Problem A. Young Adleman

Program:	`young.(c|cpp|java)`
Input:	`young.in`
Balloon Color:	`Blue`

Adleman and Rivest are very interested in the field of cryptography and Data Security. This interest started at a very young age when they were still in school, they used to pass encrypted messages to each other during boring classes like history and geography.

They used their own complete cryptosystem where they used a technique called substitution to encode their messages. It used a key which is an integer that they would agree upon beforehand. They decided to add an extra layer of security (and fun) by creating a way to choose a different key every time.

Adleman would give Rivest two strings S_1 and S_2, both consist of lowercase English letters only. In order for Rivest to conclude the secret key, he must find out the minimum length of a sub-string in S_1 that contains all the letters in S_2 at least once.

Can you write a program that helps Rivest find out the minimum length that satisfies Adleman's condition? If there is no substring in S_1 that contains all the letters in S_2 at least once, print -1.

Input

The first line contains an integer T ($1 \le T \le 100$) — the number of test cases.

T test cases follow, each consists of two lines. The first line contains the first string S_1 of length N_1 ($1 \le N_1 \le 10^6$). The second line contains the second string S_2 of length N_2 ($1 \le N_2 \le 10^6$). It is guaranteed that the sum of the lengths of all strings in the input will not exceed $2 \cdot 10^6$

Output

For each test case print out a single number, the minimum length of the subsegment in S_1 that contains all the letters in S_2 at least once. If there is no solution, print -1.

Example

young.in	Standard Output
4 abcdeed cee abcaab aa abcaatb aabc abbcda baaab	4 2 4 -1

Problem B. Self Describing Numbers

Program:	`self.(c|cpp|java)`
Input:	`self.in`
Balloon Color:	`Yellow`

In one of the classes, Dr. Edsger explained that an integer is said to be "self-describing" if it has the property that: the first digit represents the number of zeros in the integer; the second digit represents the number of ones in the integer, and so on.

For example, 1210 is a self-describing integer since:

- the first digit is 1 and there is 1 zero in the number &
- the second digit is 2 and there are 2 ones in the number &
- the third digit is 1 and there is 1 two in the number &
- the fourth digit is 0 and there are 0 threes in the number.

At the end of the class, Dr. Edsger gave a small homework: create a program to check whether an integer is self-describing or not.

Input

The first line of the input contains an integer T $(1 \leq T \leq 100)$ — the number of test cases.

Each test case consists of a single integer N $(0 \leq N < 10^{10})$, without leading zeros.

Output

For each test case, print a single line containing the answer which is either "self-describing" or "not self-describing" without the quotes.

Example

self.in	Standard Output
2 1210 2017	self-describing not self-describing

Problem C. Numberica

Program:	`numberica.(c|cpp|java)`
Input:	`numberica.in`
Balloon Color:	`Rose`

A list of N integers, each of which contains D digits, is called *connected* if there is an ordering of the N integers such that the last $D-1$ digits of each integer (except the last one) is the same as the first $D-1$ digits of the next integer.

For example, the list of the three integers: $[839, 183, 392]$ is connected because we can order them as $[183 - 839 - 392]$. However, the list of four integers: $[22, 50, 66, 93]$ is not connected because there is no way to order them so that the above rule applies.

Input

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

Each test case starts with two integers N and D $(1 \le N \le 10^5, 2 \le D \le 18\ )$ — the number of integers in the list and the number of digits in each integers.

The next line contains N space-separated positive integers a_i $(10^{D-1} \le a_i < 10^D)$ — the integers of the list with no leading zeros.

Output

For each test case, you will print one or two lines.

If there is a way to order the N integers such that the above rule applies, print on the first line `Connected` and on the second line the correct order of the numbers. Otherwise (i.e. there is no way to order the N integers), print `Not Connected` on a single line.

If there are several ways to order the list, print the lexicographical minimal order.

Example

numberica.in	Standard Output

5
3 3
839 183 392
4 2
22 50 66 93
2 3
111 110
3 3
715 157 571
9 3
715 157 571 153 535 357 573 731 315
	Connected
	183 839 392
	Not Connected
	Connected
	111 110
	Connected
	157 571 715
	Connected
	153 535 357 571 715 157 573 731 315

Problem D. Cycles in a String

Program:	`cycles.(c|cpp|java)`
Input:	`cycles.in`
Balloon Color:	`White`

You are given a string S of length N and a permutation P of the indexes of S. We can use this permutation to construct a graph on the string indices. Formally, for $1 \leq i \leq N$, P_i means that there is a directed edge from index P_i to index i. The constructed graph is a set of disjoint cycle graphs. A move is to shift the letters in one cycle graph one step according to the edges' direction.

Consider the following string $S =$ `deacfgb` and the permutation $P = [4\ 5\ 1\ 3\ 6\ 7\ 2]$. The edges defined by the permutation are $(1 \leftarrow 4)$, $(2 \leftarrow 5)$, $(3 \leftarrow 1)$, $(4 \leftarrow 3)$, $(5 \leftarrow 6)$, $(6 \leftarrow 7)$ and $(7 \leftarrow 2)$. The constructed graph consists of two cycle graphs: $(1 \leftarrow 4 \leftarrow 3)$ and $(2 \leftarrow 5 \leftarrow 6 \leftarrow 7)$. If we make one move on the first cycle graph, we will get $S' =$ `cedafgb`. After that, if we make two moves on the second cycle graph, we will get $S'' =$ `cgdabef`.

you are given a special case of the problem where the condition P[i] $>=$ i or P[i] $=$ the smallest number in the cycle

What is the minimum number of moves to obtain the lexicographically smallest string?

Input

The first line of input contains an integer T $(1 \leq T \leq 100)$ — the number of test cases. Then T test cases follow.

Each test case consists of three lines. The first line contains N $(1 \leq N \leq 10^5)$. The second line contains an array of N integers that represent a permutation P of the original string indices. The third and last line contains S — the original string.

Output

For each test case print one line, which contains the minimum number of moves, and the lexicographically smallest string separated by a space.

Example

cycles.in	Standard Output
2 6 2 3 4 5 6 1 bababa 3 3 2 1 baa	1 ababab 1 aab

Problem E. Donald's Riddle

Program:	`donald.(c|cpp|java)`
Input:	`donald.in`
Balloon Color:	`Black`

Dr. Knuth is a great computer science professor and his students love him. His classes are both entertaining and useful, and he always has the most interesting examples and riddles.

In one of his beginner classes, he gave the students two strings S and T of the same length N. They both consisted of lowercase and uppercase English letters only. Then, he asked the students to transform string S to string T with the minimum cost they can obtain. They were allowed to apply the following operations each with its own cost:

- Any lowercase letter in S can be transformed to any lowercase letter with cost a.
- Any uppercase letter in S can be transformed to any uppercase letter with cost b.
- Any lowercase letter in S can be transformed to its respective uppercase letter with cost c.
- Any uppercase letter in S can be transformed to its respective lowercase letter with cost d.

Note that you cannot change the order of the letters in any of the strings, you can only apply the operations above without swapping or changing positions of the letters.

Can you write a program to compute the minimum cost needed to transform S into T ?

Input

The first line contains an integer T $(1 \leq T \leq 100)$ — the number of test cases.

T test cases follow, each consists of three lines. The first line contains four integers a, b, c, d $(1 \leq a, b, c, d \leq 10^5)$, the cost for each of the operations mentioned above, respectively. The following two lines contain the strings S and T respectively, both have the same length N $(1 \leq N \leq 10^5)$.

Output

For each test case print the minimum cost needed to transform the string S into the string T using the mentioned operations.

Example

donald.in	Standard Output
5	44
2 5 6 5	19
dIADTLC	48
UegMDBy	39
2 4 7 6	31
VHifn	
APwyN	
1 9 6 10	
TYtszO	
wflEGs	
5 8 1 10	
CgUem	
rcZQx	
1 10 2 5	
PNxNN	
MudUH	

Problem F. Sort Segments

Program:	`sort.(c|cpp|java)`
Input:	`sort.in`
Balloon Color:	`Green`

Albert was bored of doing research all day, so he decided to take a break and play a game of numbers with his wife Mileva.

Mileva would give Albert an array of size N and two integers M and K. Albert needs to:

- Sort the elements in the segment $[1..M]$
- Sort the elements in the segment $[2..M+1]$
- Sort the elements in the segment $[3..M+2]$
- And so on, until you finally sort the elements in the segment $[N-M+1..N]$

Then Albert's final answer must be the element at index K of the final array (indexes start from 1). Given the array of size N and the integers M and K, can you help Albert write a program to find out the K_{th} element after sorting the segments as defined above and answer Mileva quickly and correctly?

Input

The first line contains a single integer T $(1 \le T \le 100)$ — the number of test cases. Then T test cases follow, each consisting of two lines.

The first line of each test case contains three integers N, M, K $(1 \le M, K \le N \le 10^5)$ — the length of the array, the length of the segments to be sorted and the index of the element, respectively.

The second line contains N space-separated integers $a_1, a_2, ...a_N$ $(1 \le a_i \le 10^9)$ — the array elements.

Output

For each test case, print a single integer which is the value of the element at index K.

Example

sort.in	Standard Output

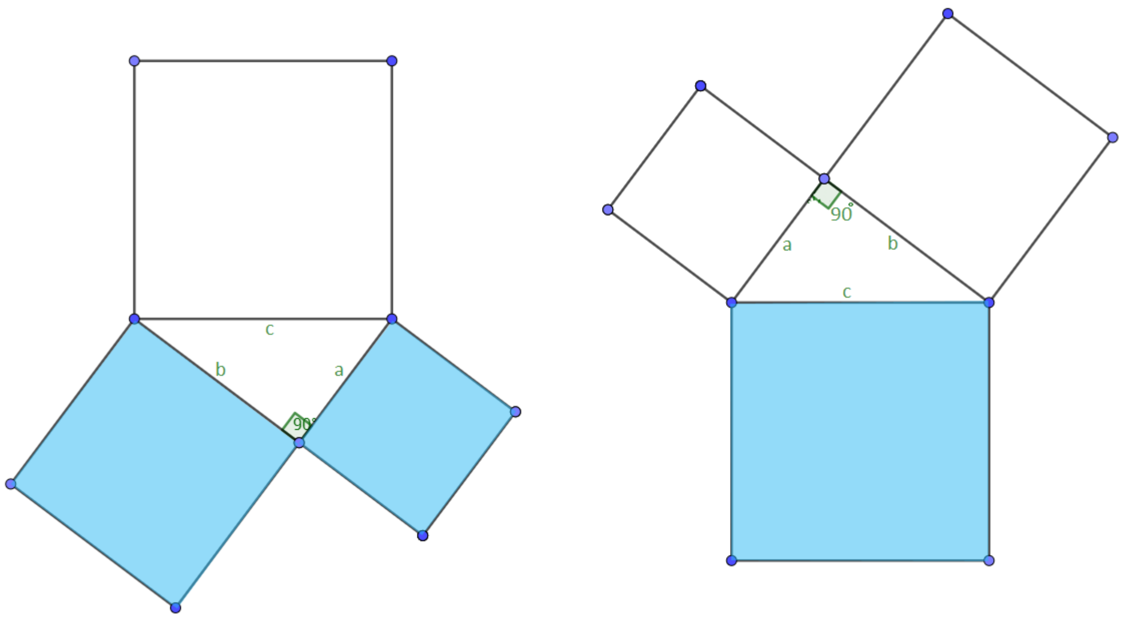
5
2 1 2
45 72
1 1 1
97
2 1 1
82 84
7 4 6
2 28 37 57 39 18 11
9 6 8
68 68 16 40 63 93 49 91 10
	72
	97
	82
	39
	91

Problem G. Geometry is Beautiful

Program:	`alan.(c|cpp|java)`
Input:	`alan.in`
Balloon Color:	`Red`

Alan is a great computer scientist, but he also loves decorating his home, making it more pleasant and unique. He is really good at decorations and internal design, he is the best at using geometrical shapes to create the most beautiful abstract designs.

His latest invention was creating the geometrical piece of art shown in the picture below. He basically made a right-angled triangular revolving wooden frame, then attached a cube-shaped glass container to each side of it. He filled the two smaller cubes with water, then he would turn the wooden frame upside down and watch the water from the two smaller cubes fill up the big cube. It was fascinating!



Alan's friends loved the idea, and wanted to have the same piece of art in their houses, but each of them wants it in a different size. Given A and B, the lengths of the two smaller sides of the triangular wooden frame, D, the depth of the three glass cubes, can you find out the volumes for the three required cubes sorted from smallest to biggest, considering that the water from the two smaller cubes must fill the big cube completely (no more and no less)?

Input

The first line contains an integer T $(1 \le T \le 100)$, the number of test cases.

Each test case consists of a single line which contains three integers: D $(1 \le D \le 10^5)$, the depth of the three glass cubes, A $(1 \le A \le 10^5)$, B $(1 \le B \le 10^5)$ the lengths of the two shorter sides of the rectangular right-angled frame sorted from shortest to longest.

Output

For each test case, print a single line that contains three values, the volumes of the required cubes sorted from smallest to largest.

Example

alan.in	Standard Output

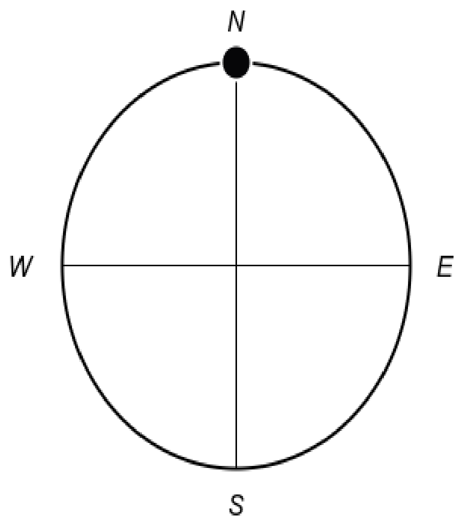
3
2 3 4
3 6 8
5 11 30
	18 32 50
	108 192 300
	605 4500 5105

Problem H. Point on a Circle

Program:	`point.(c|cpp|java)`
Input:	`point.in`
Balloon Color:	`Purple`

We have a point on a circle and we have a sequence of movements where each move is one of the following:
1) `N`: Go North
2) `E`: Go East
3) `S`: Go South
4) `W`: Go West.

In each move, the point can cross no more than a quarter of the circle (clockwise or counter-clockwise). In other words, the point can only move to its adjacent directions, otherwise, the move is ignored. For example, the point in the figure is in the North. If the next move is `E` or `W`, it will move a quarter of the circle clockwise or counter-clockwise, respectively. Otherwise, it will stay in its position.



Initially, the point is in the North. Let's denote a *positive cycle* as crossing the 4 quarters of the circle clockwise and ending in the North. For example, the sequences `ESWN`, `ESESNWN` and `WNESWN` will form a positive cycle while the sequences `ESWSW`, `SSWN` and `WNESW` will not. Also, let's denote a *negative cycle* as crossing the 4 quarters of the circle counter-clockwise and ending in the North.

Given the sequence of movements and an integer K. Your task is to calculate the minimum number of moves to change such that the difference between the number of positive cycles and negative cycles is exactly K after applying all moves (i.e. $Count_{pos} - Count_{neg} = K$).

Input

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

The first line of each case contains two integers N and K ($1 \le K \le N \le 1000$) — the length of the sequence and the required difference, respectively. The second line of each case contains a string of length N consisting only of the characters `[ N, E, S, W ]`.

Output

For each test case, print a single integer which is the minimum number of moves to change such that the difference between positive cycles and negative cycles is exactly K. If there is no solution, print `-1`.

Example

point.in	Standard Output
10 8 2 NNNESSWW 1 1 S 4 1 NNNS 3 1 SEW 6 2 EEWNNW 6 1 EWWSEN 4 1 WEES 3 1 NSW 8 2 SNENWNNS 3 1 EEN	6 -1 4 -1 -1 1 4 -1 7 -1

Note

For any given sequence of moves, the number of positive and negative cycles are unique and the cycles can overlap only at one point. For example, the sequence `NESWNE` contains only one positive cycle which is `NESWN` while the sequence `NESWNESWN` contains two positive cycles `NESWN` and `NESWN`. You should count cycles from left to right and if a cycle overlaps with a valid cycle in more than one point, then this cycle is not valid.

Problem I. Laser Gun

Program:	`laser.(c|cpp|java)`
Input:	`laser.in`
Balloon Color:	`Gold`

You have two parallel mirrors each of length N and they are positioned one unit apart from each other. Each mirror has N green points located on its integer coordinates $1, 2, 3, ..., N$.

You installed one laser gun on the K^{th} green point on the left mirror.

The gun will shoot one laser beam to any random point of the $N - 1$ other green points on the right mirror and this beam will be reflected to the first mirror then bounce back to the second mirror and so on until it gets out from the two mirrors range.

For example, consider having two mirrors of the length 5 where the laser gun is located at $K = 2$ of the first mirror. The beam will be shot to points 1, 3, 4 or 5:

If it was shot to point 1 it will be reflected once and bounces out of the two mirrors' range.
If it was shot to point 3 it will be reflected three times ($3 \to 4$ then $4 \to 5$ then $5 \to outofmirrors'range$).
If it was shot to point 4 it will be reflected once.
If it was shot to point 5 it will be reflected once.

Can you calculate the expected value of the number of reflections that would occur to the laser beam?

Input

The first line of the input contains a single integer T $(1 \leq T \leq 100)$ — the number of test cases.

Each test case consists of a single line that contains two integers N and K $(1 \leq K \leq N \leq 10^9)$ the length of the mirrors and the point where the laser gun is located at, respectively.

Output

For each test case, print the expected value of the number of reflections rounded to 6 decimal places.

Example

laser.in	Standard Output
6	1.666667
4 1	1.333333
4 2	1.333333
4 3	1.666667
4 4	20.877698
1000000000 1	20.184551
1000000000 500000000	

Problem J. Descartes Coordinates

Program:	`descartes.(c|cpp|java)`
Input:	`descartes.in`
Balloon Color:	`Pink`

Descartes loves drawing lines on 2D planes. He was doodling on a piece of paper when he encountered the following problem. He Drew N distinct lines on a plane, where:

- K lines of them have equal slope S.
- No two lines from the other $N - K$ lines have an equal slope, or have a slope equal to S.

Descartes was wondering what the total number of intersection points would be (while ignoring the two axes). He guarantees that each intersection point is formed by two lines only. Can you help him find out the solution?

Input

The first line contains an integer T $(1 \leq T \leq 100)$ — the number of test cases.

T test cases follow, each consists of a single line with two integers N and K $(1 \leq K \leq N \leq 10^9)$ — the total number of lines and the number of lines with equal slopes.

Output

For each test case print a single number representing the total number of intersection points.

Example

descartes.in	Standard Output

2
4 2
3 3
	5
	0

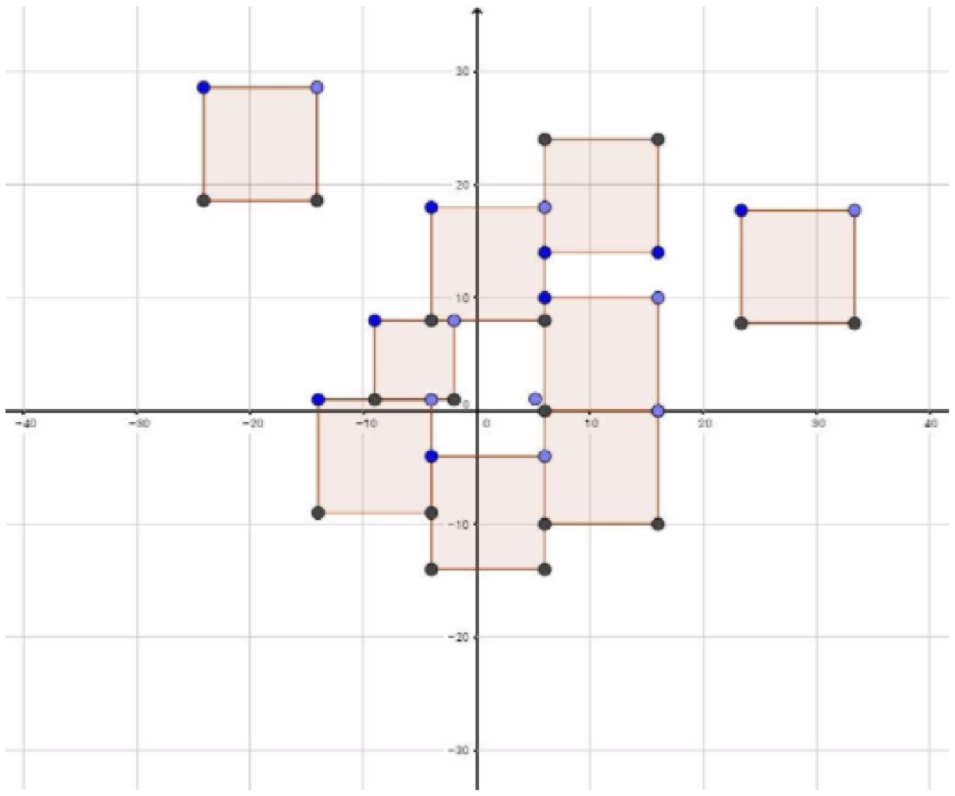
Problem K. IBM Balloons

Program:	`squares.(c|cpp|java)`
Input:	`squares.in`
Balloon Color:	`Grey`

As most of you know, IBM offers the World Finals every year a collection of colorful beautiful balloons to use for solved problems during the contest. They always add a huge blue IBM balloon with every set of balloon colors. This year, IBM invented irregular balloons, that are cube-shaped, and they can expand infinitely.

ICPC volunteers found this fascinating, so they wanted to play a game. Consider yourself looking at these volunteers from the top floor. From the top, the contest hall looks like a 2D plane and volunteers are points on the plane. One of the volunteers stands at the center of the hall (i.e at point $(0, 0)$) while the other volunteers stand at other locations carrying the cubic balloons. The balloons appear as squares from the top where the sides of the squares are parallel to the x and y axes. The center of each square is at the point where the volunteer stands and its area is initially zero since volunteers have not started blowing up the balloons yet.

Each volunteer will start blowing up his/her balloon and the side of the balloon square will expand at a specific rate R. That's, each second, the length of the side of the balloon square increases by R units of distance.



What is the minimum time before the volunteer standing in $(0, 0)$ gets his vision blocked (in all 2D directions) by the balloons around him?

Input

The first line of the input contains a single integer T ($1 \leq T \leq 70$) — the number of test cases.

Each test case starts with two integers N and R ($1 \leq N \leq 25000$, $1 \leq R \leq 10^9$) — the number of volunteers holding balloons and the increase rate of the side length of the balloon square.

Then N lines follow, each contains two integers x_i and y_i ($-10^6 \leq x_i, y_i \leq 10^6$) — the locations of the volunteers.

Output

For each test case, print the earliest time at which the vision of the volunteer standing at point $(0, 0)$ will be blocked, rounded to 3 decimal places.

Examples

squares.in	Standard Output

2
1 2
0 0
1 2
10 10
	0.000
	10.000

Problem L. Vowely Pairs

Program:	`vowely.(c|cpp|java)`
Input:	`vowely.in`
Balloon Color:	`Orange`

A pair of strings is called "vowely" if the two strings in the pair contain the same number of vowels*.

It is a new term invented by Dr. Rick Sanchez during one of his sessions for the young computer scientists. At the end of the session, Morty Smith, a fresh graduate, asked the following question:

If we have a sequence of strings $S_1, S_2, \ldots, S_N$, and we considered only the ordered pairs (S_i, S_j), where $i < j$, how many distinct vowely pairs can be constructed out of these pairs?

For example, in the following sequence of four strings: "car", "the", "age", "in", the answer to Morty's question would be 3 pairs which are: ("car", "the"), ("car", "in"), and ("the", "in").

As usual, Dr. Sanchez made fun of Morty and asked the attendees to answer this question as a homework. If you were one of the attendees, can you write a program to answer this question?

Input

The first line of the input contains an integer T $(1 \leq T \leq 10)$ — the number of test cases.

Each test case starts with a line containing an integer N $(1 \leq N \leq 10^5)$ representing the number of strings in the given sequence. Then, N lines follow, each containing a string S $(1 \leq |S| \leq 26)$** which consists of lowercase letters.

Output

For each test case, print the answer to Morty's question.

Example

vowely.in	Standard Output
1 4 car the age in	3

Note

* Vowels are the letters: 'a', 'e', 'i', 'o', and 'u'.

** $|S|$ represents the length of the string S.