Problem A. Always Return a Value

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

In CUC (Completely Useless Community) the new programming language C∼ was invented. Everybody agrees that it is the worst programming language in the world.
Firstly, this programming language supports only operations with signed 32-bit integer numbers. Secondly, it has a very strange syntax. For example, a function definition in C∼ looks as follows:

```
int <Function_Name>(int <Argument1>, int <Argument2>)
{
    <Operators>
}
```

Here `<Function_Name>` is a string consisting of lowercase and uppercase Latin letters, underscore characters ("_") and digits. The first character of `<Function_Name>` cannot be a digit. `<Argument1>` and `<Argument2>` are single lowercase or uppercase Latin letters.

```
<Operators> ::= {<Operator>}
<Operator> ::= <Return_Operator> | <If_Operator>
<Return_Operator> ::= "return "<Integer>";"
<If_Operator> ::= "if ("<Predicate>")"<Statements>["else"<Statements>]
<Statements> ::= "{"<Operators>"}" | <Operator>
<Predicate> ::= <And_Predicate> | <Or_Predicate>
<And_Predicate> ::= <Condition>[" && "<And_Predicate>]
<Or_Predicate> ::= <Condition>[" || "<Or_Predicate>]
<Condition> ::= <Argument>" "<Comparison_Operator>" "<Integer>
<Comparison_Operator> ::= "<" | ">" | "==" | "<=" | ">=" | "!="
<Argument> ::= <Argument1> | <Argument2>
<Integer> ::= ["-"]<Digit>{<Digit>}
<Digit> ::= "0" | "1" | "2" | "3" | "4" | "5" | "6" | "7" | "8" | "9"
```

Here "||" denotes the logical operator "or" and "&&" denotes the logical operator "and". Also, note that one `<Predicate>` cannot contain more than one type of logical operators.
Sometimes CUC programmers use this language to write their quite complex but still useless programs. They use functions to make their source codes even more ugly. Unfortunately, CUC programmers are very careless, so in some cases they forget to return a value.
You are given a function implementation in C∼ language. You are to write a program that determines whether this function always returns a value, not always returns a value or doesnt return a value at all.

Input

The input file contains several (not more than ten and at least one) test cases consisting of source codes of some functions in C∼. There is a single blank line after each test case. All functions have two integer parameters and return integer values. You may assume that functions have no syntax errors.
The input is terminated with a line containing a single string "`int main() {}`". There are no two functions with equal names. There is no function named "main". All numbers in the input do not exceed $2 \cdot 10^9$ by their absolute value and do not have leading zeros. There can be arbitrary number of whitespace characters (tab characters, newlines, spaces) in a function body in the beginning of each line, after "`else`" and after characters "{", "}", "`)`" and "`;`". The input file doesnt exceed 1500 bytes limit.

Output

For each function in the input file print its name, a single space and then print "`always returns a value`" if this function returns a value with any values of its parameters, "`doesnt always return a value`" if only for some values of its parameters this function returns a value and "`doesnt return a value`" if it never returns a value.
Print a blank line after the output for each test case.
Follow the format of the sample output.

Examples

standard input	standard output

int sample1(int x, int y)
{
    if (x > 8) return 4;
    else if (y > 0) return 1;
        else return 2;
}
 	
int sample2(int a, int b)
{
    if (a > 44 || b < 18)
    {
        return 0;
    }
}
 	
int main() {}
	sample1 always returns a value
	
	sample2 doesn't always return a value

Note

Items repeating 0 or more times are enclosed in curly brackets ("{", "}") in C∼ syntax description.

Problem B. Cheese

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

There are N pieces of cheese, numbered from 1 to N. Each piece has cost V_i and mass M_i. Your task is to divide them into two groups having equal total masses and equal total costs. To achieve this goal you can make two or less cuts.

A cut is the following procedure. You choose some piece, say, i-th one, you choose some proportion p ($0 \le p \le 1$) in which to make a cut; then you cut the i-th piece into two pieces with masses $M_i \cdot p$ and $M_i \cdot (1-p)$ and costs $V_i \cdot p$ and $V_i \cdot (1-p)$. The first of the resulting pieces becomes the new i-th piece, the second one becomes the piece number $(N+1)$, and N increases by one. You can cut a piece that is a result of previous cut.

Input

The first line contains integer N. The next N lines contain space-separated pairs of integers V_i, M_i. $1 \le N \le 10^5, 1 \le V_i, M_i \le 10^6$.

Output

If there is no solution, write "NIE" on the only line of the output. Otherwise, print any solution in the following format. On the first line output the number of cuts C ($0 \le C \le 2$). On each of the next C lines output integer i and real number p ($0 \le p \le 1$) describing the cut. On the next line output one integer L — the number of pieces in the first group. On the next line output L integers k_i — the indices of pieces of the first group ($1 \le k_i \le N+C$, all k_i must be different). All other pieces are considered to be in the second group. Degenerate cuts (with $p = 0$ or $p = 1$) are allowed. Absolute or relative difference between total masses (and costs) of groups must not exceed 10^{-9}.

Examples

standard input	standard output
4 1 3 7 2 10 3 4 4	0 2 1 3

Problem C. Colored Graph

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Consider a regular polygon (i. e. a polygon with all sides having the same length and all angles having the same value) with vertices numbered in clockwise order by integer numbers from 1 to N. Let its vertices correspond to vertices of a complete undirected graph, and its diagonals and edges correspond to edges of this graph. Also, each edge of this graph is colored either black or white.
Let's call a spanning tree of this graph valid if all its edges have the same color and no two of them intersect (except for inevitable intersections in vertices).
Given the coloring, find any valid spanning tree of this graph. The coloring of the edges is given in the following manner. Let the integer $C_{a,b}$ denote the color of the edge between a and b (0 means white, 1 means black, $C_{a,b} = C_{b,a}$). For some pairs (a, b) the color $C_{a,b}$ is given explicitly. The colors of all other edges can be found using the following formula:

$$C_{a,b} = \begin{cases} 0, & \text{if } (\min(a,b) \cdot X + \max(a,b) \cdot Y) \mod Z < P_a + P_b \\ 1, & \text{otherwise} \end{cases}$$

where X, Y, Z and all P_i are given.

Input

The first line contains a single integer N ($3 \le N \le 5 \cdot 10^5$).
The second line contains the number of edges E ($0 \le E \le 2 \cdot 10^5$) of edges for which the color is given explicitly.
The next E lines contain three integers each: a, b, c ($1 \le a, b \le N$, $0 \le c \le 1$), which means that $C_{a,b} = c$.
The next line contains integers X, Y and Z. The next N lines contain integers P_i.
Here, $0 \le X, Y, P_i \le 10^{11}$; $1 \le Z \le 10^{11}$.

Output

Output $N - 1$ lines, describing the edges of a spanning tree. Each line should contain two integers — the vertices to connect with an edge. If no solution exists, just print "`No solution`". If multiple solutions exist, output any of them.

Examples

standard input	standard output
4 1 2 4 0 13 17 23 5 10 7 3	1 2 2 4 3 2

Problem D. Cross-section

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

You are given a polygon (not necessarily convex), boundary of which has neither any self-intersections nor self-touchings. Also you are given a point. Consider a line containing the given point and having random orientation, i. e. an angle a line makes with the positive x axis is uniformly distributed on $[0, \pi)$. The intersection of this line and the given polygon is a set (possibly empty) of segments. You need to find the expected total length of these segments (total length of zero segments is zero).

Input

First line contains three integers: N, x_0, y_0 — the number of vertices of the polygon and the coordinates of the given point. Each of the next N lines contains two integers: x and y — the coordinates of the corresponding vertex. Vertices are given in order of either clockwise or counter-clockwise traverse of the polygon.
It is guaranteed that $3 \le N \le 10^3$ and $-10^6 \le x_0, y_0, x, y \le 10^6$.
For better numerical stability, point (x_0, y_0) will have a distance of either 0 or at least 0.001 to each line passing through a side of the polygon.

Output

The only line should contain one real number — the expected total length with absolute or relative error not exceeding 10^{-6}.

Examples

standard input	standard output
3 1 2 2 2 1 2 1 3	0.396757510511804768273662801945 84

Problem E. Grouping by Prefixes

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

You are given a set of N strings $S = \{S_1, S_2, \ldots, S_N\}$.
Consider a set of M nonempty strings $P = \{P_1, P_2, \ldots, P_M\}$ such that set S can be explicitly divided into M subsets $T_1, T_2, \ldots, T_M$ with the following properties:

1. $i = 1..M : T_i = \{S_j \mid j \in \{1, 2, \ldots, N\} \wedge P_i \sqsubseteq S_j\}$
2. $i = 1..M : T_i \neq \varnothing$
3. $i = 1..M, j = 1..M, i \neq j : T_i \cap T_j = \varnothing$
4. $\bigcup_{i=1}^{M} T_i = S$

Where $A \sqsubseteq B$ denotes that A is a prefix of B.
You are to find the number of such sets P modulo $10^9 + 7$ and output any of them if at least one such set exists.

Input

The first line of the input gives two integers N and M ($1 \le M \le N \le 2000$) — the cardinalities of sets S and P respectively.
Then N lines follow, each line contains a single nonempty string consisting of lowercase Latin letters. Length of each string is not greater than 200.

Output

Write the number of different sets P modulo $10^9 + 7$ to the first line of the output.
If the number of ways is positive, output M lines containing the elements of any set of M nonempty strings P, satisfying the given conditions.

Examples

standard input	standard output
4 2 aba cab cad abcd	4 a ca

Problem F. Inserting Lines

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	256 mebibytes

Consider a two-dimensional array of pairs of integers $A_{x,y}$, infinite in all four directions. Initially each element has its coordinates as its value, i. e. $A_{x,y} = (x, y)$. You need to process Q queries numbered from 1 to Q. Each query, say i-th, can be of one of the following types:

1) Insert a (infinite) row at $y = k$, shifting rows with $y \geq k$. The elements of the inserted line have values (x, i). More formally, let B denote array A after processing the query, then adding a row at $y = k$ means that (note that i is a number of the current query)

$$B_{x,y} = \begin{cases} A_{x,y}, & y < k \\ (x, i), & y = k \\ A_{x,y-1}, & y > k \end{cases}$$

2) Insert a column at $x = k$. Defined similarly to the previous query type:

$$B_{x,y} = \begin{cases} A_{x,y}, & x < k \\ (i, y), & x = k \\ A_{x-1,y}, & x > k \end{cases}$$

3) Get the value of $A_{x,y}$ for given x and y.

Input

The first line of the input contains integer Q ($1 \leq Q \leq 10^5$), the number of queries. Q queries follow. Let (p, q) be the answer to the latest query of type 3, or both zero if there was no type 3 query yet. Each query has one of the forms:

1) 1 k_0 — insert a row at $k = k_0 + p + q$.
2) 2 k_0 — insert a column at $k = k_0 + p + q$.
3) 3 x_0 y_0 — get $A_{x,y}$, where $x = x_0 + p + q$, $y = y_0 + p - q$ (note the "−" sign). Remember that this query also assigns its result to (p, q).

For each query: $-10^7 \leq k, x, y \leq 10^7$.

Output

For each query of the third type output a line containing two integer numbers separated by a space, the value of $A_{x,y}$ at the moment of the query.

Examples

standard input	standard output
5 3 1 2 3 1 2 2 -6 1 -3 3 -6 -1	1 2 4 1 -1 4

Problem G. Paths in Tree

Input file:	*standard input*
Output file:	*standard output*
Time limit:	2 seconds
Memory limit:	**512** mebibytes

Consider a tree with n vertices. You have to select at most k simple paths without intersections by vertices. The total length of these paths should be the maximal possible. In this problem, the length of a path is the number of vertices in it.

Input

Each input contains one or more test cases.

Each test case starts with a line containing two integers n and k ($1 \le n \le 10\,000$, $1 \le k \le 500$, $k \le n$). It is followed by $n - 1$ lines which contain pairs of integers from 1 to n: edges of the tree. The sum of all n does not exceed 10 000.

The input ends with pair $n = 0$, $k = 0$. This pair should not be considered a test case.

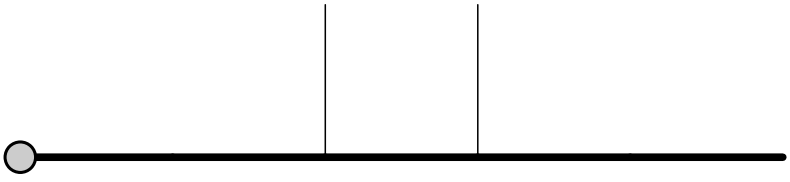
Output

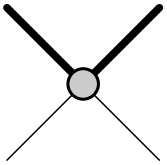
For each test case, print the total length of selected paths on a separate line (it is an integer from 0 to n). On the next line, print integer x ($0 \le x \le k$): the number of paths in the optimal answer you found. Next x lines should contain pairs of integers from 1 to n: endpoints of paths in the optimal answer you found. Endpoints of a single path may coincide: in this case, the path consists of exactly one vertex. Note that each simple path in a tree is uniquely determined by its endpoints. If there are several optimal answers, you may output any one of them.

Example

standard input	standard output
8 1 1 2 2 3 3 4 4 5 5 6 3 7 4 8 5 2 1 2 1 3 1 4 1 5 0 0	6 1 6 1 4 2 2 3 4 4

Example explanation





Problem H. Restoring Points

Input file:	*standard input*
Output file:	*standard output*
Time limit:	15 seconds
Memory limit:	256 mebibytes

Eva has marked 11 distinct points with integer coordinates on the plane. Then, she calculated the squared distance between each pair of distinct points. After that, she wrote down the 55 numbers she got in the order she wanted, and then erased the marked points. Finally, she showed the numbers to Andrew. Help Andrew restore the points on the plane.

Recall that the squared distance between points (x_1, y_1) and (x_2, y_2) on the plane is equal to

$$(x_2 - x_1)^2 + (y_2 - y_1)^2.$$

Input

The first line of input contains 55 integers separated by spaces: the pairwise squared distances between sought points. It is guaranteed that these are squared distances between 11 distinct points, and these points' coordinates are integers not exceeding 5 000 by absolute value.

Output

Print 11 lines containing the coordinates of points. Coordinates of a point are two integers: x-coordinate and y-coordinate. Coordinates of each point should be printed on a separate line and separated by a space. The collection of squared distances between the points you output should be the same as the collection of numbers in the input. You can output any answer in which coordinates of all points do not exceed 10 000 by absolute value.

Example

standard input	standard output
1 4 9 16 25 36 49 64 81 100 1 4 9 16 25 36 49 64 81 1 4 9 16 25 36 49 64 1 4 9 16 25 36 49 1 4 9 16 25 36 1 4 9 16 25 1 4 9 16 1 4 9 1 4 1	0 0 1 0 2 0 3 0 4 0 5 0 6 0 7 0 8 0 9 0 10 0

Example explanation

In the example above, distances are given in the following order:
$(1, 2)$, $(1, 3)$, $(1, 4)$, ..., $(1, 11)$;
$(2, 3)$, $(2, 4)$, ..., $(2, 11)$;
...;
$(9, 10)$, $(9, 11)$;
$(10, 11)$.
All points lie on the same line.

The numbers in the example input occupy multiple lines only for readers' convenience. In fact, there is only one line with 55 numbers in each input.

Problem I. Security Policy

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

It was discovered that the life is possible on the planet Gliese 581. People think that there are a lot of countries on this planet. In 2203 the humanity travelled to this planet and discovered the following:
- Government type of each country is either "Monarchy", "Democracy" or "Republic".
- Religion of each country is either "Christianity", "Islam" or "Buddhism".
- Main resource of each country is either "Gas", "Oil" or "Coal".
- Primary production of each country is either "Cars", "Electronics" or "Textile".
- The most popular sport of each country is either "Football", "Hockey" or "Volleyball".

Every country on this planet has a shape of convex non-degenerate polygon on a map. Map of the Gliese 581 is a Cartesian plane. No two polygons overlap. Countries that have more than one common point are considered to be neighbors.
Different kinds of unions exist on the Gliese 581. There is exactly one union for every government type, religion, main resource type, primary production and the most popular sport type.
Each country must decide which union to join. The particular country must join the exactly one union according to its government type, religion, main resource type, primary production or the most popular sport type. So there are exactly five different options for every country — exactly five different unions to choose from. There is no limit for the number of countries in the particular union. Note that some unions may remain empty.
Due to the "Security policy" document it is forbidden for two neighboring countries to join the same union, because they can form the battle alliance in this case.
Inhabitants of the Gliese 581 asked you for help. You are to help every country to choose which union to join. Write a program that for given map and country descriptions will choose the possible distribution.

Input

The first line of input contains the integer number N ($2 \le N \le 500$) — the number of countries on the map. The i-th of the following N lines describes the i-th country.
Each of them begins with a string consisting of five characters. Each of the characters describes the features of the corresponding country in order they were enumerated above (with the respective first capital letters, for example "`M`" is for "Monarchy"). After that there is an integer number K_i ($3 \le K_i \le 50$) — the number of vertices in the corresponding polygon. The line ends with K_i pairs of numbers X_j and Y_j ($-10^4 \le X_j, Y_j \le 10^4$). They describe the vertices of the convex polygon in the clockwise order. No three consecutive points lie on the same line.

Output

If it is impossible to avoid the creation of battle alliances, output "`No solution`". Otherwise output a line of length N consisting of characters "`G`" (Government), "`R`" (Religion), "`M`" (Main resource), "`P`" (Production) and "`S`" (Sports). The i-th character should describe which of the five possible unions the i-th country should join. For example, if the i-th character is "`G`" than the i-th country should join the union according to its government type, etc.

Examples

standard input	standard output
4 MCOTH 4 2 2 2 0 0 0 0 2 DCOTH 3 2 1 3 0 2 0 RIGEV 3 2 2 3 1 2 1 RBCEF 3 3 0 1 -1 1 0	GPRS

Problem J. Encryption

Input file:	*standard input*
Output file:	*standard output*
Time limit:	1 second
Memory limit:	256 mebibytes

Petr has invented a new encryption algorithm. His algorithm allows to encrypt positive integers.
The idea is very simple. To encode integer X, one needs to calculate the sum of all the numbers not greater than X that are coprime to X. This sum is the code for X.
Your task is to show Petr that his algorithm is not so cryptographically secure as he thinks it is.

Input

The first line of the input gives the number of test cases, T ($T \leq 1000$). T lines follow. Each line contains a single positive integer N ($N \leq 10^{18}$) — the encoded integer.

Output

For each test case output one line containing the original integer that was encoded. If there are several such integers, output any of them. You may assume that for the given test data the answer always exists.

Examples

standard input	standard output
2 21 20	7 10

Note

Two integers are said to be coprime if their greatest common divisor is 1.

Problem K. Triangle and Segment

Input file:	*standard input*
Output file:	*standard output*
Time limit:	6 seconds
Memory limit:	256 mebibytes

You are given a set of points on a Cartesian plane. No three points lie on the same line.
You are to find the number of ways to place a triangle and a segment on this plane, so that the following conditions are satisfied simultaneously:

1. Vertices of the triangle and endpoints of the segment belong to the given set.
2. The border of the triangle doesn't have any common points with the segment.
3. Sides of the triangle and the segment have nonzero length.

Two placements are considered different if there is a point that is occupied by a vertex of the triangle or by a endpoint of the segment in the first placement and that is free in the second one, or, alternatively, if there is a point occupied by a vertex of the triangle in the first placement and by a endpoint of the segment in the second one.

Input

The first line of the input gives the number of points in the set, N ($5 \le N \le 250$).
Then N lines follow, each line contains two integers X_i, Y_i ($|X_i|, |Y_i| \le 10^4$) — coordinates of the i^{th} point.

Output

Write to the output the single integer that is the answer to the problem.

Examples

standard input	standard output
5 9 6 1 4 2 5 6 1 3 2	5