2008 Peking University Campus Programming Contest

Problem Set

May 10, 2008

Problem A. Escort of Dr. Who How

Description

The notorious ringleader and cyber-criminal Derek Wu, better known as Dr. Who How, has been convicted for multiple abductions and cyber-frauds several minutes ago. He will soon be transferred from the court to the prison to serve a life sentence. Intelligence from undercover officers in Dr. Who How's outlawed gang indicates a possible forcible rescue operation by gang members in the case that Dr. Who How is convicted. To ensure a successful escort, besides adopting enhanced security measures, the police force further desires to minimize the escort time—the time that the motorcade of escort has to spend between departure from the court and arrival at the prison.

The complex traffic condition in the city imposes complications on the police force's escort effort. While the municipal government has agreed to temporarily shut down a few roads for exclusive police use, the shut-down time windows vary among roads and are generally narrow to avoid excessively disturbing the life of the citizens. The motorcade of escort must pass through any road entirely within its shut-down time window and not use any road that the municipal government has not agreed to shut down.

Given information about all roads available to the police force for the escort, determine the shortest escort time.

Input

The input contains a single test case. The first line contains four integers n, m, s and t ($2 \leq n \leq 100$; $0 \leq m \leq 1000$; $1 \leq s, t \leq n$; $s \neq t$). There are n junctions, some pairs of which are connected by m roads. The court and the prison are located at the s-th and the t-th junctions, respectively. The next m lines each contain five integers x, y, b, e and c ($1 \leq x, y \leq n$; $0 \leq b < e \leq 10^4$; $1 \leq c \leq 10^4$), indicating that the directed lanes running from the x-th junction to the y-th junction of a road that takes the motorcade of escort c units of time to pass through will be shut down between time b and e. The earliest time when the motorcade can leave the court is time 0.

Output

If the escort is possible, print the shortest escort time; otherwise, print "`Impossible`".

Sample

Input	Output
4 5 1 4 1 2 0 1 1 1 2 0 1 2 1 3 1 3 2 2 4 3 4 1 3 4 3 4 1	3

Hint

The sample is depicted in Figure 1. Each junction is represented by a numbered spot. Each road is represented by an arrow labeled with $\langle\langle b, e\rangle, c\rangle$, its shut-down time window and pass-through time.

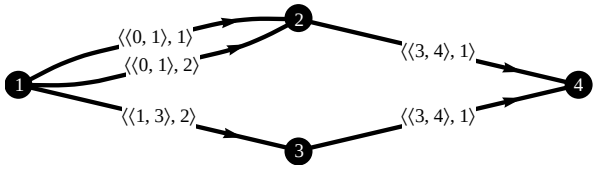


Figure 1: Depiction of the sample

The second road listed in the sample is virtually unusable for it takes the motorcade of escort longer time to pass through than it can stay available. The remaining four roads enable the following two escort routes:

1. $1 \overset{\langle 0,1\rangle}{\rightarrow} 2 \overset{\langle 3,4\rangle}{\rightarrow} 4$,
2. $1 \overset{\langle 1,3\rangle}{\rightarrow} 3 \overset{\langle 3,4\rangle}{\rightarrow} 4$.

(The label $\langle s, t\rangle$ over an arrow indicates that the motorcade passes through the corresponding road between time s and t.) The first route takes four units of escort time, whereas the second takes three, which is the shortest possible.

Problem B. Full Steiner Topologies

Description

A *full Steiner topology* for a given point set $P = \{p_1, p_2, \ldots, p_n\}$ is an undirected tree $T = (V, E)$ where $V = \{v_1, v_2, \ldots, v_n, v_{n+1}, \ldots, v_{2n-2}\}$ is the set of vertices, and E is the set of edges. The n distinctly labeled leaves of T, $v_1, v_2, \ldots, v_n$, correspond to $p_1, p_2, \ldots, p_n$, respectively; the remaining $n - 2$ vertices, $v_{n+1}, v_{n+2}, \ldots, v_{2n-2}$, called the *Steiner vertices*, are mutually indistinguishable and each have a degree of three. Figure 2 shows the only full Steiner topology for $P = \{p_1, p_2, p_3\}$. Figure 3 shows all three different full Steiner topologies for $P = \{p_1, p_2, p_3, p_4\}$.

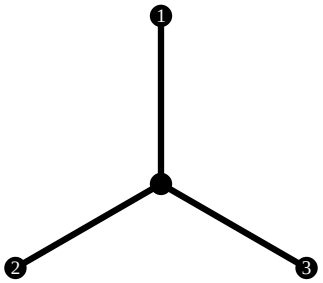


Figure 2: Full Steiner topology for $P = \{p_1, p_2, p_3\}$

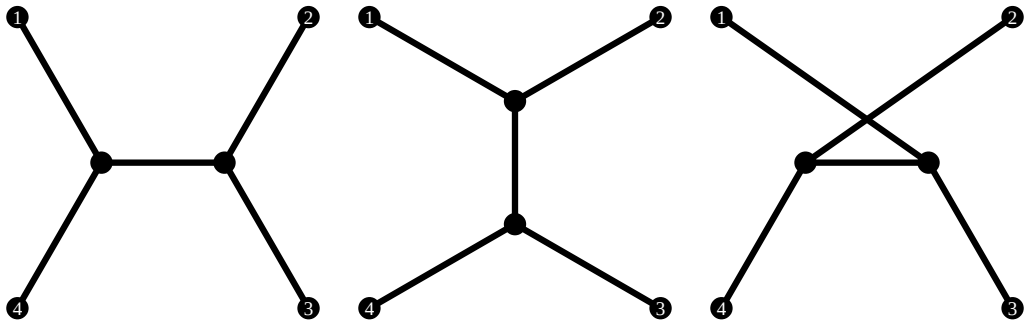


Figure 3: Full Steiner topologies for $P = \{p_1, p_2, p_3, p_4\}$

Given n, the cardinality of P, compute the number of distinct full Steiner topologies.

Input

The input contains multiple test cases. Each test case consists of a single integer n ($3 \leq n \leq 10^7$) on a separate line. The input ends where EOF is met.

Output

For each test case, print the answer on a separate line. You shall print the answer rounded to four significant digits. Let $m \cdot 10^e$ be the scientific form of the rounded answer, you shall print "$m\mathrm{E}e$", giving all four significant digits of m and stripping any leading zeroes before e.

Sample

Input	Output
3	1.000E0
30	8.687E36
300	5.677E697
3000	1.462E10024
30000	1.983E130306
300000	4.215E1603145
3000000	7.937E19031556

Problem C. Illuminated Planet

Description

A space probe is on a mission to observe a planetary system. It is instructed to shoot images of a planet, which orbits a star, using its optical camera. To ensure image quality, it must take the images only when a positive portion of the planet's surface illuminated by the star is visible.

Given the positions and radii of the planet and the star, and the position of the space probe, determine whether images can be taken in this setting. You do not have to consider the situation that the planet is occulted by the star.

Input

The input contains a single test case. The first line contains four numbers $(x_\oplus, y_\oplus, z_\oplus)$ and $r_\oplus$, the Cartesian coordinates of the planet and its radius. The second line contains four numbers $(x_\odot, y_\odot, z_\odot)$ and $r_\odot$, the Cartesian coordinates of the star and its radius. The third line contains three numbers (x, y, z), the Cartesian coordinates of the space probe. The input is physically consistent.

Output

If images can be taken, print "`Yes`"; otherwise, print "`No`".

Sample

Input
`0 0 0 6.371e3` `1.027e8 4.400e8 7.548e8 6.96e5` `3.489e9 -1.036e10 -5.249e9`
Output
`Yes`

Hint

The sample uses the real physical setting of Earth, the Sun and the Voyager 2 space probe. Voyager 2 is located far south to the ecliptic and thus always has the illuminated side of Earth within its sight.

Problem D. Polygon Division

Description

Given a regular polygon, there are numerous ways to divide it into several triangles and/or quadrangles by adding some diagonals that do not properly intersect each other. For example, Figure 4 shows all ten different divisions of a regular pentagon into triangles and quadrangles.

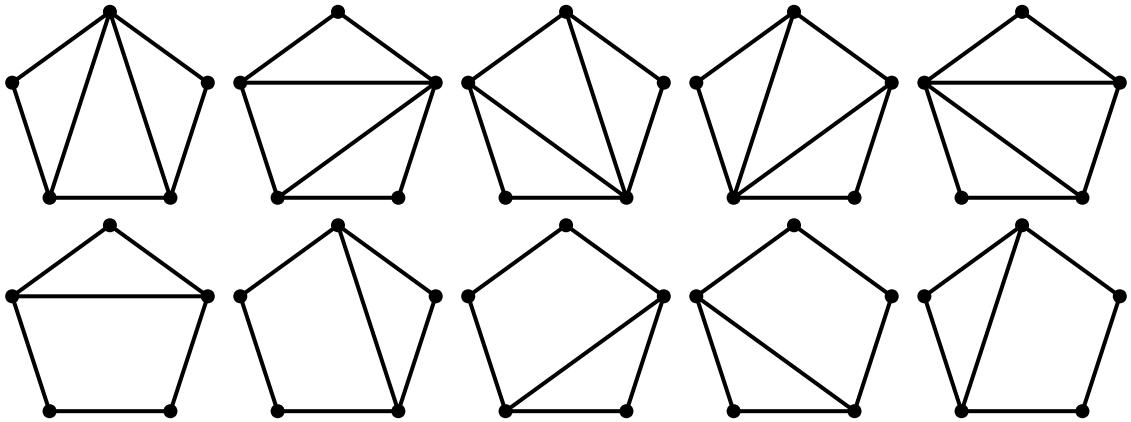


Figure 4: Divisions of a regular pentagon into triangles and quadrangles

Given n, the number of sides of the polygon, compute the number of such divisions.

Input

The input contains multiple test cases. Each test case consists of a single integer n ($3 \leq n \leq 5000$) on a separate line. The input ends where EOF is met.

Output

For each test case, print the answer modulo 2^{64} on a separate line.

Sample

Input	Output
3 4 5 6 7 8 9 10	1 3 10 38 154 654 2871 12925

Problem E. PopKart

Description

Xiao Feng is a big fan of the online multiplayer racing game PopKart. Among various types of play, Xiao Feng prefers the Speed Game, in which players compete to be the fastest racer. In order to win the races, apart from solid driving skills, the player must also own a top-class kart. Results of races among experienced players are mostly decided by the karts which they ride.

Now a new generation of karts has hit the market. Xiao Feng wants to purchase one of them. Two factors prevents him from going straightly for the best ones. First, the available (virtual) funds are limited. Second and more importantly, it is a non-trivial task to decide which karts are the best.

To find out the best karts that he can afford, Xiao Feng employs a simple scheme to classify the karts by their performance.

The performance of a kart is characterized by two parameters—the straight-line speed v and the drift speed w. A kart is said to be superior to another if and only if it is superior in either speed and not inferior in the other. All karts to which no other karts are superior are classified as karts of class 1. For each $m \geq 2$, all karts to which only karts of classes 1 through $m - 1$ are superior are classified as karts of class m.

Given the straight-line and drift speeds of n karts, classify them as specified above.

Input

The input contains a single test case. The first line contains n ($1 \leq n \leq 10^5$). The remaining n lines each contain two integers v and w ($0 \leq v, w < 10^6$), the straight-line and drift speeds of a kart.

Output

For each class in increasing order of m, print a list of karts of that class using the format

$$\text{“}k\texttt{:}\,\texttt{␣}(v_1\texttt{,}\,w_1)\,\texttt{␣}(v_2\texttt{,}\,w_2)\,\texttt{␣}\cdots\texttt{␣}(v_k\texttt{,}\,w_k)\text{”}$$

where k is the number of karts of that class, and $(v_1, w_1), (v_2, w_2), \ldots, (v_k, w_k)$ are their straight-line and drift speeds. The k pairs of speeds shall be sorted so that $v_i < v_j \vee (v_i = v_j \wedge w_i \leq w_j)$ for any $1 \leq i < j \leq k$ (“$\vee$” and “$\wedge$” represent logical OR and AND, respectively).

Sample

Input	Output
4 1 2 2 1 2 3 3 2	2: (2,3) (3,2) 2: (1,2) (2,1)

Problem F. Pumping Lemma

Description

In the theory of formal languages, the *pumping lemma for regular languages* bears importance for its characterization of an essential property of all regular languages.

In case of your unfamiliarity, we shall first review several related concepts. A *regular language* is the set of all strings accepted by a *deterministic finite automaton* (*DFA*). A DFA is an abstract model of computation informally explained below.

A DFA can be depicted by a state chart. Each node in the state chart represents a *state*. Of all states, exactly one is designated as the *start state*; and a subset of them is designated as a set of *final states*. Each labeled directed arc in the state chart represents a *state transition*. When the DFA works, it starts in the start state, reads symbols from a string one by one, and performs state transition accordingly. A state transition " $x \xrightarrow{c} y$ " means that if the DFA reads a symbol c when it is in state x, it will transit to state y. The DFA is said to accept a string if it is in a final state when it finishes reading symbols from the string.

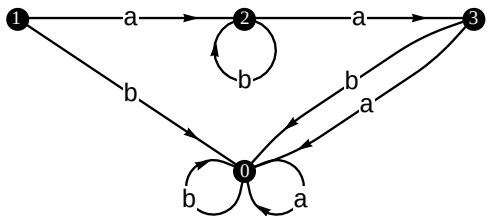


Figure 5: Transition function δ of DFA M

Consider the state chart in Figure 5, which describes a DFA M. The states of M are $\{0, 1, 2, 3\}$. We designate 1 as the start state and 3 as the only final state. When fed with the string `aba`, M starts in state 1 and performs the state transitions " $1 \xrightarrow{\texttt{a}} 2 \xrightarrow{\texttt{b}} 2 \xrightarrow{\texttt{a}} 3$ " to reach the final state 3. Consequently, M accepts the string `aba`. Precisely speaking, M accepts the regular language $L = \{\texttt{ab}^i\texttt{a} : i \geq 0\} = \{\texttt{aa}, \texttt{aba}, \texttt{abba}, \ldots\}$.

The pumping lemma for regular languages, put informally, states that for any sufficiently long string w of a regular language L can be written, subject to some length constraints, as the concatenation of three substrings x, y and z such that the strings resulting from removing or repeating y in w, i.e. xz and $xyyz$, $xyyyz$, ..., are also strings of L.

Given a DFA accepting a regular language L, find a string $w \in L$ satisfying that w can be written as $w = xyz$ so that $\{xz, xyz, xyyz, \ldots\} \subseteq L$, and $0 < |y| \leq |w| < 3n$ ($|\cdot|$ means the length of a string).

Input

The input contains a single test case describing a DFA whose states are $\{0, 1, 2, \ldots, n\}$ and which reads only lowercase letters. The first line contains n and the start state s ($1 \leq s \leq n \leq 1000$). The second line contains an integer m ($0 \leq m \leq 26n$). Each of the following m lines contains an integer x, a symbol c and another integer y ($1 \leq x \leq n$; $c \in \Sigma$; $0 \leq y \leq n$), specifying a state transition " $x \xrightarrow{c} y$ ". The state transition " $x \xrightarrow{c} 0$ " is implicitly assumed for any state x and symbol c if not otherwise specified. The next line contains an

integer k ($0 \leq k \leq n$), which is followed by a line containing k integers, the final states of M. 0 is not a final state.

If multiple choices of w exist, you may choose any one.

Output

If some $w = xyz \in L$ satisfies the given requirements, print "$x\ (y)\ z$"; otherwise, print "$\star$".

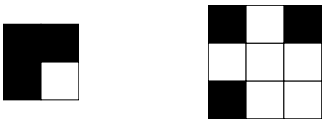
Sample

Input	Output
3 1 3 1 a 2 2 a 3 2 b 2 1 3	a(b)a

Problem G. Subimage Recognition

Description

An image A is said to be a subimage of another image B if it is possible to remove some rows and/or columns of pixels from B so that the resulting image is identical to A. Figure 6 illustrates an example. Image A, shown in Figure 6(a), is a subimage of image B, shown in Figure 6(b), because the image resulting from the removal of the middle row and column of pixels from B is identical to A.



(a) Image A (b) Image B

Figure 6: An example of a subimage

Given two black-and-white images A and B, determine whether A is a subimage of B.

Input

The input contains a single test case. The first line contains two integers r and c ($1 \leq r, c \leq 20$), the dimensions of A. The following r lines, each containing a string of length c, give an $r \times c$ 0-1 matrix representing the pixels of A. The next line contains two integers R and C ($r \leq R \leq 20$; $c \leq C \leq 20$), the dimensions of B. The following R lines, each containing a string of length C, give an $R \times C$ 0-1 matrix representing the pixels of B. A 0 indicates a white pixel; a 1 indicates a black pixel.

Output

If A is a subimage of B, print "`Yes`"; otherwise, print "`No`".

Sample

Input	Output
2 2 11 10 3 3 101 000 100	Yes

Problem H. Tower of Hanoi

Description

The Tower of Hanoi is a puzzle consisting of three pegs and a number of disks of different sizes which can slide onto any peg. The puzzle starts with the disks neatly stacked in order of size on one peg, the smallest at the top, thus making a conical shape.

The objective of the puzzle is to move the entire stack to another peg, obeying the following rules:

- Only one disk may be moved at a time.
- Each move consists of taking the upper disk from one of the pegs and sliding it onto another peg, on top of the other disks that may already be present on that peg.
- No disk may be placed on top of a smaller disk.

For n disks, it is a well-known result that the optimal solution takes $2^n - 1$ moves.

To complicate the puzzle a little, we allow multiple disks to be of the same size. Moreover, equisized disks are mutually distinguishable. Their ordering at the beginning should be preserved at the end, though it may be disturbed during the process of solving the puzzle.

Given the number of disks of each size, compute the number of moves that the optimal solution takes.

Input

The input contains multiple test cases. Each test case consists of two lines. The first line contains two integers n and m ($1 \le n \le 100$; $1 \le m \le 10^6$). The second lines contains n integers $a_1, a_2, \ldots, a_n$ ($1 \le a_1, a_2, \ldots, a_n \le 10^5$). For each $1 \le i \le n$, there are a_i disks of size i. The input ends where EOF is met.

Output

For each test case, print the answer modulo m on a separate line.

Sample

Input	Output
1 1000 2 5 1000 1 1 1 1 1 5 1000 2 2 2 2 2 5 1000 1 2 1 2 1	3 31 123 41

Problem I. Typographical Ligatures

Description

Typesetting involves the presentation of textual material in graphic form on paper or some other medium. Close as it is related to our daily life, typesetting exhibits certain complexities which may be unfamiliar and sometimes unimaginable to the layman. The use of typographical ligatures is one such complexity.

Compare the two appearances of the word "define" in Figure 7. Figure 7(a) illustrates the word typeset with the "fi" ligature as Knuth's TEX does. The letters "f" and "i" are combined into a single glyph.[1] Figure 7(b) shows the word typeset without the ligature as Microsoft Word does. The letters "f" and "i" remain separate. Other examples include the "ff", "ffi", "fl" and "ffl" ligatures, as shown in Figure 8. Perhaps the most notable ligature in active use is the "&" (ampersand), which originated from "et", the Latin word for "and".

define (a) define (b)

Figure 7: The word "define" typeset with and without the "fi" ligature

effect office reflect waffle

Figure 8: Examples of the "ff", "ffi", "fl" and "ffl" ligatures

Ligatures are primarily intended to improve spacing between letters. Despite that their impact on legibility is debated and that their use is declining, some people insist that they are an essential part of quality typesetting. In order to typeset ligatures, separate glyphs have to be used since ligatures generally differ from direct combinations of their constituent letters in most cases.

Given some text, count the number of glyphs that have to be used to typeset it. Only the "ff", "fi", "ffi", "fl", and "ffl" ligatures are considered. Note that they are case-sensitive. Ligatures are recognized following the leftmost longest rule—find the leftmost match, then the longest match if there are any ties. Each distinct letter, ligature or punctuation mark requires a separate glyph. However, if a letter appears only in ligatures and nowhere else, it shall not be assigned a glyph. Furthermore, left single and double quotes differ from their right counterparts. Spaces do not require a glyph.

Input

The input contains a single test case consisting of some text. The text is given on multiple lines not longer than 100 characters each. The text contains only letters (lowercase and uppercase), punctuation marks and spaces. Punctuation marks include periods, commas, semicolons,

[1] This problem set is typeset using LATEX, a variant of TEX. You may discover quite a few occurrences of ligatures through scrutiny of the text.

colons, single and double quotes, exclamation and question marks. Quotes are represented by "`" (left single quote), "´" (right single quote), "``" (left double quote) and "´´" (right double quote), respectively. They are also recognized following the leftmost longest rule. The input ends where EOF is met.

Output

Print the number of glyphs that have to be used to typeset the given text.

Sample

Input	Output
`` ```define, effect; office. reflect? waffle!´´´ ````	23

Hint

The text in the sample is typeset as Figure 9. The 23 used glyphs are listed in Figure 10 in the order of their first appearances in Figure 9.

“‘define, effect; office.
reflect? waffle!”’

Figure 9: Result of typesetting the text in the sample

“	‘	d	e	fi	n	,	ff
c	t	;	o	ffi	.	r	fl
?	w	a	ffl	!	”	’	

Figure 10: Glyphs appearing in Figure 9

Problem J. Zen Puzzle Garden

Description

Zen Puzzle Garden is a single-player game in which the player has to control a little monk who must endeavor to rake all of the sand in a Japanese rock garden. Figure 11 shows three screenshots of the game.

(a) (b) (c)

Figure 11: Screenshots of Zen Puzzle Garden

The sand in the garden is divided into a rectangular grid of squares. Some of the squares are occupied by rocks. As shown in Figure 11(a), before he starts, the monk stands outside the sand. Then he repeatedly walks along a path through the sand consisting of adjacent squares and rakes every square in the path. Adding to the complexity, he cannot walk over the squares that have already been raked or that are occupied by rocks. Furthermore, he is not allowed to change direction unless he cannot move ahead any more. Figure 11(b) illustrates that the monk has raked some squares, and he now must walk right until he exits the sand. To complete the game, the monk must rake all sand-covered squares and not be "trapped" in the sand when he finishes, as shown in Figure 11(c).

Given a solvable puzzle, find a solution to it.

Input

The input contains a single test case describing a solvable puzzle. The first line contains two integers r and c ($2 \leq r, c \leq 12$). The next r lines, each containing c integers, give an $r \times c$ 0-1 matrix describing the sand in the garden. A 0 indicates a sand-covered square; a 1 indicates a rock-occupied square. All squares are not occupied by rocks.

Output

Print your solution as follows. The first line contains an integer n. Each of the next n lines is in the format

$$\text{“}k:\text{␣}(r_0,c_0)\text{␣}(r_1,c_1)\text{␣}\cdots\text{␣}(r_{k-1},c_{k-1})\text{␣}(r_k,c_k)\text{”}$$

where $(r_1, c_1), (r_2, c_2), \ldots, (r_{k-1}, c_{k-1})$ is a path through the sand, and (r_0, c_0) and (r_k, c_k) are two fictional squares outside the sand and immediately next to (r_1, c_1) and (r_{k-1}, c_{k-1}), respectively.

If multiple solutions exist, you may print any one.

Sample

Input

```
5 5
0 0 0 0 0
0 0 0 0 0
0 1 0 0 0
0 0 0 1 0
0 0 0 0 0
```

Output

```
6
7: (0,1) (1,1) (2,1) (3,1) (4,1) (5,1) (6,1)
7: (0,2) (1,2) (2,2) (2,3) (2,4) (2,5) (2,6)
4: (4,6) (4,5) (5,5) (6,5)
8: (6,2) (5,2) (4,2) (4,3) (3,3) (3,4) (3,5) (3,6)
5: (0,3) (1,3) (1,4) (1,5) (1,6)
4: (6,3) (5,3) (5,4) (6,4)
```