Codeforces Round #597 (Div. 2)

A. Good ol' Numbers Coloring

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Consider the set of all nonnegative integers: $0, 1, 2, \ldots$. Given two integers a and b ($1 \le a, b \le 10^4$). We paint all the numbers in increasing number first we paint 0, then we paint 1, then 2 and so on.

Each number is painted white or black. We paint a number i according to the following rules:

- if $i = 0$, it is colored white;
- if $i \ge a$ and $i - a$ is colored white, i is also colored white;
- if $i \ge b$ and $i - b$ is colored white, i is also colored white;
- if i is still not colored white, it is colored black.

In this way, each nonnegative integer gets one of two colors.

For example, if $a = 3$, $b = 5$, then the colors of the numbers (in the order from 0) are: white (0), black (1), black (2), white (3), black (4), white (5), white (6), black (7), white (8), white (9), ...

Note that:

- It is possible that there are infinitely many nonnegative integers colored black. For example, if $a = 10$ and $b = 10$, then only $0, 10, 20, 30$ and any other nonnegative integers that end in 0 when written in base 10 are white. The other integers are colored black.
- It is also possible that there are only finitely many nonnegative integers colored black. For example, when $a = 1$ and $b = 10$, then there is no nonnegative integer colored black at all.

Your task is to determine whether or not the number of nonnegative integers colored **black** is infinite.

If there are infinitely many nonnegative integers colored black, simply print a line containing "`Infinite`" (without the quotes). Otherwise, print "`Finite`" (without the quotes).

Input

The first line of input contains a single integer t ($1 \le t \le 100$) — the number of test cases in the input. Then t lines follow, each line contains two space-separated integers a and b ($1 \le a, b \le 10^4$).

Output

For each test case, print one line containing either "`Infinite`" or "`Finite`" (without the quotes). Output is case-insensitive (i.e. "`infinite`", "`inFiNite`" or "`finiTE`" are all valid answers).

Example

input
4 10 10 1 10 6 9 7 3
output
Infinite Finite Infinite Finite

B. Restricted RPS

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Let n be a positive integer. Let a, b, c be nonnegative integers such that $a + b + c = n$.

Alice and Bob are gonna play rock-paper-scissors n times. Alice knows the sequences of hands that Bob will play. However, Alice has to play rock a times, paper b times, and scissors c times.

Alice wins if she beats Bob in at least $\lceil \frac{n}{2} \rceil$ ($\frac{n}{2}$ rounded up to the nearest integer) hands, otherwise Alice loses.

Note that in rock-paper-scissors:

- rock beats scissors;
- paper beats rock;
- scissors beat paper.

The task is, given the sequence of hands that Bob will play, and the numbers a, b, c, determine whether or not Alice can win. And if so, find any possible sequence of hands that Alice can use to win.

If there are multiple answers, print any of them.

Input
The first line contains a single integer t ($1 \le t \le 100$) — the number of test cases.

Then, t testcases follow, each consisting of three lines:

- The first line contains a single integer n ($1 \le n \le 100$).
- The second line contains three integers, a, b, c ($0 \le a, b, c \le n$). It is guaranteed that $a + b + c = n$.
- The third line contains a string s of length n. s is made up of only 'R', 'P', and 'S'. The i-th character is 'R' if for his i-th Bob plays rock, 'P' if paper, and 'S' if scissors.

Output
For each testcase:

- If Alice cannot win, print "NO" (without the quotes).
- Otherwise, print "YES" (without the quotes). Also, print a string t of length n made up of only 'R', 'P', and 'S' — a sequence of hands that Alice can use to win. t must contain exactly a 'R's, b 'P's, and c 'S's.
- If there are multiple answers, print any of them.

The "YES" / "NO" part of the output is case-insensitive (i.e. "yEs", "no" or "YEs" are all valid answers). Note that 'R', 'P' and 'S' are case-sensitive.

Example

input
2 3 1 1 1 RPS 3 3 0 0 RPS
output
YES PSR NO

Note
In the first testcase, in the first hand, Alice plays paper and Bob plays rock, so Alice beats Bob. In the second hand, Alice plays scissors and Bob plays paper, so Alice beats Bob. In the third hand, Alice plays rock and Bob plays scissors, so Alice beats Bob. Alice beat Bob 3 times, and $3 \ge \lceil \frac{3}{2} \rceil = 2$, so Alice wins.

In the second testcase, the only sequence of hands that Alice can play is "RRR". Alice beats Bob only in the last hand, so Alice can't win. $1 < \lceil \frac{3}{2} \rceil = 2$.

C. Constanze's Machine

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Constanze is the smartest girl in her village but she has bad eyesight.

One day, she was able to invent an incredible machine! When you pronounce letters, the machine will inscribe them onto a piece of paper. For example, if you pronounce 'c', 'o', 'd', and 'e' in that order, then the machine will inscribe "code" onto the paper. Thanks to this machine, she can finally write messages without using her glasses.

However, her dumb friend Akko decided to play a prank on her. Akko tinkered with the machine so that if you pronounce 'w', it will inscribe "uu" instead of "w", and if you pronounce 'm', it will inscribe "nn" instead of "m"! Since Constanze had bad eyesight, she was not able to realize what Akko did.

The rest of the letters behave the same as before: if you pronounce any letter besides 'w' and 'm', the machine will just inscribe it onto a piece of paper.

The next day, I received a letter in my mailbox. I can't understand it so I think it's either just some gibberish from Akko, or Constanze made it using her machine. But since I know what Akko did, I can just list down all possible strings that Constanze's machine would have turned into the message I got and see if anything makes sense.

But I need to know how much paper I will need, and that's why I'm asking you for help. Tell me the number of strings that Constanze's machine would've turned into the message I got.

But since this number can be quite large, tell me instead its remainder when divided by $10^9 + 7$.

If there are no strings that Constanze's machine would've turned into the message I got, then print 0.

Input
Input consists of a single line containing a string s ($1 \leq |s| \leq 10^5$) — the received message. s contains only lowercase Latin letters.

Output
Print a single integer — the number of strings that Constanze's machine would've turned into the message s, modulo $10^9 + 7$.

Examples

input
ouuokarinn
output
4

input
banana
output
1

input
nnn
output
3

input
amanda
output
0

Note
For the first example, the candidate strings are the following: "`ouuokarinn`", "`ouuokarim`", "`owokarim`", and "`owokarinn`".

For the second example, there is only one: "`banana`".

For the third example, the candidate strings are the following: "`nm`", "`mn`" and "`nnn`".

For the last example, there are no candidate strings that the machine can turn into "`amanda`", since the machine won't inscribe 'm'.

D. Shichikuji and Power Grid

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

Shichikuji is the new resident deity of the South Black Snail Temple. Her first job is as follows:

There are n new cities located in Prefecture X. Cities are numbered from 1 to n. City i is located x_i km North of the shrine and y_i km East of the shrine. It is possible that $(x_i, y_i) = (x_j, y_j)$ even when $i \neq j$.

Shichikuji must provide electricity to each city either by building a power station in that city, or by making a connection between that city and another one that already has electricity. So the City has electricity if it has a power station in it or it is connected to a City which has electricity by a direct connection or via a chain of connections.

- Building a power station in City i will cost c_i yen;
- Making a connection between City i and City j will cost $k_i + k_j$ yen **per km of wire** used for the connection. However, wires can only go the cardinal directions (North, South, East, West). Wires can cross each other. Each wire must have both of its endpoints in some cities. If City i and City j are connected by a wire, the wire will go through any shortest path from City i to City j. Thus, the length of the wire if City i and City j are connected is $|x_i - x_j| + |y_i - y_j|$ km.

Shichikuji wants to do this job spending as little money as possible, since according to her, there isn't really anything else in the world other than money. However, she died when she was only in fifth grade so she is not smart enough for this. And thus, the new

resident deity asks for your help.

And so, you have to provide Shichikuji with the following information: minimum amount of yen needed to provide electricity to all cities, the cities in which power stations will be built, and the connections to be made.

If there are multiple ways to choose the cities and the connections to obtain the construction of minimum price, then print any of them.

Input

First line of input contains a single integer n ($1 \leq n \leq 2000$) — the number of cities.

Then, n lines follow. The i-th line contains two space-separated integers x_i ($1 \leq x_i \leq 10^6$) and y_i ($1 \leq y_i \leq 10^6$) — the coordinates of the i-th city.

The next line contains n space-separated integers $c_1, c_2, \ldots, c_n$ ($1 \leq c_i \leq 10^9$) — the cost of building a power station in the i-th city.

The last line contains n space-separated integers $k_1, k_2, \ldots, k_n$ ($1 \leq k_i \leq 10^9$).

Output

In the first line print a single integer, denoting the minimum amount of yen needed.

Then, print an integer v — the number of power stations to be built.

Next, print v space-separated integers, denoting the indices of cities in which a power station will be built. Each number should be from 1 to n and all numbers should be pairwise distinct. You can print the numbers in arbitrary order.

After that, print an integer e — the number of connections to be made.

Finally, print e pairs of integers a and b ($1 \leq a, b \leq n$, $a \neq b$), denoting that a connection between City a and City b will be made. Each unordered pair of cities should be included at most once (for each (a, b) there should be no more (a, b) or (b, a) pairs). You can print the pairs in arbitrary order.

If there are multiple ways to choose the cities and the connections to obtain the construction of minimum price, then print any of them.

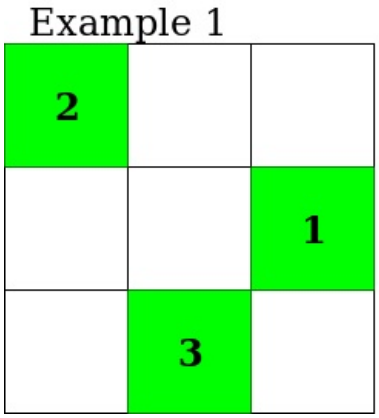
Examples

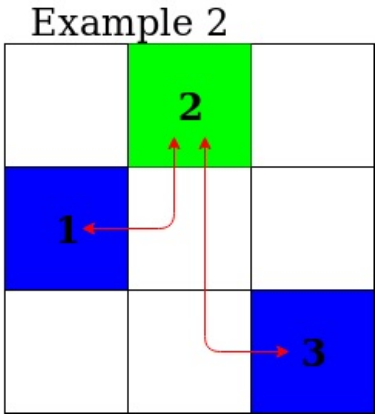
input
3 2 3 1 1 3 2 3 2 3 3 2 3
output
8 3 1 2 3 0

input
3 2 1 1 2 3 3 23 2 23 3 2 3
output
27 1 2 2 1 2 2 3

Note

For the answers given in the samples, refer to the following diagrams (cities with power stations are colored green, other cities are colored blue, and wires are colored red):





For the first example, the cost of building power stations in all cities is $3 + 2 + 3 = 8$. It can be shown that no configuration costs less than 8 yen.

For the second example, the cost of building a power station in City 2 is 2. The cost of connecting City 1 and City 2 is $2 \cdot (3 + 2) = 10$. The cost of connecting City 2 and City 3 is $3 \cdot (2 + 3) = 15$. Thus the total cost is $2 + 10 + 15 = 27$. It can be shown that no configuration costs less than 27 yen.

E. Hyakugoku and Ladders

time limit per test: 1 second
memory limit per test: 256 megabytes
input: standard input
output: standard output

Hyakugoku has just retired from being the resident deity of the South Black Snail Temple in order to pursue her dream of becoming a cartoonist. She spent six months in that temple just playing "Cat's Cradle" so now she wants to try a different game — "Snakes and Ladders". Unfortunately, she already killed all the snakes, so there are only ladders left now.

The game is played on a 10×10 board as follows:

- At the beginning of the game, the player is at the bottom left square.
- The objective of the game is for the player to reach the Goal (the top left square) by following the path and climbing vertical ladders. Once the player reaches the Goal, the game ends.
- The path is as follows: if a square is not the end of its row, it leads to the square next to it along the direction of its row; if a square is the end of its row, it leads to the square above it. The direction of a row is determined as follows: the direction of the bottom row is to the right; the direction of any other row is opposite the direction of the row below it. See Notes section for visualization of path.
- During each turn, the player rolls a standard six-sided dice. Suppose that the number shown on the dice is r. If the Goal is less than r squares away on the path, the player doesn't move (but the turn is performed). Otherwise, the player advances exactly r squares along the path and then stops. If the player stops on a square with the bottom of a ladder, the player **chooses whether or not to climb up** that ladder. If she chooses not to climb, then she stays in that square for the beginning of the next turn.
- Some squares have a ladder in them. Ladders are only placed vertically — each one leads to the same square of some of the upper rows. In order for the player to climb up a ladder, after rolling the dice, she must stop at the square containing the bottom of the ladder. After using the ladder, the player will end up in the square containing the top of the ladder. She cannot leave the ladder in the middle of climbing. And if the square containing the top of the ladder also contains the bottom of another ladder, she is not allowed to use that second ladder.
- The numbers on the faces of the dice are 1, 2, 3, 4, 5, and 6, with each number having the same probability of being shown.

Please note that:

- it is possible for ladders to overlap, but the player cannot switch to the other ladder while in the middle of climbing the first one;
- it is possible for ladders to go straight to the top row, but not any higher;
- it is possible for two ladders to lead to the same tile;
- it is possible for a ladder to lead to a tile that also has a ladder, but the player will not be able to use that second ladder if she uses the first one;
- the player can only climb up ladders, not climb down.

Hyakugoku wants to finish the game as soon as possible. Thus, on each turn she chooses whether to climb the ladder or not optimally. Help her to determine the minimum expected number of turns the game will take.

Input

Input will consist of ten lines. The i-th line will contain 10 non-negative integers $h_{i1}, h_{i2}, \ldots, h_{i10}$. If h_{ij} is 0, then the tile at the i-th row and j-th column has no ladder. Otherwise, the ladder at that tile will have a height of h_{ij}, i.e. climbing it will lead to the tile h_{ij} rows directly above. It is guaranteed that $0 \leq h_{ij} < i$. Also, the first number of the first line and the first number of the last line always contain 0, i.e. the Goal and the starting tile never have ladders.

Output

Print only one line containing a single floating-point number — the minimum expected number of turns Hyakugoku can take to finish

the game. Your answer will be considered correct if its absolute or relative error does not exceed 10^{-6}.

Examples

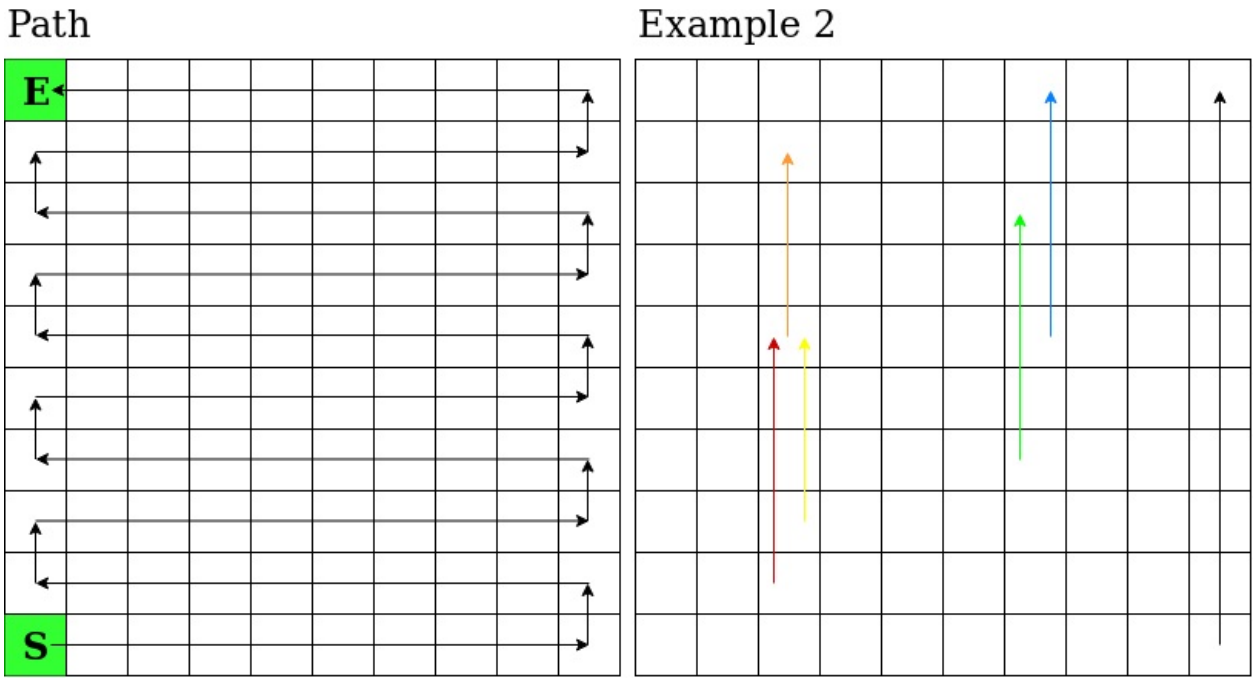
input
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
output
33.0476190476

input
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 3 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 3 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 9
output
20.2591405923

input
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 6 6 6 6 6 6 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
output
15.9047592939

Note

A visualization of the path and the board from example 2 is as follows:



The tile with an 'S' is the starting tile and the tile with an 'E' is the Goal.

For the first example, there are no ladders.

For the second example, the board looks like the one in the right part of the image (the ladders have been colored for clarity).

It is possible for ladders to overlap, as is the case with the red and yellow ladders and green and blue ladders. It is also possible for ladders to go straight to the top, as is the case with the black and blue ladders. However, it is not possible for ladders to go any higher (outside of the board). It is also possible that two ladders lead to the same tile, as is the case with the red and yellow ladders. Also, notice that the red and yellow ladders lead to the tile with the orange ladder. So if the player chooses to climb either of the red and yellow ladders, they will not be able to climb the orange ladder. Finally, notice that the green ladder passes through the starting tile of the blue ladder. The player cannot transfer from the green ladder to the blue ladder while in the middle of climbing the green ladder.

F. Daniel and Spring Cleaning

time limit per test: 2 seconds
memory limit per test: 256 megabytes
input: standard input
output: standard output

While doing some spring cleaning, Daniel found an old calculator that he loves so much. However, it seems like it is broken. When he tries to compute $1 + 3$ using the calculator, he gets 2 instead of 4. But when he tries computing $1 + 4$, he gets the correct answer, 5. Puzzled by this mystery, he opened up his calculator and found the answer to the riddle: the full adders became half adders!

So, when he tries to compute the sum $a + b$ using the calculator, he instead gets the xorsum $a \oplus b$ (read the definition by the link: https://en.wikipedia.org/wiki/Exclusive_or).

As he saw earlier, the calculator sometimes gives the correct answer. And so, he wonders, given integers l and r, how many pairs of integers (a, b) satisfy the following conditions:

$$a + b = a \oplus b$$
$$l \leq a \leq r$$
$$l \leq b \leq r$$

However, Daniel the Barman is going to the bar and will return in two hours. He tells you to solve the problem before he returns, or else you will have to enjoy being blocked.

Input
The first line contains a single integer t ($1 \leq t \leq 100$) — the number of testcases.

Then, t lines follow, each containing two space-separated integers l and r ($0 \leq l \leq r \leq 10^9$).

Output
Print t integers, the i-th integer should be the answer to the i-th testcase.

Example

input
3 1 4 323 323 1 1000000
output
8 0 3439863766

Note
$a \oplus b$ denotes the bitwise XOR of a and b.

For the first testcase, the pairs are: $(1, 2)$, $(1, 4)$, $(2, 1)$, $(2, 4)$, $(3, 4)$, $(4, 1)$, $(4, 2)$, and $(4, 3)$.