c_{100}, \ldots, c_1$ describes Cynthia's money. Each of them has at most 30 coins (i.e. $a_{10}+a_5+a_1, b_{10}+b_5+b_1$, and $c_{10}+c_5+c_1$ are all less than or equal to 30) and the total amount of all their money together (Alice's plus Bob's plus Cynthia's) is always less than 1000 crowns.

Output specifications

For each test case there should be one line of output containing the minimum number of bank notes and coins needed to settle the balance. If it is not possible at all, output the string "`impossible`".

Sample input

```
3
10 0 0
0 1 0 0 0 0
0 0 0 3 0 10
0 0 3 0 0 0
-10 -10 -10
0 0 0 0 0 0
0 0 0 0 0 0
0 0 0 0 0 0
-10 10 10
3 0 0 0 2 0
0 2 0 0 0 1
0 0 1 1 0 3
```

Output for sample input

```
5
0
impossible
```

C

C Cat vs. Dog

The latest reality show has hit the TV: "Cat vs. Dog". In this show, a bunch of cats and dogs compete for the very prestigious BEST PET EVER title. In each episode, the cats and dogs get to show themselves off, after which the viewers vote on which pets should stay and which should be forced to leave the show.

Each viewer gets to cast a vote on two things: one pet which should be kept on the show, and one pet which should be thrown out. Also, based on the universal fact that everyone is either a cat lover (i.e. a dog hater) or a dog lover (i.e. a cat hater), it has been decided that each vote must name exactly one cat and exactly one dog.

Ingenious as they are, the producers have decided to use an advancement procedure which guarantees that as many viewers as possible will continue watching the show: the pets that get to stay will be chosen so as to maximize the number of viewers who get both their opinions satisfied. Write a program to calculate this maximum number of viewers.

Input

On the first line one positive number: the number of testcases, at most 100. After that per testcase:

- One line with three integers c, d, v ($1 \leq c, d \leq 100$ and $0 \leq v \leq 500$): the number of cats, dogs, and voters.
- v lines with two pet identifiers each. The first is the pet that this voter wants to keep, the second is the pet that this voter wants to throw out. A pet identifier starts with one of the characters '`C`' or '`D`', indicating whether the pet is a cat or dog, respectively. The remaining part of the identifier is an integer giving the number of the pet (between 1 and c for cats, and between 1 and d for dogs). So for instance, "`D42`" indicates dog number 42.

Output

Per testcase:

- One line with the maximum possible number of satisfied voters for the show.

Sample in- and output

Input	Output
2 1 1 2 C1 D1 D1 C1 1 2 4 C1 D1 C1 D1 C1 D2 D2 C1	1 3

C

D

I White Water Rafting

You have been hired by a big theme park to design a new attraction: a white water rafting ride. You already designed the track; it is a round trip that is described by an inner and an outer polygon. The space in between the two polygons is the track.

You still need to design the rafts, however. It has been decided that they should be circular, so that they can spin freely along the track and increase the fun and excitement of the ride. Besides that, they should be as big as possible to fit the maximum number of people, but they can't be too big, for otherwise they would get stuck somewhere on the track.

What is the maximum radius of the rafts so that they can complete the track?

Input

On the first line one positive number: the number of testcases, at most 100. After that per testcase:

- One line with an integer n_i ($3 \leq n_i \leq 100$): the number of points of the inner polygon.
- n_i lines with two integers each: the coordinates of the points of the inner polygon in consecutive order.
- One line with an integer n_o ($3 \leq n_o \leq 100$): the number of points of the outer polygon.
- n_o lines with two integers each: the coordinates of the points of the outer polygon in consecutive order.

All coordinates have absolute value no larger than $1\,000$. The points of the polygons can be given in either clockwise or counterclockwise order and the two polygons do not intersect or touch themselves or each other. The outer polygon encloses the inner polygon.

Output

Per testcase:

- One line with a floating point number: the maximal radius of the white water rafts. This number must have a relative or absolute error less than 10^{-6}.

Sample in- and output

Input	Output
2 4 -5 -5 5 -5 5 5 -5 5 4 -10 -10 -10 10 10 10 10 -10 3 0 0 1 0 1 1 5 3 -3 3 3 -4 2 -1 -1 -2 -2	2.5 0.70710678

E: Minesweeper

Minesweeper is a game played on a **R** x **C** rectangular board. Some of the cells contain mines, and others are empty. For each empty cell, calculate the number of its adjacent cells that contain mines. Two cells are adjacent if they share a common edge or point. This means that each cell has a maximum of 8 neighbors (up, down, left, right, four diagonals).

Input

There will be multiple test cases. The first line of each test case will have two integers, **R** and **C** (**1 ≤ R,C ≤ 100**), indicating the number of rows and columns of the board. The next **R** lines each contain exactly **C** characters. Each character is either a '*****' (asterisk) indicating a mine, or a '**.**' (period) indicating an empty cell. The last data set is followed by a line containing two **0**'s.

Output

Print each board on **R** lines with **C** characters per line, and replace every '**.**' with the appropriate digit indicating the number of adjacent cells that contain mines. Leave the '*****' cells intact. Do not print any whitespace between cells. Do not print any blank lines between answers.

Sample Input

```
3 2
..
.*
..
5 5
*.*.*
..*..
*****
.....
..**.
0 0
```

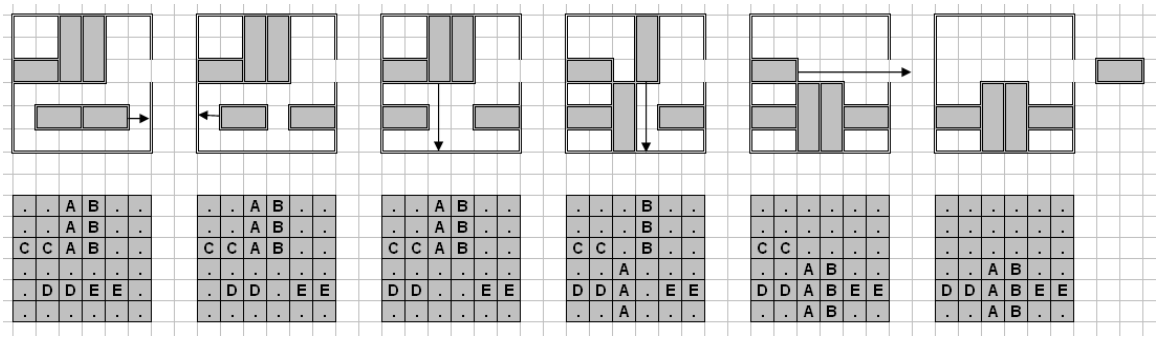
Sample output

```
11
1*
11
*3*3*
36*63
*****
24553
01**1
```

E

A: Block Game

Bud bought a new board game. He is hooked. He has been playing it over and over again, and he thinks can solve any board with the minimum number of moves, but he is uncertain. He wants you to write a program to calculate the minimum number of moves required to solve different boards, so that he can double check his answers.



You are given a 6x6 board, and a set of 2x1 or 3x1 (vertical) or 1x2 or 1x3 (horizontal) pieces. You can slide the horizontal pieces horizontally only, and the vertical pieces vertically only. You may slide a piece if there are no other pieces, nor walls, obstructing its path.

There will be one special 1x2 horizontal piece. There will also be a gap in the wall, on the right side, on the same row as the special piece, that only the special piece can fit through.The goal of the game is to get that one special horizontal piece out of the gap on the right side.

Sliding a piece any number of squares is considered one move. (i.e. sliding a piece horizontally one square is one move, and sliding it two squares at once is also considered one move).

Input

There will be several test cases. Each test case will begin with a line with a single capital letter, indicating the special piece which must be moved off of the board. The next 6 lines will consist of 6 characters each. These characters will either be a '**.**' (period), indicating an empty square, or a capital letter, indicating part of a piece. The letters are guaranteed to form pieces that are 1x2, 1x3, 2x1 or 3x1, and no letter will be used to represent more than one piece on any given board. The letter indicating the special piece is guaranteed to correspond to a 1x2 piece somewhere on the board. The end of data is indicated by a single '*****' (asterisk) on its own line.

Output

For each test case, print a single integer, indicating the smallest number of moves necessary to remove the given special piece, or `-1` if it isn't possible. Print each integer on its own line. There should be no blank lines between answers.

Sample Input

```
C
..AB..
..AB..
CCAB..
......
.DDEE.
......
A
......
......
......
......
AA....
......
Z
.ZZ..X
.....X
.....X
.....Y
.....Y
.....Y
*
```

Sample Output

```
5
1
-1
```

Problem L — limit 5 seconds

Tongues

Gandalf's writings have long been available for study, but no one has yet figured out what language they are written in. Recently, due to programming work by a hacker known only by the code name ROT13, it has been discovered that Gandalf used nothing but a simple letter substitution scheme, and further, that it is its own inverse—the same operation scrambles the message as unscrambles it.

This operation is performed by replacing vowels in the sequence

(a i y e o u)

with the vowel three advanced, cyclicly, while preserving case (i.e., lower or upper). Similarly, consonants are replaced from the sequence

(b k x z n h d c w g p v j q t s r l m f)

by advancing ten letters. So for instance the phrase

One ring to rule them all.

translates to

Ita dotf ni dyca nsaw ecc.

The fascinating thing about this transformation is that the resulting language yields pronounceable words.

For this problem, you will write code to translate Gandalf's manuscripts into plain text.

Input

The input file will contain multiple test cases. Each test case consists of a single line containing up to 100 characters, representing some text written by Gandalf. All characters will be plain ASCII, in the range space (32) to tilde (126), plus a newline terminating each line. The end of the input is denoted by the end-of-file.

Output

For each input test case, print its translation into plaintext. The output should contain exactly the same number of lines and characters as the input.

Sample Input	**Sample Output**
`Ita dotf ni dyca nsaw ecc.`	`One ring to rule them all.`

G

Problem B — limit 5 seconds

Magic Multiple

The Elvish races of Middle Earth believed that certain numbers were more significant than others. When using a particular quantity n of metal to forge a particular sword, they believed that sword would be most powerful if the thickness k were chosen according to the following rule:

Given a nonnegative integer n, what is the smallest k such that the decimal representations of the integers in the sequence:

$$n, \quad 2n, \quad 3n, \quad 4n, \quad 5n, \quad \ldots, \quad kn$$

contain all ten digits (0 through 9) at least once?

Lord Elrond of Rivendell has commissioned you with the task to develop an algorithm to find the optimal thickness (k) for any given quantity of metal (n).

Input

Input will consist of a single integer n per line. The end of input will be signaled by end of file. The input integer will be between 1 and 200,000,000, inclusive.

Output

The output will consist of a single integer per line, indicating the value of k needed such that every digit from 0 through 9 is seen at least once.

Sample Input	Sample Output
1 10 123456789 3141592	10 9 3 5

H

I

D Bad Wiring

Problem

The ninja Ryu has infiltrated the Shadow Clan fortress and finds himself in a long hallway. Although ninjas are excellent fighters, they primarily rely on stealth to complete their missions. However, many lights are turned on in the hallway, and this way it will not take long before Ryu is spotted by a guard. To remain unseen, Ryu will need to turn off all the lights as quickly as possible.

The hallway contains a sequence of n lights $L_1 \dots L_n$. Some of these lights are turned on. Destroying the lights with his shurikens would be too loud, so he needs to turn them off the old-fashioned way, using light switches. Luckily, there is a switch box nearby with a light switch S_i for every light L_i. However, after trying one of the switches, he notices something funny. When he flips the switch S_i, it does not only turn on/off light L_i, but also some of the neighboring lights. Ryu notices that there is a parameter D such that flipping switch S_i turns on/off all the lights $L_{i-D} \dots L_{i+D}$, if they exist[2]. Turning on or off lights can attract the attention of the guards, so Ryu would like to turn off all the lights with the minimum number of times flipping a switch. Can you help him out?

Input

The first line of the input contains a single number: the number of test cases to follow. Each test case has the following format:

- One line with two integers n ($1 \leq n \leq 100$) and D ($0 \leq D \leq 15$): the number of lights and the parameter mentioned above.
- One line with n integers. The i^{th} integer describes the current state of light L_i, where 0 means off and 1 means on.

Output

For every test case in the input, the output should contain one integer on a single line: the minimum number of times Ryu needs to flip a switch to turn off all the lights. If it is impossible to turn off all the lights, then output the string "`impossible`" instead.

Examples

In the first example below, flipping switch S_4 followed by S_7 will turn off all the lights.

Input	Output
2 7 3 1 1 1 0 0 0 0 5 1 1 0 1 0 1	2 3

[2]This means that S_1 turns on/off all the lights $L_1 \dots L_{D+1}$ and S_n turns on/off all the lights $L_{n-D} \dots L_n$. Of course, if $D \geq n$, then L_{D+1} and L_{n-D} will not exist either.

I

J

I Parking Ships

Problem

Life on the great oceans has been good for captain Blackbeard and his fellow pirates. They have gathered so many treasures, that each of them is able to buy a house on their favorite island. The houses on this island are all placed in a long row along the beach line of the island. Next to a house, every pirate is also able to buy his own ship to do their own bit of plundering. However, this causes a whole new kind of problem.

Along the beach line there is a long pier where every pirate can park his ship. Although there is enough space along the pier for all the ships, not every pirate will be able to park in front of his house. A pirate is happy with his parking space if some part of the parking space is in front of the center of his house. Captain Blackbeard has been given the difficult task of assigning the parking spaces to the pirates. A parking space for a pirate i is a range $[a_i, b_i]$ $(a_i, b_i \in \mathbb{R})$ along the pier such that $l_i \leq b_i - a_i$, where l_i is the length of the ship of pirate i. Thus, pirate i is happy if $a_i \leq x_i \leq b_i$, where x_i is the center of the house of pirate i. Clearly, parking spaces of different pirates must be interior disjoint (the ends of ranges can coincide).

Above all, the captain wants a good parking space for himself, so he gives himself the parking space such that the center of his ship is aligned with the center of his house. Next to that, he wants to make as many pirates happy as possible. Can you help him out?

Input

The first line of the input contains a single number: the number of test cases to follow. Each test case has the following format:

- One line with one integer n $(1 \leq n \leq 1,000)$: the number of pirates including the captain.
- n lines with on each line two integers x_i $(-10^9 \leq x_i \leq 10^9)$ and l_i $(1 \leq l_i \leq 10^9)$: the center of the house of the i^{th} pirate and the total length of his ship, respectively. The first pirate in the input is always the captain.

Output

For every test case in the input, the output should contain one integer on a single line: the maximum number of happy pirates using an optimal assignment of the parking spaces. This number includes the captain himself. You can assume that the space along the pier is unbounded in both directions.

J

Examples

Input	Output
2 5 0 6 -5 2 -4 1 4 2 5 3 4 0 4 -5 4 3 4 5 3	5 3