

Problem A. Apple

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

Let's define a figure as an "apple" if it can be represented as a combination of a single closed polygonal line and a single open polygonal line, both originating from the same vertex, without self-intersections. The closed line must have at least 3 vertices, while the open line must have at least 2 vertices, and they must not share any points other than the starting vertex.

Given the number of vertices and a set of segments connecting these vertices, determine if it is possible to arrange the vertices of the polygonal lines in the plane such that they form the "apple" figure.

Input

The first line contains two integers n and m ($1 \leq n, m \leq 100$) — the number of vertices and the number of connections between them, respectively. Each of the next m lines contains two integers a_i and b_i ($1 \leq a_i < b_i \leq n$, $i = 1, 2, \dots, m$) indicating that the vertices a_i and b_i are connected by i -th segment. Each pair of vertices is connected by at most one segment.

Output

If the given figure can be classified as an "apple", output "Yes"; otherwise, output "No".

Examples

standard input	standard output
9 9 1 2 2 3 3 5 5 6 4 6 4 7 1 7 7 9 8 9	Yes
5 5 1 2 2 3 1 3 1 4 1 5	No
4 4 1 2 2 3 1 3 1 4	Yes
5 4 1 2 2 3 1 3 1 4	No

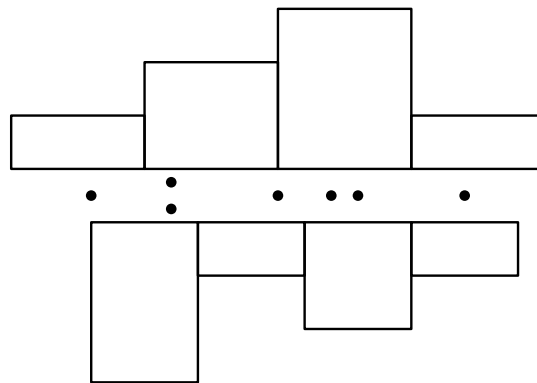
Problem B. Bar charts

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

Let's define a *data set* as a non-decreasing sequence of integers $A = (a_1, a_2, \dots, a_n)$.

A *bar chart* is a graphical representation of a data set, consisting of several rectangles of varying heights, positioned adjacent to each other. The bar chart $H_{k,b,s}(A)$ consists of k rectangles, where the height of the i -th rectangle, denoted as h_i , equals the count of values from the data set A that fall within the half-interval $[b + (i - 1)s, b + is)$, for $1 \leq i \leq k$. The parameters k , b , and s must be chosen such that all values from the data set are taken into account; specifically, it must hold that $b \leq a_1$ and $a_n < b + ks$.

It is clear that different values of k , b , and s can produce various bar charts for the same data set A . For instance, for the data set $A = (4, 7, 7, 11, 13, 14, 18)$, the bar charts $H_{4,1,5}(A)$ and $H_{4,4,4}(A)$ look as follows:



You are given two bar charts. Your task is to determine whether they could originate from the same data set.

Input

The first and second lines describe the first bar chart. The first line contains the parameters k , b , and s ($1 \leq k, b, s \leq 100$), while the second line contains k integers, where the i -th integer reflects the height of the i -th rectangle h_i ($0 \leq h_i \leq 100, 1 \leq i \leq k$).

The third and fourth lines provide the description of the second bar chart in the same format.

Output

If both bar charts could have been generated from some data set, output “Yes”. Otherwise, output “No”.

Examples

standard input	standard output
4 1 5 1 2 3 1 4 4 4 3 1 2 1	Yes
4 1 5 1 1 4 1 4 4 4 3 1 2 1	No

Problem C. Constructor

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

The well-known manufacturer of children's construction sets, OLEG, is going through some tough times. For reasons yet to be determined, all factories producing the parts for the sets have stopped. While the situation with the factories is being clarified, the company's management needs to release a new batch of sets, which will have to be assembled from the currently available parts.

There are n types of construction parts. For each type i , the mass m_i and the count c_i of the parts are known. The management wants to assemble as many *identical* sets as possible, such that the total mass of the parts included in each set is at least $m_{\min}$ and at most $m_{\max}$. Two sets are considered identical if the number of parts of each type included in both sets is the same.

Determine the maximum number of sets that can be produced.

Input

The first line contains the integers n , $m_{\min}$, and $m_{\max}$ ($1 \leq n \leq 50$, $1 \leq m_{\min} \leq m_{\max} \leq 10^4$). The next n lines contain numbers m_i and c_i for each type i ($1 \leq m_i \leq 10^4$, $1 \leq c_i \leq 10^6$, $1 \leq i \leq n$).

Output

Output a single integer — the maximum number of sets that can be produced.

Example

standard input	standard output
3 12 13 3 8 4 6 7 9	4

Note

In the example, one possible set can be constructed by taking two parts of type 1 and one part of type 3. The total mass of the set would be $3 + 3 + 7 = 13$, and from the available parts, a total of 4 such sets can be assembled.

Problem D. Dinner Party

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

Today is a celebration, and Aunt Margaret is expecting many guests for a dinner party. For this grand feast, Uncle Henry brought n square tables and m chairs from his estate.

Aunt Margaret wants to arrange the square tables of same size to form one or several large rectangular tables, and place the chairs along the perimeter of each large table. It is essential to use all n tables to maximize the food serving area, and all m chairs to ensure there's enough seating for all guests. Chairs must be placed in every available spot, which means the total perimeter of all large rectangular tables must equal the number of chairs.

For example, given $n = 10$ and $m = 18$, the available square tables can be combined into two large tables measuring 2×4 and 1×2 , with a total perimeter of exactly 18. Another possible arrangement for these parameters could be 2×3 and 2×2 .

Help Margaret arrange the furniture accordingly.

Input

The first line contains an integer T — the number of test cases ($1 \leq T \leq 200$).

Each of the next T lines contains two integers n and m ($1 \leq n \leq 1\,000$, $1 \leq m \leq 4\,000$).

Output

For each test case, output “Yes”, if it is possible to arrange the tables as needed, or “No” if it is not possible. If the answer is positive, print the number z — the total number of large tables, followed by z lines each containing two integers a_i and b_i — the dimensions of each large table ($1 \leq i \leq z$).

If multiple solutions exist, output any valid arrangement.

See the example for a clearer understanding of the output format.

Example

standard input	standard output
4	Yes
10 18	2
6 12	1 2
5 20	2 4
1 2	No
	Yes
	5
	1 1
	1 1
	1 1
	1 1
	1 1
	No

Problem E. Expressions of Dice

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

This is an interactive problem.

In this task, you need to form any true arithmetic statement using symbols available on dice. You have an unlimited number of dice of 6 available types, with the following symbols on their faces:

- **Type 1:** = < > != <= >=
- **Type 2:** 4 + - (()
- **Type 3:** 0 / / / 8 +
- **Type 4:** 2 3 4 5 *)
- **Type 5:** + - * / 1 9
- **Type 6:** 6 7 + - ()

The symbols you can use are determined as follows. You first need to choose one die of any type, roll it, and find out the resulting symbol. After that, you choose the next die of any type, roll it, obtain the next symbol, and so on. Once you decide that you have enough symbols, you need to create one **any true** statement from **all** the rolled symbols that satisfies the following grammar:

$$\begin{aligned}\langle \text{Statement} \rangle &\longrightarrow \langle \text{Expression} \rangle \langle \text{RelationalOperator} \rangle \langle \text{Expression} \rangle \\ \langle \text{RelationalOperator} \rangle &\longrightarrow = | < | > | != | <= | >= \\ \langle \text{Expression} \rangle &\longrightarrow \langle \text{Term} \rangle | \langle \text{Expression} \rangle + \langle \text{Term} \rangle | \langle \text{Expression} \rangle - \langle \text{Term} \rangle \\ \langle \text{Term} \rangle &\longrightarrow \langle \text{Factor} \rangle | \langle \text{Term} \rangle * \langle \text{Factor} \rangle | \langle \text{Term} \rangle / \langle \text{Factor} \rangle \\ \langle \text{Factor} \rangle &\longrightarrow \langle \text{Number} \rangle | (\langle \text{Expression} \rangle) \\ \langle \text{Number} \rangle &\longrightarrow \langle \text{Digit} \rangle | \langle \text{NonZeroDigit} \rangle \langle \text{Digits} \rangle \\ \langle \text{Digits} \rangle &\longrightarrow \langle \text{Digit} \rangle | \langle \text{Digits} \rangle \langle \text{Digit} \rangle \\ \langle \text{NonZeroDigit} \rangle &\longrightarrow 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 \\ \langle \text{Digit} \rangle &\longrightarrow 0 | \langle \text{NonZeroDigit} \rangle\end{aligned}$$

Please note that the division referenced here is specifically **real division**, not integer division. It is prohibited to multiply by 0, divide by 0, and also to divide 0.

Interaction Protocol

Your program must output one of two commands:

1. “<num>”, where <num> — the type of the die being rolled. In response, the jury’s program will output a single line with the resulting symbol, chosen randomly and uniformly.
2. “0 <ans>”, where <ans> — your composed statement.

The number of rolls must not exceed 1 000.

Examples

standard input	standard output
) (+ 5 - 4 (7 5) > 0 - 5	2 6 3 4 5 4 6 6 4 2 1 3 2 4 0 5-7>4-((5+50))
/ 2 != / 4 9 4	3 4 1 3 4 5 4 0 2/4!=4/9

Problem F. Flight Trip

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

This is an interactive problem.

Let's consider the planet as a perfect sphere with an equator length of 40 000 km, where the coordinates are defined in the standard way: latitude ϕ ($-90^\circ \leq \phi < 90^\circ$) and longitude λ ($-180^\circ \leq \lambda \leq 180^\circ$). Moving north increases the latitude, while moving west increases the longitude.

The planet is divided into 24 time zones, where the i -th time zone ($i = -11, -10, \dots, 12$) covers the longitude interval $[15i - 7.5, 15i + 7.5)$ degrees. Time zones determine the difference in local time across various regions of the planet, specifically in the time zone with number i , it is exactly i hours later than in the time zone with number 0. The North and South Poles fall under the time zone with number 0.

You are on a helicopter that has a turn speed of $1\,440^\circ/\text{min}$ and a travel speed of $3\text{ km}/\text{min}$, in an unknown location on the planet and flying in some direction. Your goal is to reach a specified point on the planet, using only the information about the local time of the regions you will fly over.

Interaction Protocol

First, the jury's program outputs two real numbers with 10 decimal places —latitude and longitude of the target point. You need to approach it within a distance of 10^{-5} km.

In the next line, the jury's program outputs the local time in the format: "<day of the week> <hours>:<minutes>", where hours and minutes are exactly two digits long, and the day of the week is a string of 3 characters, which can take one of the following values: "mon" (Monday), "tue" (Tuesday), "wed" (Wednesday), "thu" (Thursday), "fri" (Friday), "sat" (Saturday), "sun" (Sunday). The initial moment of time in each test is always Monday 0 hours 0 minutes in the local time of your starting position.

After that, your program should make requests consisting of two real numbers a and d ($-360 < a < 360$, $0 \leq d < 40\,000$), meaning that you will first turn a° counterclockwise and then fly straight for a distance of d km. The angle of turn in degrees and the distance in kilometers for your next action should be provided. The output numbers must represent a decimal fraction with no more than 12 decimal places. For each request, the jury's program will output either a string containing the local time in the format described above or the string "--- 00:00", indicating that you have reached the target point with the required accuracy, and your program should terminate.

The number of moves must not exceed 25 000.

Example

standard input	standard output
-15.5837712391 -118.7986557214	
mon 00:00	
mon 04:33	0 1000
mon 04:33	90 0
sat 13:50	10.101010 25252.52525252
--- 00:00	123 4567.89

Problem G. Glasses of Solutions

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

You are given n glasses containing solutions of salt. For each solution, the total mass and the mass of the salt contained within it are known. The objective is to determine the number of non-empty subsets of these n glasses such that if their contents are combined into an empty glass, the resulting solution will achieve a specified mass fraction of salt.

Input

The first line contains three integers n , a , and b ($2 \leq n \leq 35$, $0 \leq a \leq 10\,000$, $\max(a, 1) \leq b \leq 10\,000$, $\gcd(a, b) = 1$) — the number of glasses containing solutions, the numerator, and the denominator of the fraction $\frac{a}{b}$ representing the desired mass fraction of salt, respectively.

The following n lines each contain two integers m_i and t_i ($0 \leq m_i \leq 10\,000$, $\max(m_i, 1) \leq t_i \leq 10\,000$) — the mass of salt and the total mass of the i -th solution ($i = 1, 2, \dots, n$).

Output

Output a single integer — the number of ways to achieve a solution with a mass fraction of salt $\frac{a}{b}$.

Examples

standard input	standard output
5 1 2 1 2 1 2 1 2 1 2 1 4	15
2 0 1 0 1 1 1	1

Problem H. Hamburgers

Input file: **standard input**
Output file: **standard output**
Time limit: 3 seconds
Memory limit: 256 megabytes

In a city, there are several cafes that offer a wide range of various hamburgers.

Each hamburger, as well as the preferences of each customer, will be characterized by a list of additional ingredients that may be included in the hamburgers. Let's say a hamburger will please a customer if it contains all the ingredients that the customer likes. A customer will be satisfied with a cafe visit if there is a hamburger on the menu that they will enjoy.

Write a program to choose a cafe for a group of friends such that as many people from the group as possible are satisfied.

Input

The first line contains a single integer n ($1 \leq n \leq 1\,000$) — the number of groups of friends.

Following this, there are n blocks of data, each describing a group. The first line of each block contains a single non-negative integer k_i — the number of people in the i -th group ($i = 1, 2, \dots, n$). Next, each of the k_i non-empty strings describes the preferences of a specific person in this group. Each string consists of at most 6 distinct lowercase Latin letters, with each letter representing one of the 26 possible additional ingredients.

The total number of people across all groups does not exceed 50 000.

Following this, there is a line containing a single integer m ($1 \leq m \leq 1\,000$) — the number of cafes. Next, there are m blocks of data, each describing a cafe. The first line of each block contains a single non-negative integer l_j — the number of hamburgers on the menu of the j -th cafe ($j = 1, 2, \dots, m$). Next, there are l_j non-empty strings, each containing up to 6 characters and describing the additional ingredients of the hamburgers in the format mentioned above.

The total number of hamburgers across all cafes does not exceed 50 000.

Output

For each group, output on a separate line a single integer from 1 to m — the number of the most suitable cafe. If there are multiple such cafes, output the **minimum** of these.

Example

standard input	standard output
5	1
1	1
a	2
3	1
a	2
ba	
cb	
3	
vba	
d	
ba	
3	
ca	
da	
da	
3	
as	
ba	
af	
2	
4	
abc	
abcd	
a	
a	
2	
abcdf	
abcdv	

Problem I. Intricate Path

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

In this problem, your task is to construct a polygon with obstacles for a robot. The robot must enter the polygon through one of its corners, collide with obstacles exactly k times, and then exit.

The robot is programmed with exactly k commands, which instruct it to either turn left or turn right. The robot can only move forward, and upon each collision, it executes the next command in its program, turning left or right as specified.

The polygon must represent a rectangular grid where both the robot and an obstacle occupy single cells, and the robot can only move in directions parallel to the grid's edges. The robot collides with an obstacle if it is in a neighboring cell in the direction of the robot's movement.

Input

The first line contains a single string s ($1 \leq |s| \leq 50$) — the robot's program, consisting solely of the characters "L" and "R", which represent left and right turns, respectively.

Output

Choose the dimensions of the polygon, n ($1 \leq n \leq 50$) and m ($1 \leq m \leq 50$), and output n lines containing m characters: "@", ".", and "#", which represent the following:

1. The symbol "." denotes an empty cell;
2. The symbol "#" denotes a cell containing an obstacle;
3. The symbol "@" indicates the robot's initial position, facing right. This symbol must appear exactly once and must be located in either the **first** or **last** row of the **first** column.

Examples

standard input	standard output
L	@#
LLLLLL	###### ####.# #....# ###.## ###.## @...##
RLLRLLRRLRLRLLRLLRLL	@## ..# ###
LRRLLLLRRRLRRRRR	#... ...#.....# .#.....#..#.... ..#.....#...#.. @.#..#.....

Problem J. Jumping Through Hyperspace

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

Year 2333. After the discovery of Hyperspace, interplanetary travel has become a commonplace activity. By this time, humanity has already colonized n distinct planets. Each planet has a hyperport that facilitates travel to other planets using hyperjumps.

Due to the peculiarities of spacetime fabric, the duration of a hyperjump is not constant. It depends on the time the journey begins and the type of hyperjump. Specifically, for hyperjump type i and start time s_i , the total travel time is given by the equation $T = ((a_i s_i + b_i) \bmod c_i) + d_i$, where a_i , b_i , c_i , and d_i are parameters corresponding to the i -th type of hyperjump.

You start on planet number 1 and wish to reach planet number n . What is the minimum amount of time it will take to accomplish this?

Input

The first line contains the integers n , m , and s — the number of planets, the number of hyperjump types, and the current time ($2 \leq n \leq 2000$, $1 \leq m \leq 50$, $1 \leq s \leq 2000$).

The next m lines consist of the i -th line containing the character z_i — the identifier of the i -th hyperjump type (a lowercase or uppercase Latin letter), followed by the integers a_i , b_i , c_i , and d_i — the parameters of the hyperjump of type i ($1 \leq a_i, b_i, c_i, d_i \leq 2000$). The hyperjump identifiers are unique.

Each of the subsequent n lines contains n characters. The j -th character in the i -th line corresponds to the hyperjump type that allows moving from planet i to planet j . If no hyperjump is available from planet i to planet j , the character will be “.”.

Output

Output a single integer — the minimum time required to travel from planet number 1 to planet number n .

If it is impossible to reach planet n from planet 1 using a sequence of one or more hyperjumps, output -1 .

Example

standard input	standard output
3 2 1 A 1 1 5 1 a 2 2 7 1 .A. ..a a..	6

Note

Explanation for the example: the cheapest way to jump from planet 1 to planet 2 is to jump at time 1 and arrive at time 4; when jumping from planet 2 to planet 3, it is optimal to wait for two time units, execute the jump in one time unit, and arrive at time 7.

Problem K. King's Island

Input file: `standard input`
Output file: `standard output`
Time limit: 1 second
Memory limit: 256 megabytes

The king wants to create an artificial island with the following properties: it must be a simple, non-self-intersecting polygon with n vertices, all vertices having integer coordinates, no three consecutive vertices lying on the same line, the lengths of all edges being integers, and no edge being parallel to any of the coordinate axes. Given the number n , construct such an island with coordinates that do not exceed an absolute value of 10^4 .

Input

The first and only line contains one integer n ($3 \leq n \leq 30$).

Output

Output n lines, each containing two integers x_i, y_i ($-10\,000 \leq x_i, y_i \leq 10\,000$) — the coordinates of the i -th vertex. The vertices should be output in a clockwise or counterclockwise order.

Examples

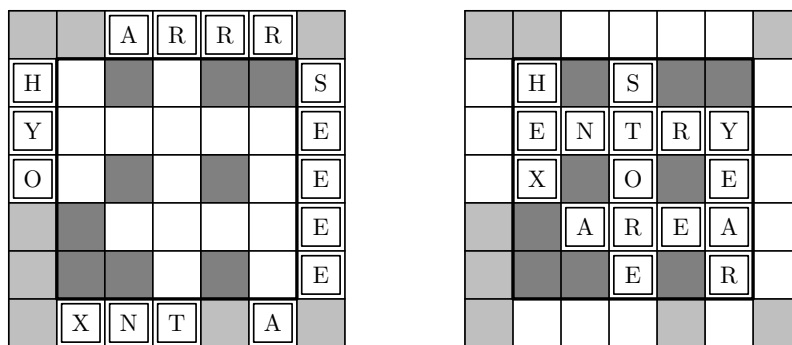
standard input	standard output
3	0 0 4 3 -20 21
4	0 3 4 0 0 -3 -4 0

Problem L. Lexica

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

The Lexica puzzle consists of a square grid divided into $n \times n$ cells. Some cells are active, while others are not, with all active cells forming a crossword grid. Letter pieces are placed along the edges of the square. The pieces on the left and right edges can be moved to active cells in the same row, while the pieces on the top and bottom edges can be moved to active cells in the same column. It is not allowed to place two pieces in the same active cell. The number of pieces is equal to the number of active cells. The goal of the game is to move all pieces to the active cells so that a valid crossword grid is formed. This state of the game is referred to as a solution.

For instance, the initial state of the puzzle and its solution are shown in the following illustration:



You are given a Lexica puzzle. It is known that on any two pieces, one of which can be moved horizontally and the other vertically, the letters are different. Your task is to count the number of distinct solutions to the puzzle. Two solutions are considered distinct if there exists at least one active cell where the corresponding solutions have differing letters.

Input

The first line contains the integer n ($4 \leq n \leq 13$). The following $(n + 2)$ lines provide the description of the puzzle. Each line consists of exactly $(n + 2)$ characters. Active cells are represented by the symbol “.”, pieces are represented by uppercase Latin letters, and all other cells are represented by the symbol “#”. Please refer to the example for a clearer understanding.

Output

Output a single integer — the number of distinct solutions.

Example

standard input	standard output
<pre> 5 ##ARRR# H.#.##S Y.....E O.##.E ##...E ###.#.E #XNT#A# </pre>	<pre> 768 </pre>

Problem M. Maximum Paths

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

You are given a complete binary tree consisting of n nodes, numbered with integers from 1 to n . For each node $i > 1$, its parent is the node numbered $\lfloor i/2 \rfloor$. Each edge in the tree is labeled with a digit from 1 to 9.

For each pair of nodes u and v , let $N(u, v)$ denote the sum of the digits that can be read on the edges when moving from node u to node v along the shortest path $p(u, v)$. If $u = v$, then $N(u, v) = 0$.

For a node v , let's define the set of *hanging* paths $H(v)$: these are the paths that include node v but do not contain its parent (for node 1, all paths containing it are considered hanging).

Let $M(v)$ denote the hanging path $p(u, w)$ (i.e., $p(u, w) \in H(v)$), which has the maximum value of $N(u, w)$. Such a path is referred to as a *maximum path*.

In this problem, you need to find the maximum path $p(u, w) = M(v)$ for each node v , sum the corresponding values of $N(u, w)$, and output the resulting sum for all v .

Input

The input consists of a single line containing two integers n and $seed$ ($1 \leq n \leq 10^7$, $0 \leq seed < 2^{31}$).

The digits written on the edges of the tree are generated using a linear congruential generator. Specifically, a sequence r is generated as follows:

$$r_1 = seed,$$

$$r_i = (1103515245 \cdot r_{i-1} + 12345) \bmod 2^{31}, \quad 2 \leq i \leq n.$$

Then, for each edge connecting nodes i and $\lfloor i/2 \rfloor$, the digit on that edge is computed as

$$a_i = ((r_i \gg 16) \bmod 9) + 1, \quad 2 \leq i \leq n.$$

Output

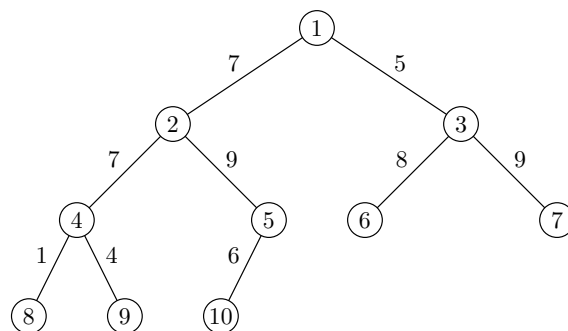
Output a single integer, which is the required sum.

Example

standard input	standard output
10 2017	102915

Note

The drawing below illustrates the tree generated by the sample input.



The values of $N(u, w)$ for the maximum paths $M(v)$ are as follows: 95796, 6974, 98, 41, 6, 0, 0, 0, 0, 0. The sum of these values is 102915.