Problem Set for the 2nd Day of Winter Camp

Japanese Alumni Group

Contest Held on 24 Feb 2007

Status of Problems

Problem	Source
Problem A	The 2005 ACM Asia Programming Contest, Hangzhou
Problem B	The 2006 ACM Southeastern Europe Programming Contest
Problem C	The 2006 ACM Southeastern Europe Programming Contest
Problem D	*new problem*
Problem E	The 2006 ACM Southeastern Europe Programming Contest
Problem F	*new problem*
Problem G	The 2005 ACM Asia Programming Contest, Seoul
Problem H	The 2005 ACM Asia Programming Contest, Seoul
Problem I	The 2005 ACM Asia Programming Contest, Seoul

The problems listed with "*new problem*" in the Source column were newly created by the members of Japanese Alumni Group.

The other problems were taken from past programming contests as indicated above. Most of them were modified by the members of Japanese Alumni Group for clarification of the statement, adjustment of difficulty level, change of data input style, attempt to hide the source of each problem from the participants during the contest proper, and/or other purposes. The sources of all problems were revealed to the participants after the contest ended.

Terms of Use

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem A

Cells

Input File: cells.in

Scientists are conducting research on the behavior of a newly discovered Agamic Cellular Microbe. This special kind of microbe is capable of massively reproducing by itself in a short time. The lifetime of an ACM consists of three phases:

1. The infancy phase, which starts from its birth and lasts for approximately several seconds;
2. The multiplication phase, in which one ACM can procreate up to 100 offspring in only several milliseconds;
3. The mature phase, in which it remains inactive for the rest of its life.

At the beginning of the experiment, a newborn, single cell of ACM, is put into a suitable circumstance for its production. This cell, numbered as 0, starts to multiply and its descendants are numbered, starting from 1, according to their positions in the family hierarchy. During the experiment special equipment is used to record the numbers of the offspring generated by each of the ACM's. The experiment is stopped after a certain time period.

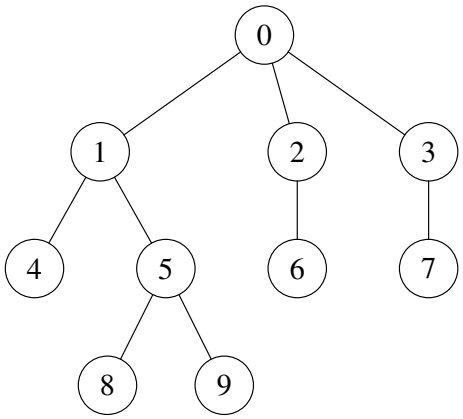


Figure 1: The Family Tree of ACM's in the First Case of Sample Input

Your task is to help the scientists to determine whether one ACM is an ancestor of another.

Input

The input will contain multiple test cases. The first line of the input is a single integer T ($1 \le T \le 20$) which is the number of test cases. T test cases follow, each preceded by a single blank line.

Each test case starts with a single integer N ($1 \le N \le 300,000$) which is the number of ACM's that have their descendants recorded. The following N integers (not necessarily on a same line), C_i ($0 \le i < N$, $0 \le C_i \le 100$), give the number of offspring of the i-th ACM. The next line contains an integer M ($1 \le M \le 500,000$) which is the number of queries. M lines follow, each contains two distinct integers a and b, querying whether the a-th ACM is an ancestor of the b-th ACM.

The total number of ACM's may be greater than N, but would never exceed 2,000,000.

Output

For each case, print the number of queries which should be responded as *yes*, i.e. the a-th ACM is an ancestor of the b-th ACM.

Practice Contest for World Finals 2007: Thinking Contest

Sample Input	Output for the Sample Input
2 6 3 2 1 1 0 2 5 0 1 2 4 3 5 1 8 6 9 5 2 0 3 0 1 4 2 6 1 6 2 3 3 5	2 2

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem B

Japan

Input File: japan.in

Japan plans to welcome the ACM ICPC World Finals and a lot of roads must be built for the venue. Japan is tall island with N cities on the East coast and M cities on the West coast ($M \leq 1000$, $N \leq 1000$). K superhighways will be build. Cities on each coast are numbered $1, 2, \ldots$ from North to South. Each superhighway is straight line and connects city on the East coast with city of the West coast. The funding for the construction is guaranteed by ACM. A major portion of the sum is determined by the number of crossings between superhighways. At most two superhighways cross at one location. Write a program that calculates the number of the crossings between superhighways.

Input

The input file starts with T — the number of test cases. Each test case starts with three integers — N, M, K. Each of the next K lines contains two integers — the numbers representing cities connected by the superhighway. The first one is the city number for the East coast and the second is for the West.

Output

For each test case write one line containing the case number and the number of crossings. Follow the output format shown below.

Sample Input	Output for the Sample Input
1 3 4 4 1 4 2 3 3 2 3 1	Test case 1: 5

Practice Contest for World Finals 2007: Thinking Contest

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem C

Maximum

Input File: maximum.in

Let $x_1, x_2, \ldots, x_m$ be real numbers satisfying the following conditions for some integers $a(> 0)$ and b:

$$\begin{cases} -\frac{1}{\sqrt{a}} \le x_i \le \sqrt{a} \\ x_1 + x_2 + \cdots + x_m = b \times \sqrt{a} \end{cases}$$

Determine the maximum value of $x_1^p + x_2^p + \cdots + x_m^p$ for some even positive integer p.

Input

Each input line contains four integers: m, p, a, b ($m \le 2000$, $p \le 12$, $a \le 8$, p is even). Input is correct, i.e. for each input numbers there exists $x_1, x_2, \ldots, x_m$ satisfying the given conditions.

Output

For each input line print one number — the maximum value of expression, given above. The answer must be printed with three digits after the decimal point, and must not contain an error greater than 0.001.

Sample Input	Output for the Sample Input
1997 12 3 -318 10 2 4 -1	189548.000 6.000

Practice Contest for World Finals 2007: Thinking Contest

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem D

The Phantom

Input File: mirror.in

Mr. Hoge is in trouble. He just bought a new mansion, but it's haunted by a phantom. He asked a famous conjurer Dr. Huga to get rid of the phantom. Dr. Huga went to see the mansion, and found that the phantom is scared by its own mirror images. Dr. Huga set two flat mirrors in order to get rid of the phantom.

As you may know, you can make many mirror images even with only two mirrors. You are to count up the number of the images as his assistant. Given the coordinates of the mirrors and the phantom, show the number of images of the phantom which can be seen through the mirrors.

Input

The input consists of multiple test cases. Each test cases starts with a line which contains two positive integers, p_x and p_y ($1 \leq p_x, p_y \leq 50$), which are the x- and y-coordinate of the phantom. In the next two lines, each line contains four positive integers u_x, u_y, v_x and v_y ($1 \leq u_x, u_y, v_x, v_y \leq 50$), which are the coordinates of the mirror.

Each mirror can be seen as a line segment, and its thickness is negligible. No mirror touches or crosses with another mirror. Also, you can assume that no images will appear within the distance of 10^{-5} from the endpoints of the mirrors.

The input ends with a line which contains two zeros.

Output

For each case, your program should output one line to the standard output, which contains the number of images recognized by the phantom. If the number is equal to or greater than 100, print "`TOO MANY`" instead.

Sample Input	Output for the Sample Input
4 4 3 3 5 3 3 5 5 6 4 4 3 3 5 3 3 5 5 5 0 0	4 TOO MANY

Practice Contest for World Finals 2007: Thinking Contest

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem E

Payment System

Input File: payment.in

The payment system in the University of Mineral Water Production is completely automated (written entirely in Tomato Programming Language) and lets you input the amount of money you want to withdraw. Due to the high payment rates of the professors they may input the amounts in exponential forms. So if you want to withdraw 16 MWU (mineral water units) you can enter `16`, `2^4` or `2^2^2`.

One day, Stanescu tried to withdraw some money from his account which had balance of 80 MWU. He mistakenly entered `2^3^2` and for his surprise he got 512 MWU, although he should not be able to take more than 80. The system was composed of two main modules — the first module checks whether the account has enough money to execute the transaction and the second module gives the money to the user. It turned out that the first module has a problem with the '^' operator. It evaluates it from left to right, while the second evaluates them from right to left (the correct way). Thus for the first module `2^3^2=(2^3)^2=64` while for the second it's `2^3^2=2^(3^2)=512`.

You have to write program which helps Stanescu get as much as he can from the university system. If you think it's not legal or something, be sure that the University of Mineral Water Production is bad and evil.

Input

In the input file the amounts of the accounts of Stanescu will be given. Each amount is given on a separate line and is an integer between 2 and $10^{100} - 1$.

Output

For each given amount, your program should print to the standard output what Stanescu should enter to get maximal number of money. The output should:

- consist only of integers and the '^' operator between them.
- pass the check of the first module and be as much as possible for the second.
- not contain the number 1 (it is useless anyway).

If more than one answer exists, output the one whose first number is minimal, if still more exist, choose the one whose second number is minimal and so on.

Sample Input	Output for the Sample Input
16	2^2^2
80	2^3^2
49	7^2
1025	2^2^5
12341234	2^2^2^5
12345678901234567890	2^2^2^2^3^2

Practice Contest for World Finals 2007: Thinking Contest

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem F

Rakunarok

Input File: rakunarok.in

You are deeply disappointed with the real world, so you have decided to live the rest of your life in the world of MMORPG (Massively Multi-Player Online Role Playing Game). You are no more concerned about the time you spend in the game: all you need is *efficiency*.

One day, you have to move from one town to another. In this game, some pairs of towns are connected by roads where players can travel. Various monsters will raid players on roads. However, since you are a high-level player, they are nothing but the source of experience points. Every road is bi-directional. A path is represented by a sequence of towns, where consecutive towns are connected by roads.

You are now planning to move to the destination town through the *most efficient* path. Here, the efficiency of a path is measured by the total experience points you will earn in the path divided by the time needed to travel the path.

Since your object is moving, not training, you choose only a *straightforward* path. A path is said straightforward if, for any two consecutive towns in the path, the latter is closer to the destination than the former. The *distance* of two towns is measured by the shortest time needed to move from one town to another.

Write a program to find a path that gives the highest efficiency.

Input

The first line contains a single integer c that indicates the number of cases.

The first line of each test case contains two integers n and m that represent the numbers of towns and roads respectively. The next line contains two integers s and t that denote the starting and destination towns respectively. Then m lines follow. The i-th line contains four integers u_i, v_i, e_i, and t_i, where u_i and v_i are two towns connected by the i-th road, e_i is the experience points to be earned, and t_i is the time needed to pass the road.

Each town is indicated by the town index number from 0 to $(n - 1)$. The starting and destination towns never coincide. n , m , e_i's and t_i's are positive and not greater than 1000.

Output

For each case output in a single line, the highest possible efficiency. Print the answer with four decimal digits. The answer may not contain an error greater than 10^{-4}.

Sample Input	Output for the Sample Input
2 3 3 0 2 0 2 240 80 0 1 130 60 1 2 260 60 3 3 0 2 0 2 180 60 0 1 130 60 1 2 260 60	3.2500 3.0000

Practice Contest for World Finals 2007: Thinking Contest

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

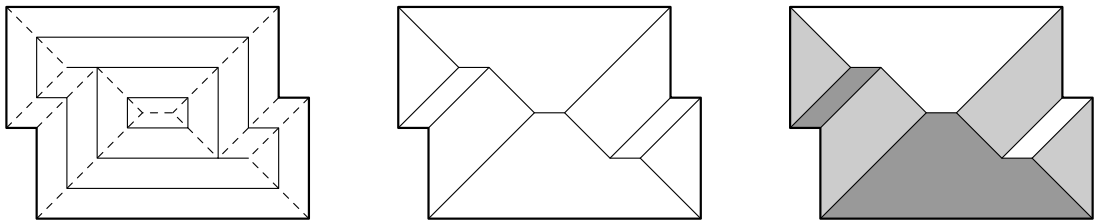
Thinking Contest

Problem G

Roof

Input File: roof.in

You are going to build a new house. To know the minimum construction cost of the roof, you want to know its height. But you have the information only on the boundary of the house, so you have to compute the height of the roof from the boundary information. The formal description is as follows. The boundary of a house is defined as a rectilinear polygon with only axis-parallel edges, horizontal or vertical. Let P be the rectilinear polygon with n vertices. A *straight skeleton* SK(P) is a trace of the vertices of P when P is shrunk, each edge moving at the same speed and keeping the same direction. (a) and (b) in Figure 2 show the shrinking process for P and SK(P), respectively.



(a) (b) (c)

Figure 2: P, SK(P), and the Corresponding Roof

Now we assume that P is on the xy-plane, and P is shrunk at unit speed while moving upward ($+z$-direction) at unit speed. Then P traces a three dimensional polyhedral surface, called terrain, and SK(P) can be seen as the projection of the edges of this terrain onto the xy-plane. From the definition, each face of 45 degree with xy-plane. Another fact for this terrain is that each of its faces is bounded by at least one edge of P. We call this terrain the roof of the polygon P. For instance, (c) in Figure 1 shows the roof. The height of a point q on the roof is the altitude, i.e. distance between q and its projected point on the xy-plane. The height of a roof is the maximum height of points on the roof. Your task is to compute the height of the roof of the input rectilinear polygon P.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input. Each test case starts with a line containing an integer n, the number of vertices of the input polygon P ($4 \leq n \leq 1000$). In the next line, (x, y)-coordinates of the n vertices of P are given according to the counterclockwise order. The first pair of integers is x and y coordinates of the first vertex of P, the second pair is x and y coordinates of the second vertex of P, and so on. The coordinates are separated by a single space, and are positive integer values between 1 and 100,000, both inclusive.

It is guaranteed that P is not degenerate, and that no edges in the same direction (horizontal or vertical) share a single vertex.

Output

Print exactly one line for each test case. For each test case, print the height of the roof of P with exactly one digit in the fraction part.

Practice Contest for World Finals 2007: Thinking Contest

Sample Input	Output for the Sample Input
3 4 1 1 4 1 4 4 1 4 6 4 1 8 1 8 6 1 6 1 4 4 4 14 1 1 4 1 4 2 7 2 7 1 11 1 11 10 8 10 8 8 1 8 1 6 3 6 3 4 1 4	1.5 2.0 3.0

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem H

Seminar Room

Input File: seminar.in

My department, Department of Computer Science, has a seminar room equipped with luxurious conveniences. Many research groups want to utilize the seminar room among others. To make out a weekly schedule of the seminar room, each research group was requested to submit a candidate time interval for its own seminar. Having realized that some candidate time intervals were overlapped, the department chairman Prof. Yang fell in a difficult situation. In order to assign the seminar room to the groups as fair as possible, he decided to request the seminar groups to submit two candidate time intervals. Now, Prof. Yang's problem of preparing a weekly schedule is to assign one of the candidate time intervals submitted by each group to the group so that the time intervals assigned to groups do not overlap each other. If the finish time of one time interval coincides with the start time of the other, the two time intervals are not considered to be overlapped.

A candidate time interval is specified by a pair of the start time and finish time. The time is written in format *ddd*`:`*hh*`:`*mm*, where *ddd* are three capital letters representing the day of a week, *hh* are two digits representing full hours ($00 \leq hh \leq 23$), and `mm` are two digits representing minutes ($00 \leq mm \leq 59$). On Sundays, a reservation will not be made. Thus, *ddd* are one of `MON`, `TUE`, `WED`, `THU`, `FRI`, `SAT`.

For example, suppose you are given three seminar groups and its candidate time intervals as shown in the table below.

seminar group 1	`MON:09:00 MON:11:00` `MON:10:00 MON:12:00`
seminar group 2	`MON:09:00 MON:11:30` `TUE:13:25 TUE:14:27`
seminar group 3	`MON:09:30 MON:11:00` `MON:23:00 TUE:01:00`

Table 1: Candidate Time Intervals for Three Seminar Groups

If three time intervals [`MON 09:00`–`MON 11:00`], [`TUE 13:25`–`TUE 14:27`], and [`MON 23:00`–`TUE 01:00`] are assigned to seminar groups 1, 2, and 3, respectively, the time intervals do not overlap each other, and thus all the seminar groups will be happy.

Now, Prof. Yang wants to find an efficient method to determine, for given two candidate time intervals per each seminar group, whether or not one of the candidate time intervals can be assigned to each group so that all the time intervals assigned to groups do not overlap each other. Write a program that can help Prof. Yang.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input. Each test case starts with a line containing a positive integer n, $n \leq 1,000$, representing the number of seminar groups. In the next n lines, each line contains two candidate time intervals for a seminar group in the form $d_1d_1d_1:h_1h_1:m_1m_1$, $d_2d_2d_2:h_2h_2:m_2m_2$, $d_3d_3d_3:h_3h_3:m_3m_3$, and $d_4d_4d_4:h_4h_4:m_4m_4$. The two candidate time intervals are [$d_1d_1d_1:h_1h_1:m_1m_1$–$d_2d_2d_2:h_2h_2:m_2m_2$] and [$d_3d_3d_3:h_3h_3:m_3m_3$–$d_4d_4d_4:h_4h_4:m_4m_4$]. There is a single space between the start time and finish time of a time interval and between the two time intervals.

Output

Print exactly one line for each test case. Print `YES` if one of the candidate time intervals can be assigned to each group so that all the time intervals assigned to groups do not overlap each other. Otherwise, print `NO`.

Practice Contest for World Finals 2007: Thinking Contest

Sample Input	Output for the Sample Input
3 3 MON:09:00 MON:11:00 MON:10:00 MON:12:00 MON:09:00 MON:11:30 TUE:13:25 TUE:14:27 MON:09:30 MON:11:00 MON:23:00 TUE:01:00 2 SAT:08:00 SAT:09:00 FRI:13:13 FRI:14:14 SAT:08:00 SAT:09:00 FRI:13:13 FRI:14:14 4 FRI:13:13 FRI:14:14 SAT:08:00 SAT:09:00 FRI:13:13 FRI:14:14 SAT:08:00 SAT:09:00 FRI:13:13 FRI:14:14 SAT:08:00 SAT:09:00 FRI:13:13 FRI:14:14 SAT:08:00 SAT:09:00	YES YES NO

Practice Contest for ACM-ICPC World Finals 2007

presented by JAG

Thinking Contest

Problem I

String Compression

Input File: string.in

Run Length Encoding(RLE) is a simple form of compression. RLE consists of the process for searching for a repeated runs of a single character in a string to be compressed, and replacing them by a single instance of the character and a run count. For example, a string `abcccddddddefgggggggggghijk` is encoded into a string `ab3c6def10ghijk` by RLE.

A new compression method similar to RLE is devised and the rule of the method is as follows: if a substring S is repeated k times, replace k copies of S by $k(S)$. For example, `letsgogogo` is compressed into `lets3(go)`. The length of `letsgogogo` is 10, and the length of `lets3(go)` is 9. In general, the length of $k(S)$ is ⟨number of digits in k⟩ + ⟨length of S⟩ + 2 (for '`(`' and '`)`'). For example, the length of `123(abc)` is 8. It is also possible to nest compression, so the substring S may itself be a compressed string. For example, `nowletsgogogoletsgogogo` could be compressed as a `now2(lets3(go))`, and `nowletsgogogoletsgogogoandrunrunrun` could be compressed as `now2(lets3(go))and3(run)`.

Write a program that, for a given string, gives the length of the shortest compressed string using the compression rules as described above.

Input

The input consists of T test cases. The number of test cases T is given in the first line of the input. Each test case consists of a single line containing one string of no more than 200 characters drawn from a lower case alphabet. The length of shortest input string is 1.

Output

Your program is to write to standard output. Print exactly one line for each test case. For each test case, print the length of the shortest compressed string.

Sample Input	Output for the Sample Input
4 ababcd letsgogogo nowletsgogogoletsgogogo nowletsgogogoletsgogogoandrunrunrun	6 9 15 24