A. Jumps

Time limit: 2s Memory limit: 256 MB

A frog lives in a one-dimensional world in the point with the coordinate 0. He needs to get to the point with the coordinate *x*. For some reason he cannot make jumps of arbitrary length, and can jump only by a_1, ..., a_n in any direction. Is he able to reach *x*?

Input

The first line contains two integers *n* and *x* separated by a space ($1 \le n \le 200000$, $-10^9 \le x \le 10^9$) — the number of variants of jump length and the coordinate of the point to reach.

The second line contains *n* integers a_i separated by spaces ($1 \le a_i \le 10^9$) — the lengths of jumps the frog can make.

Output

Output «YES» (without quotes), if the frog can reach the point *x*, otherwise output «NO» (without quotes).

Examples

input

```
3 17
3 5 4
```

output

```
YES
```

input

```
4 5
10 20 30 40
```

output

```
NO
```

X Samara Regional Intercollegiate Programming Contest

B. Killing everything

Time limit: 4s Memory limit: 64 MB

There are many enemies in the world such as red coders and hackers. You are trying eliminate everybody. Everybody is standing on a road, which is separated into 10^9 sections. The sections are numbered $1, 2, 3, 4, \ldots 10^9$ from west to east. You want to kill N enemies. The i^{th} enemy will be standing on the section A_i. In order to kill the enemies, you prepared P small bombs and Q large bombs. You can choose a positive integer w as a parameter for energy consumption. Then, a small bomb can kill all enemies in at most w consecutive sections, and a large bomb can kill all enemies of at most $2w$ consecutive sections.

Enemies can be killed by more than one bomb. You want to kill all enemies. Since it is expected that many civilians will walk down that road, for the sake of safety, you have to fix the positions of the bombs and minimize the value of w.

So you decided to Write a program that, given information of the enemies and the number of bombs, determine the minimum value of w so all enemies can be killed.

Input

The input consists of several test cases, the first line contains the number of test cases T. For each test case: The first line of input contains three space separated integers N, P, Q $(1 \le N \le 2000, 0 \le P \le 10^5, 0 \le Q \le 10^5)$, where N is the number of the enemies, P is the number of small bombs, and Q is the number of large bombs.

The i^{th} line $(1 \le i \le N)$ of the following N lines contains an integer A_i, the section where the i^{th} enemy will be standing.

Output

Output: For each test cases print the solution of the problem on a new line.

Examples

input

```
1
3 1 1
2
11
17
```

output

```
4
```

Note

In the sample test case you have 3 enemies at positions: $2, 11, 17$.

For $w = 4$, one possible solution is to throw one small bomb on segment $1 - 4$, and one large bomb on segment $11 - 18$. This configuration will kill all three enemies.

There is no configuration with $w < 4$ that can kill them all.

2015 AlBaath Collegiate Programming Contest

C. Matts Trip

Time limit: 8s Memory limit: 1536 MB

Matt finds himself in a desert with N ($2 \le N \le 10$) oases, each of which may have food, water, and/or a palm tree. If oasis i has food, then $F_i = 1$ - otherwise, $F_i = 0$. Similarly, $W_i = 1$ if and only if oasis i has water, and $P_i = 1$ if and only if it has a palm tree. These 3 values are completely independent of one another.

Some pairs of these oases are connected by desert paths, which each take 1 hour to traverse. There are M ($0 \le M \le 45$) such paths, with path i connecting distinct oases A_i and B_i in both directions ($1 \le A_i, B_i \le N$). No pair of oases is directly connected by more than one path, and it's not guaranteed that all oases are connected by some system of paths.

Matt starts at an oasis S, and wants to end up at a different oasis E ($1 \le S, E \le N$).

Both of these oases are quite nice - it's guaranteed that $F_S = W_S = P_S = F_E = W_E = P_E = 1$.

Since he's in a hurry to get out of the desert, he wants to travel there in at most H ($1 \le H \le 10^9$) hours.

However, he can only survive for up to MF hours at a time without food, and up to MW hours at a time without water ($1 \le MF, MW \le 4$). For example, if MF = 1 and MW = 2, then every single oasis he visits along the way must have food (as he would otherwise spend more than 1 hour without it), and he cannot visit 2 or more oases without water in a row.

Since Matt is a computer scientist, before actually going anywhere, he's interested in the number of different paths he can take that will get him from oasis S to oasis E alive in at most H hours.

Note that there may be no such paths.

Being a computer scientist, he of course only cares about this number modulo ($10^9 + 7$).

Input

Line 1: 7 integers, N, M, H, S, E, MF, and MW

Next N lines: 3 integers, F_i, W_i, and P_i, for i = 1..N

Next M lines: 2 integers, A_i and B_i, for i = 1..M

Output

1 integer, the number of different valid paths, modulo ($10^9 + 7$)

Example 1

Input:

3 3 3 1 2 1 4

1 1 1

1 1 1

0 1 0

1 2

2 3

1 3

Output:

2

Explanation:

The two possible paths, described in terms of oases visited, are 1 → 2 and 1 → 2 → 1 → 2. Matt can never go to oasis 3, as it doesn't contain food, which he can't survive without for more than 1 hour. The path 1 → 2 → 1 → 2 → 1 → 2 is not valid, as it would take 5 hours rather than at most 3.

Note that oasis 3 is the only oasis without a palm tree.

Example 2

Input:

5 5 3 3 2 3 2

1 0 0

1 1 1

1 1 1

0 0 1

0 1 0

1 2

1 3

1 4

3 4

4 2

Output:

2

Explanation:

The two possible paths are 3 → 1 → 2 and 3 → 4 → 2.

This time, oases 1 and 5 are lacking in palm trees.

Own problem

D. Print Mix Strings

Time limit: 2s Memory limit: 64 MB

While I was working in the company, the internet broke. I didn't have anything to do without internet, so I decided to write this ACM problem.

We need to mix between two strings and also we should keep the same order for both strings.

Example: if we have s_1 = "*ab*" and s_2 = "*cd*", we can generate six strings:

abcd acbd

acdb cdab

cadb cabd

thank you for your help in ”Count Mix Strings” problem. now I can calculate the complexity and make input and output files. You will be given the two strings and you should print all the distinct strings that could be generated in the alphabetical order.

Input

Your program will be tested on one or more test cases. The first line of the input will be a single integer T, the number of test cases ($1 \le T \le 100$).

Followed by the test cases, each test case is on one line. it contains two strings s_1 and s_2, both strings consist of at least 1 and at most 8 lower case English letters (from ‘*a*’ to ‘*z*’).

Output

For each test case, print all the distinct strings that could be generated in the alphabetical order.

print a blank line after each test case.

Examples

input

```
2
a aa
ab cd
```

output

```
aaa

abcd
acbd
acdb
cabd
cadb
cdab

```

2015 Damascus Collegiate Programming Contest (DCPC 2015)

E. Bayan Bus

Time limit: 2s Memory limit: 256 MB

The final round of Bayan Programming Contest will be held in Tehran, and the participants will be carried around with a yellow bus. The bus has 34 passenger seats: 4 seats in the last row and 3 seats in remaining rows.

The event coordinator has a list of *k* participants who should be picked up at the airport. When a participant gets on the bus, he will sit in the last row with an empty seat. If there is more than one empty seat in that row, he will take the leftmost one.

In order to keep track of the people who are on the bus, the event coordinator needs a figure showing which seats are going to be taken by *k* participants. Your task is to draw the figure representing occupied seats.

Input

The only line of input contains integer *k*, ($0 \le k \le 34$), denoting the number of participants.

Output

Print the figure of a bus with *k* passengers as described in sample tests. Character '#' denotes an empty seat, while 'O' denotes a taken seat. 'D' is the bus driver and other characters in the output are for the purpose of beautifying the figure. Strictly follow the sample test cases output format. Print exactly six lines. Do not output extra space or other characters.

Examples

input

```
9
```

output

```
+------------------------+
|O.O.O.#.#.#.#.#.#.#.#.|D|)
|O.O.O.#.#.#.#.#.#.#.#.|.|
|O.......................|
|O.O.#.#.#.#.#.#.#.#.#.|.|)
+------------------------+
```

input

```
20
```

output

```
+------------------------+
|O.O.O.O.O.O.O.#.#.#.#.|D|)
|O.O.O.O.O.O.#.#.#.#.#.|.|
|O.......................|
|O.O.O.O.O.O.#.#.#.#.#.|.|)
+------------------------+
```

F. Tesouro

Time limit: 1.865s Memory limit: 1536 MB

Autor: Tiago Mota

Um famoso explorador descobriu uma região repleta de labirintos, dentro dos quais sabe-se que há um valioso tesouro. Estes labirintos, porém, têm algumas peculiaridades.

Cada labirinto é composto por salas, tendo cada sala uma parte (não necessariamente igual) do tesouro, a qual pode ser totalmente coletada ao se visitar a sala.

As salas são ligadas entre si através de corredores. Cada corredor liga uma sala origem a uma sala destino e possui duas portas, uma em cada extremidade. Inicialmente, a porta da sala de origem está aberta, enquanto a porta da sala de destino está fechada. Ao passar pela porta da sala de origem, esta porta é fechada, e a porta da sala de destino é aberta. Após percorrer o corredor e passar pela porta da sala de destino, esta porta é fechada. Uma vez que isso ocorreu, as portas permanecem nessa posição para sempre, de modo que o corredor só pode ser utilizado uma vez, e apenas da sala de origem para a sala de destino.

Além disso, há uma sala especial (sem tesouro) que serve de entrada para o labirinto, da qual partem corredores (como os descritos anteriormente) para todas as outras salas. Da mesma forma, há uma sala especial que serve de saída para o labirinto, para a qual convergem corredores vindos de todas as outras salas. As portas de entrada para a sala de entrada e de saída da sala de saída são como as portas dos corredores, fechando-se ao serem ultrapassadas. Desse modo, só se pode entrar e sair de um labirinto uma única vez.

Mais ainda, o labirinto é construído de tal forma que, uma vez tendo saído de uma sala, não há corredores que levem de volta a esta mesma sala, não sendo possível, então, visitá-la mais de uma vez. Ou seja, não há nenhuma seqüência de corredores que podemos percorrer tal que uma sala é visitada mais de uma vez.

Levando-se em conta, então, as características desses labirintos, é pedido que você faça um programa que, dados a quantidade de tesouro encontrada em cada sala e os corredores que as ligam, calcule o máximo de tesouro que pode ser coletado de um determinado labirinto.

Entrada

A entrada é composta por vários casos de teste. Cada caso de teste corresponde a um labirinto, descrevendo suas salas e seus corredores.

A primeira linha de um caso de teste contém um único número inteiro, n ($1 <= n <= 1000$), correspondendo ao número de salas do labirinto, numeradas de 1 a n . A seguir, há n linhas, tendo a i -ésima destas linhas informações sobre a sala de número i .

As informações sobre uma sala estão separadas por espaços e na seguinte ordem: um número inteiro q ($0 <= q <= 10000$), a quantidade de tesouro encontrada na sala; um número inteiro p ($0 <= p < n$), a quantidade de corredores que têm origem na sala i ; e p números inteiros, cada um entre 1 e n , com os números das salas de destino dos corredores que têm origem na sala sendo descrita.

Não são descritas as salas de entrada e de saída do labirinto e os corredores que as ligam às outras salas, uma vez que são estruturas especiais do labirinto que dispensam descrição.

A entrada termina quando n = 0 .

Saída

Para cada caso de teste, seu programa deve imprimir, em uma única linha, o máximo de tesouro que pode ser coletado das salas do labirinto.

Exemplo

```
Entrada:
3
7 1 3
10 0
8 0
3
7 1 3
20 0
8 0
6
100 0
300 0
100 1 1
0 1 2
100 1 3
1 1 4
0

Saída:
15
20
301
```

Primeira prova de TEP - 2008/1 - UFRJ

G. Nineteen

Time limit: 1s Memory limit: 256 MB

Alice likes word "nineteen" very much. She has a string *s* and wants the string to contain as many such words as possible. For that reason she can rearrange the letters of the string.

For example, if she has string "xiineteenppnnnewtnee", she can get string "x**nineteen**pp**nineteen**w", containing (the occurrences marked) two such words. More formally, word "nineteen" occurs in the string the number of times you can read it starting from some letter of the string. Of course, you shouldn't skip letters.

Help her to find the maximum number of "nineteen"s that she can get in her string.

Input

The first line contains a non-empty string *s*, consisting only of lowercase English letters. The length of string *s* doesn't exceed 100.

Output

Print a single integer — the maximum number of "nineteen"s that she can get in her string.

Examples

input

```
nniinneetteeeenn
```

output

```
2
```

input

```
nneteenabcnneteenabcnneteenabcnneteenabcnneteenabcii
```

output

```
2
```

input

```
nineteenineteen
```

output

```
2
```

H. Clock Pictures

Time limit: 2s Memory limit: 1024 MB

You have two pictures of an unusual kind of clock. The clock has n hands, each having the same length and no kind of marking whatsoever. Also, the numbers on the clock are so faded that you can't even tell anymore what direction is up in the picture. So the only thing that you see on the pictures, are n shades of the n hands, and nothing else.

You'd like to know if both images might have been taken at exactly the same time of the day, possibly with the camera rotated at different angles.

Task

Given the description of the two images, determine whether it is possible that these two pictures could be showing the same clock displaying the same time.

Input

The first line contains a single integer n ($2 \leq n \leq 200\,000$), the number of hands on the clock.

Each of the next two lines contains n integers a_i ($0 \leq a_i < 360\,000$), representing the angles of the hands of the clock on one of the images, in thousandths of a degree. The first line represents the position of the hands on the first image, whereas the second line corresponds to the second image. The number a_i denotes the angle between the recorded position of some hand and the upward direction in the image, measured clockwise. Angles of the same clock are distinct and are not given in any specific order.

Output

Output one line containing one word: `possible` if the clocks could be showing the same time, `impossible` otherwise.

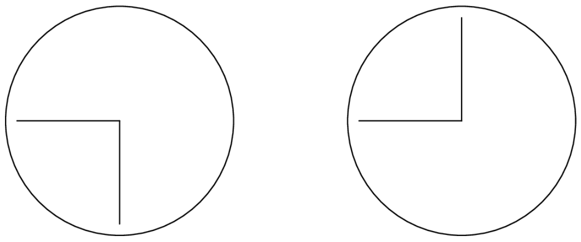


Figure 1: Sample input 2

sample input 1	**sample output 1**

6
1 2 3 4 5 6
7 6 5 4 3 1
	impossible

sample input 2	**sample output 2**

2
0 270000
180000 270000
	possible

sample input 3	**sample output 3**

7
140 130 110 120 125 100 105
235 205 215 220 225 200 240
	impossible

Marc Vinyals, Robin Lee, and Jaap Eldering (Nordic Collegiate Programming Contest 2014)

I. High Probability Cast

Time limit: 2s Memory limit: 256 MB

A mage knows n spells, the i-th of which, when casted, deals random damage (not necessarily integer), uniformly distributed from a_i to b_i. There are m monsters dwelling in the world, and to kill the j-th of them it is required to deal him at least x_j damage. Unfortunately, monsters are so fast that the mage has time to cast only a single spell while fighting each of the monsters, before being killed by that monster. Which spell should be chosen to destroy each of the monsters, so that the probability of killing that monster would be maximized?

Input

The first line contains a single integer n ($1 \le n \le 200000$) — the number of spells.

The next n lines describe spells. Each of them contains two integers a_i and b_i ($1 \le a_i \le b_i \le 10^9$) — the minimal and maximal damage the i-th spell can deal.

The next line contains a single integer m ($1 \le m \le 200000$) — the number of monsters.

The next line contains m integers x_j separated by a space ($1 \le x_j \le 10^9$) — the minimal amount of damage required to destroy the j-th monster.

Output

Output m integers separated by a space, the j-th of which should be equal to the number of spell which should be used to destroy the j-th monster. The spells are numbered from one in the order they are listed in the input. If there are many spells which give the maximal probability of destroying some monster, you can output any of these spells.

Examples

input

```
2
1 10
4 8
4
3 6 7 11
```

output

```
2 2 1 1
```

input

```
2
2 5
7 9
5
10 8 6 3 1
```

output

```
2 2 2 2 2
```

X Samara Regional Intercollegiate Programming Contest

J. Card Game

Time limit: 3s Memory limit: 64 MB

You're playing a card game with K friends of yours, and since you're a champion in this game, they will play together and you will only play with the winner.

And now when they are playing your job is just to deal N cards and distribute them among your friends, you can choose how to distribute them, but the distribution should satisfy these rules:

1- Each player must have a continuous subsequence of the original set.

2- Each card must be dealt to some player.

3- Each player must have at least one card.

Note that it is not important that players have the same number of cards.

You know that all of them are playing using the same strategy so the player with the maximum card group power will win. Each card has a power P the power of group of cards is calculated as (the number of cards in that group) * (the maximum value in the same group).

Since the winner will play with you, and he will play using the same group of cards, you decided to minimize the power of his cards as much as you can.

Write a program to help you to do so.

Input

In the first line one integer T the number of test cases.

For each test case there will be two integers N and K ($1 \le N \le 1000000$, $1 \le K \le min(N, 20000)$), then N integers representing the power of the cards in the original set and their order. ($1 \le P_i \le 1000000$).

Output

For each test case print a single line containing one integer which is the minimum group power you can make the winner player have.

Examples

input

```
1
10 3
1 2 3 4 5 6 7 8 9 10
```

output

```
25
```

2015 Damascus Collegiate Programming Contest (DCPC 2015)

K. Phobia

Time limit: 1s Memory limit: 64 MB

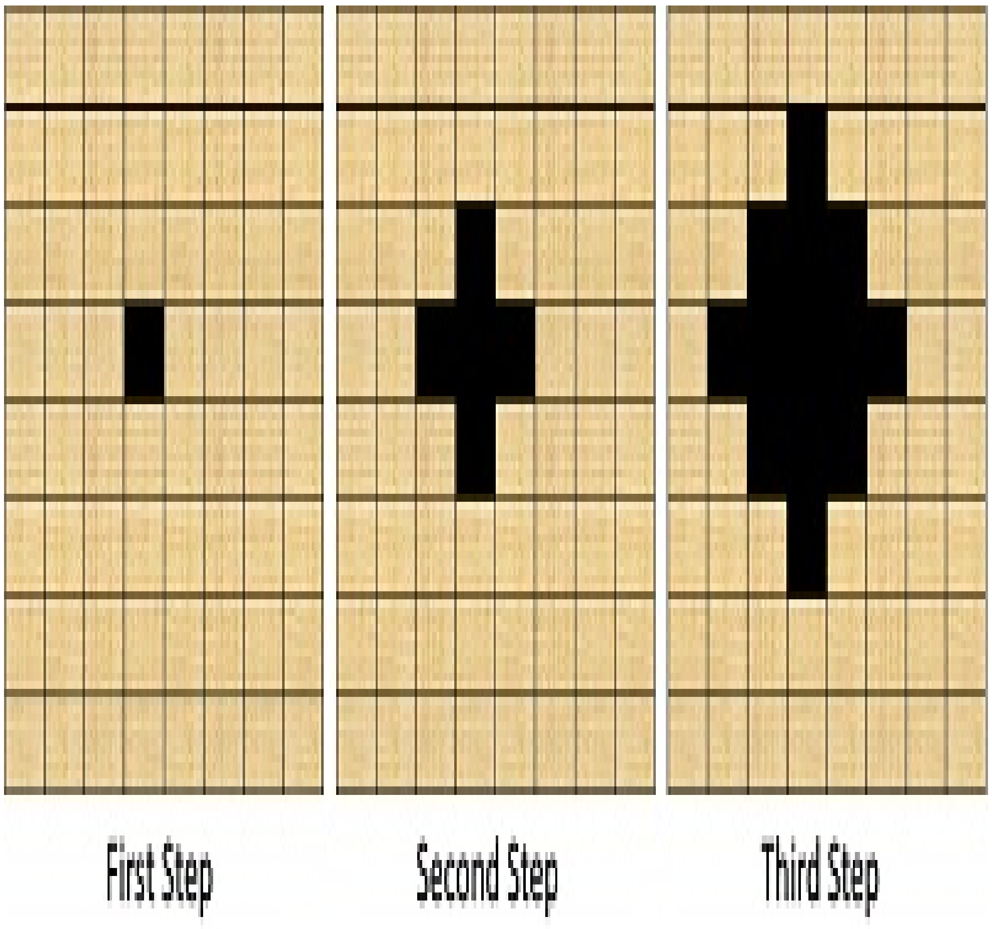
Sally has a phobia of cockroaches. Every night she is having the same horrible nightmare! Where she is standing alone in a big maze surrounded by millions and millions of cockroaches. She wakes up horrified and screaming when she sees a huge cockroach standing on her foot. After many sleepless nights like that one, she finally decided to ask you for help.

Since you have some experience with psychology (after all you are a programmer, right?) you tell her that the best way to cure her phobia is to fight it. She must be strong and fight these stinky little insects. After many argument she agrees with you and decides that she must overcome her fears. To do so she must reach for her sandals (you know, she needs a weapon for this great battle). Sally marked her sandals with a big red X sign to make sure they are clearly noticeable.

Here comes your job! You thought that you should write a program to help her train for this battle. Given the maze information, your program should tell her whether if she could reach her sandals without being touched by any cockroach.

The maze is an $N \times M$ rectangular grid, each cell of this grid is either empty or a wall or a hole. Cockroaches emerge from these holes and spread in all directions. Sally and Cockroaches cannot pass through walls, in other words Sally can only move to an empty cell, and cockroaches can only occupy an empty cell.

Formally, at time t if cell (i, j) is occupied by cockroaches then in time $t+1$ cells $(i+1, j)$, $(i-1, j)$, $(i, j+1)$, $(i, j-1)$ as well as cell (i, j) will be occupied by cockroaches if the respective cells are empty and are inside the borders of the maze.



In one second, Sally can move to an adjacent empty cell. Formally if at time t Sally is at cell (i, j), then in time $t + 1$ she can move to one of these cells: $(i + 1, j)$, $(i - 1, j)$, $(i, j + 1)$, $(i, j - 1)$ if the respective cells are empty and are inside the borders of the maze.

One empty cell contains Sally starting position, and One other empty cell Contains her Sandals. Sally's goal is to reach her sandals without touching any cockroach. Sally touches a cockroach when Sally's current position is occupied by cockroaches.

Input

The first line will be the number of test cases T.

Each test case contains two integers N and M $(1 \le N, M \le 100)$ followed by $N \times M$ grid that represents the maze where:

'S' : sally's starting position.

'X' : Sally's sandals position.

'#' : a Wall.

'*' : a hole.

'.' : an empty cell.

Output

For each test case print "yes" if Sally can exit from the maze without touching any cockroach or "no" otherwise.

Examples

input

```
2
5 5
S..#*
...#.
...##
.....
....X
5 5
S..#*
...#.
...#.
.....
....X
```

output

```
yes
no
```

2015 AlBaath Collegiate Programming Contest

L. Farm

Time limit: 2s Memory limit: 64 MB

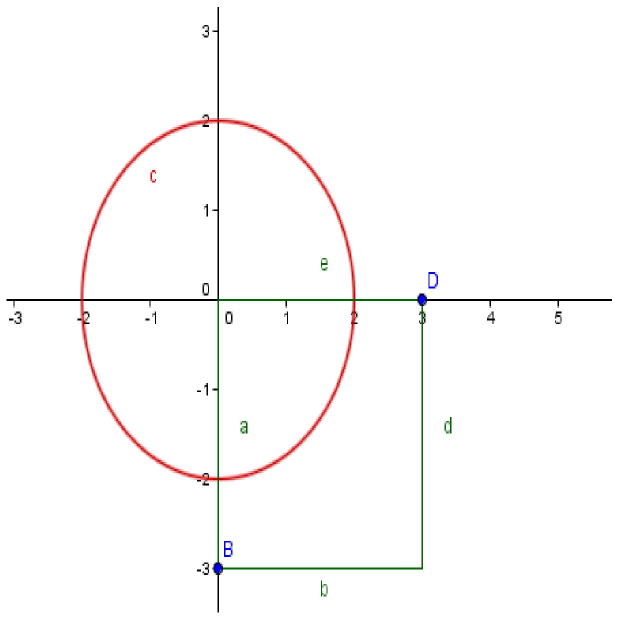
Saleem is a farmer. He has orange trees in his farm. The orange trees need a lot of water but this year there is not enough rain falls. Fortunately there is a well, but not all of this well is inside his farm. Saleem wants to make a deal with his neighbors. Everyone can take one gallon of water for every square meter of the well area inside his farm. you should help Saleem to know how many gallon of water he can take. If you know that:

- The farm is rectangle.

- The well is circle.

- The upper-left corner of his farm is inside the well.

- The width and the height of the farm are bigger than Diameter (2R) of the well.



Input

Your program will be tested on one or more test cases. The first line of the input will be a single integer T, the number of test cases ($1 \le T \le 10000$). Followed by the test cases, each test case is on two lines. the first line consists of three integers $-1000 \le Xc,Yc,R \le 1000$ where (Xc,Yc) is the center of well and R is the radius. the second line consists of four integers $-1000 \le Xb,Yb,Xu,Yu \le 1000$ where (Xb,Yb) is the bottom-left corner and (Xu,Yu) is the upper-right corner.

Output

For each test case print a line is the following format: "Case c: x" where c is the test case number starting from 1 and x is the amount of water rounded to 5 decimal places.

Examples

input

```
1
0 0 2
0 -5 5 0
```

output

```
Case 1: 3.14159
```

2015 Tishreen Collegiate Programming Contest