Problem A. Exchange Rates

Input file: `exchange.in`
Output file: `exchange.out`

Now that the Loonie is hovering about par with the Greenback, you have decided to use your $1000 entrance scholarship to engage in currency speculation. So you gaze into a crystal ball which predicts the closing exchange rate between Canadian and U.S. dollars for each of the next several days. On any given day, you can switch all of your money from Canadian to U.S. dollars, or vice versa, at the prevailing exchange rate, minus 3% commission, minus any fraction of a cent.

Assuming your crystal ball is correct, what's the maximum amount of money you can have, in Canadian dollars, when you're done?

Input

The input contains a number of test cases, followed by a line containing 0. Each test case begins with $0 < d \leq 365$, the number of days that your crystal ball can predict. d lines follow, giving the price of a U.S. dollar in Canadian dollars, as a real number.

Output

For each test case, output a line giving the maximum amount of money, in Canadian dollars and cents, that it is possible to have at the end of the last prediction, assuming you may exchange money on any subset of the predicted days, in order.

Example

exchange.in	exchange.out
3 1.0500 0.9300 0.9900 2 1.0500 1.1000 0	1001.60 1000.00

Problem B. Conformity

Input file: `conformity.in`
Output file: `conformity.out`

Frosh commencing their studies at Waterloo have diverse interests, as evidenced by their desire to take various combinations of courses from among those available.

University administrators are uncomfortable with this situation, and therefore wish to offer a conformity prize to frosh who choose one of the most popular combinations of courses. How many frosh will win the prize?

Input

The input consists of several test cases followed by a line containing 0. Each test case begins with an integer $1 \leq n \leq 10000$, the number of frosh. For each frosh, a line follows containing the course numbers of five distinct courses selected by the frosh. Each course number is an integer between 100 and 499.

The popularity of a combination is the number of frosh selecting exactly the same combination of courses. A combination of courses is considered most popular if no other combination has higher popularity.

Output

For each line of input, you should output a single line giving the total number of students taking some combination of courses that is most popular.

Example

conformity.in	conformity.out
3 100 101 102 103 488 100 200 300 101 102 103 102 101 488 100 3 200 202 204 206 208 123 234 345 456 321 100 200 300 400 444 0	2 3

Problem C. Pseudoprime numbers

Input file: `pseudoprimenumbers.in`
Output file: `pseudoprimenumbers.out`

Fermat's theorem states that for any prime number p and for any integer $a > 1$, $a^p == a \mod p$. Some (but not very many) non-prime values of p, known as base-a pseudoprimes, have this property for some a. (And some, known as Carmichael Numbers, are base-a pseudoprimes for all a.)

Given $2 < p \leq 10^9$ and $1 < a < p$, determine whether or not p is a base-a pseudoprime.

Input

Input contains several test cases followed by a line containing "`0 0`". Each test case consists of a line containing p and a.

Output

For each test case, output "`yes`" if p is a base-a pseudoprime; otherwise output "`no`".

Example

pseudoprimenumbers.in	pseudoprimenumbers.out
3 2 10 3 341 2 341 3 1105 2 1105 3 0 0	no no yes no yes yes

Problem D. Carpool

Input file: `carsdontswim.in`
Output file: `carsdontswim.out`

A group of friends has just completed their CS assignments, and because of the nice weather, they decide to go to Joe's house for a BBQ. Unfortunately, after all that coding, they are too tired to walk. Fortunately, between them they have enough cars to take everyone.

Joe remembers that he needs to stop off at the supermarket along the way to buy some burgers and pop.

Jenn proposes that they stop at her house to get a frisbee.

Jim decides that he doesn't like burgers, and wants to grab a pizza along the way.

After having spent so long in the computer lab, Jerry's eyes have become very sensitive to sunlight, so he needs to stop at his house for his sunglasses.

And so it goes: each person needs to run a little errand along the way. At this rate, the friends worry that it will be dark by the time they get to Joe's house. They launch into a heated discussion to about who should go in which car to minimize the time needed for everyone to reach Joe's house. The discussion itself, of course, wastes precious time that could be better spent at the BBQ.

To help the group, you will write a program to settle the discussion by computing an optimal assignment of people to cars. The overall travel time is the maximum of the travel times of each car. An optimal assignment is one that minimizes the overall travel time.

Your program will be provided with a representation of the roads in the city, in the form of distances between major landmarks. Assume that every car always travels at 60 kilometres per hour (one kilometre per minute). Each stop (at a supermarket, someone's house, etc.) takes five minutes.

Although the friends have many cars between them, to be nice to the environment, they decide to take no more cars than necessary. Each car can take at most five people.

Input

The first line of input contains two integers n and m, $1 \le n \le 15$, $1 \le m \le 1000$, the number of people in the group and the number of roads in the city. The places that must be visited along the way are numbered 1 through n. The campus is numbered 0, and Joe's house is numbered $n + 1$. An additional m lines follow, each containing three integers describing a stretch of road. The first two integers are in the range 0 to $n + 1$ inclusive, and describe the two places connected by the stretch of road. The third integer specifies the length of the stretch of road, in kilometres. A road may be taken in both directions. There is a sequence of roads connecting every place in the city to every other place.

Output

Output a single integer, giving the number of minutes required for everyone to reach Joe's house using an optimal assignment of people to cars.

Example

carsdontswim.in	carsdontswim.out
1 2 0 1 15 1 2 10	30

Problem E. Friend or Foe

Input file: `friendswillbefriends.in`
Output file: `friendswillbefriends.out`

Luke has a bit of trouble telling the difference between star systems in the Rebel Alliance and those in the Empire. He has a list of the x, y, z coordinates of each system in the Empire and each in the Alliance, but at warp speed he simply has insufficient time to look up systems in his lists.

After destroying the friendly planet Endor, Luke has had to admit he needs the help of his targeting computer. His computer, being an early model, can only compute the truth value of the inequality

$$ax + by + cz + d > 0$$

Where x, y, z are the coordinates of the system, and a, b, c, d are real-valued coefficients. You are to compute a, b, c, d so that the inequality holds for all systems in the Empire, and for no systems in the Alliance.

Input

Input consists of several test cases followed by $-1 - 1$. Each test case first gives the number of Alliance systems, followed by a line for each system giving the integer coordinates $-100 \le x, y, z \le 100$ of the system in 3-dimensional space. The number of Empire systems, and the coordinates of each, follow. The Empire and Alliance combined have at least one and no more than 200 systems. All systems have distinct coordinates.

Output

For each test case, output a single line containing a, b, c, d as real numbers separated by spaces. Use enough precision to ensure that your solution is correct. You may assume that a solution exists; if there is more than one solution, any one will do.

Example

friendswillbefriends.in
2 -93 48 -92 -62 12 -32 8 51 98 -61 -3 72 81 95 25 22 89 43 -99 100 -2 -96 -18 45 -63 36 -21 -8 71 -24 42 -1 -1
friendswillbefriends.out
83.000000 65.000000 -27.000000 -62.000000

Problem F. Open Source

Input file: `opensource.in`
Output file: `opensource.out`

At an open-source fair held at a major university, leaders of open-source projects put sign-up sheets on the wall, with the project name at the top in capital letters for identification.

Students then signed up for projects using their userids. A userid is a string of lower-case letters and numbers starting with a letter.

The organizer then took all the sheets off the wall and typed in the information.

Your job is to summarize the number of students who have signed up for each project. Some students were overly enthusiastic and put their name down several times for the same project. That's okay, but they should only count once. Students were asked to commit to a single project, so any student who has signed up for more than one project should not count for any project.

There are at most 10 000 students at the university, and at most 100 projects were advertised.

Input

The input contains several test cases, each one ending with a line that starts with the digit 1. The last test case is followed by a line starting with the digit 0.

Each test case consists of one or more project sheets. A project sheet consists of a line containing the project name in capital letters, followed by the userids of students, one per line.

Output

For each test case, output a summary of each project sheet. The summary is one line with the name of the project followed by the number of students who signed up. These lines should be printed in decreasing order of number of signups. If two or more projects have the same number of signups, they should be listed in alphabetical order.

Example

opensource.in	opensource.out
UBQTS TXT tthumb LIVESPACE BLOGJAM philton aeinstein YOUBOOK j97lee sswxyzy j97lee aeinstein SKINUX 1 0	YOUBOOK 2 LIVESPACE BLOGJAM 1 UBQTS TXT 1 SKINUX 0

Problem G. Antimonotonicity

Input file: antimonotonicity.in
Output file: antimonotonicity.out

I have a sequence Fred of length n comprised of integers between 1 and n inclusive. The elements of Fred are pairwise distinct. I want to find a subsequence Mary of Fred that is as long as possible and has the property that:

$$\text{Mary}[0] > \text{Mary}[1] < \text{Mary}[2] > \text{Mary}[3] < \dots$$

Input

The first line of input will contain a single integer T expressed in decimal with no leading zeroes. T will be at most 50. T test cases will follow.

Each test case is contained on a single line. A line describing a test case is formatted as follows:

$$n \text{ Fred}[0] \text{ Fred}[1] \text{ Fred}[2] \dots \text{ Fred}[n-1].$$

where n and each element of Fred is an integer expressed in decimal with no leading zeroes. No line will have leading or trailing whitespace, and two adjacent integers on the same line will be separated by a single space. n will be at most 30 000.

Output

For each test case, output a single integer followed by a newline — the length of the longest subsequence Mary of Fred with the desired properties.

Example

antimonotonicity.in	antimonotonicity.out
4 5 1 2 3 4 5 5 5 4 3 2 1 5 5 1 4 2 3 5 2 4 1 3 5	1 2 5 3

Problem H. Humidex

Input file: humidex.in
Output file: humidex.out

Adapted from Wikipedia, the free encyclopedia

The humidex is a measurement used by Canadian meteorologists to reflect the combined effect of heat and humidity. It differs from the heat index used in the United States in using dew point rather than relative humidity.

When the temperature is 30 C (86 F) and the dew point is 15 C (59 F), the humidex is 34 (note that humidex is a dimensionless number, but that the number indicates an approximate temperature in C). If the temperature remains 30 C and the dew point rises to 25 C (77 F), the humidex rises to 42.3.

The humidex tends to be higher than the U. S. heat index at equal temperature and relative humidity.

The current formula for determining the humidex was developed by J. M. Masterton and F. A. Richardson of Canada's Atmospheric Environment Service in 1979.

According to the Meteorological Service of Canada, a humidex of at least 40 causes "great discomfort" and above 45 is "dangerous." When the humidex hits 54, heat stroke is imminent.

The record humidex in Canada occurred on June 20, 1953, when Windsor, Ontario hit 52.1. (The residents of Windsor would not have known this at the time, since the humidex had yet to be invented.) More recently, the humidex reached 50 on July 14, 1995 in both Windsor and Toronto.

The humidex formula is as follows:

$$\begin{aligned} \text{humidex} &= \text{temperature} + h \\ h &= (0.5555) \cdot (e - 10.0) \\ e &= 6.11 \cdot \exp\{5417.7530 \cdot ((1/273.16) - \\ &\quad -(1/(\text{dewpoint} + 273.16)))\} \end{aligned}$$

where $\exp\{x\}$ is 2.718281828 raised to the exponent x.

While humidex is just a number, radio announcers often announce it as if it were the temperature, e. g. "It's 47 degrees out there . . . [pause] . . . with the humidex,". Sometimes weather reports give the temperature and dewpoint, or the temperature and humidex, but rarely do they report all three measurements. Write a program that, given any two of the measurements, will calculate the third.

You may assume that for all inputs, the temperature, dewpoint, and humidex are all between −100 C and 100 C.

Input

Input will consist of a number of lines. Each line except the last will consist of four items separated by spaces: a letter, a number, a second letter, and a second number. Each letter specifies the meaning of the number that follows it, and will be either `T`, indicating temperature, `D`, indicating dewpoint, or `H`, indicating humidex. The last line of input will consist of the single letter `E`.

Output

For each line of input except the last, produce one line of output. Each line of output should have the form:

`T` number `D` number `H` number

where the three numbers are replaced with the temperature, dewpoint, and humidex. Each value should be expressed rounded to the

nearest tenth of a degree, with exactly one digit after the decimal point. All temperatures are in degrees celsius.

Example

humidex.in	humidex.out
T 30 D 15 T 30.0 D 25.0 E	T 30.0 D 15.0 H 34.0 T 30.0 D 25.0 H 42.3

Problem I. Tour de France

Input file: tourdefrance.in
Output file: tourdefrance.out

A racing bicycle is driven by a chain connecting two sprockets. Sprockets are grouped into two clusters: the front cluster (typically consisting of 2 or 3 sprockets) and the rear cluster (typically consisting of between 5 and 10 sprockets). At any time the chain connects one of the front sprockets to one of the rear sprockets. The drive ratio — the ratio of the angular velocity of the pedals to that of the wheels — is n:m where n is the number of teeth on the rear sprocket and m is the number of teeth on the front sprocket. Two drive ratios $d_1 < d_2$ are adjacent if there is no other drive ratio $d_1 < d_3 < d_2$. The *spread* between a pair of drive ratios $d_1 < d_2$ is their quotient: d_2/d_1. You are to compute the maximum spread between two adjacent drive ratios achieved by a particular pair of front and rear clusters.

Input

Input consists of several test cases, followed by a line containing 0. Each test case is specified by the following input:

- f: the number of sprockets in the front cluster;
- r: the number of sprockets in the rear cluster;
- f integers, each giving the number of teeth on one of the gears in the front cluster;
- r integers, each giving the number of teeth on one of the gears in the rear cluster.

You may assume that no cluster has more than 10 sprockets and that no gear has fewer than 10 or more than 100 teeth.

Output

For each test case, output the maximum spread rounded to two decimal places.

Example

tourdefrance.in	tourdefrance.out
2 4 40 50 12 14 16 19 0	1.19

Problem J. Texas Trip

Input file: texastrip.in
Output file: texastrip.out

After a day trip with his friend Dick, Harry noticed a strange pattern of tiny holes in the door of his SUV. The local American Tire store sells fiberglass patching material only in square sheets. What is the smallest patch that Harry needs to fix his door?

Assume that the holes are points on the integer lattice in the plane. Your job is to find the area of the smallest square that will cover all the holes.

Input

The first line of input contains a single integer T expressed in decimal with no leading zeroes, denoting the number of test cases to follow. The subsequent lines of input describe the test cases.

Each test case begins with a single line, containing a single integer n expressed in decimal with no leading zeroes, the number of points to follow; each of the following n lines contains two integers x and y, both expressed in decimal with no leading zeroes, giving the coordinates of one of your points.

You are guaranteed that $T \leq 30$ and that no data set contains more than 30 points. All points in each data set will be no more than 500 units away from $(0, 0)$.

Output

Print, on a single line with two decimal places of precision, the area of the smallest square containing all of your points. An answer will be accepted if it lies within 0.1 of the correct answer.

Example

texastrip.in	texastrip.out
2 4 -1 -1 1 -1 1 1 -1 1 4 10 1 10 -1 -10 1 -10 -1	4.00 242.00

Problem K. Gangs

Input file: gangs.in
Output file: gangs.out

The downtown core of Inner City is laid out as a grid, with numbered streets running north-south from 1st Street in the west to 20th Street in the east, and numbered avenues running east-west from 1st Avenue in the north to 20th Avenue in the south. The area is controlled by two gangs, the Blips and the Cruds. The boundary between their territory is the Green Line, running diagonally from the intersection of 1st Street and 1st Avenue to the intersection of 20th Street and 20th Avenue. The Blips control the area to the southwest of the Green Line, and the Cruds the area to the northeast.

To prove their virility, the Blips go on "runs" through Crud territory, starting at 1st Avenue and 1st Street and ending at a point on the Green Line that varies from night to night. A run may return to the Green Line in between but never crosses it. A run uses avenues only in the east direction and streets only in the south direction. Thus a run can be described by a string of E's and S's of length $2N - 2$; such a run ends at Nth Street and Nth Avenue.

The Blips judge the runs made on a given night (all of which have the same length) by how "OG" they are. A run R1 is more OG than a run R2 if and only if:

- R2 returns to the Green Line for the first time at an earlier point than when R1 returns to the Green Line, or

- R1 and R2 return to the Green Line at the same point, but the portion of R1 to that point (ignoring the initial E and final S) is more OG than the portion of R2 to that point (also ignoring the initial E and final S), or
- R1 and R2 return to the Green Line at the same point and are identical to that point, but the rest of R1 is more OG than the rest of R2.

Examples corresponding to these three cases:

- EESS is more OG than ESES.
- EEESSS is more OG than EESESS
- EESSEESS is more OG than EESSESES.

If all the runs of a given length are ordered according to how OG they are, then the rank of a run is its position in the resulting list. EESS has rank 1 and ESES has rank 2.

Your task is to write a program to help the Blips plan and judge their nightly activities.

Input

The input to the program is a series of instances followed by 0 0. An instance consists of a line containing a positive integer N, representing the terminus of that night's run (Nth Street and Nth Avenue), followed by positive integer M.

Output

The output corresponding to each instance is the run of length $2N-2$ of rank M, or ERROR if there are fewer than M runs of length $2N-2$.

Example

gangs.in	gangs.out
3 1 3 2 3 3 0 0	EESS ESES ERROR

Problem L. Tournament

Input file: tournament.in
Output file: tournament.out

Every September, the Kingdom of Loowater holds a jousting tournament. In each of a series of event, a pair of knights attempt to knock each other from their respective horses . The winning knight is paired with another, while the loser is eliminated. This process continues until all but one knight is eliminated; this knight is declared champion.

The tournament schedule is organized so that no knight needs to compete in more than e events to be champion, for the minimum possible e given k, the number of knights. In order to construct the schedule, it may be necessary to identify several knights who compete in fewer than e events; these knights are said to be awarded a bye and are excluded from the first round of competition.

The first round of competition involves pairing as many knights as possible among those who are not awarded a bye. The competition is more interesting if the knights in each pair are as evenly matched in ability as possible. You are to determine which knights should be awarded a bye so as to make the first round as interesting as possible.

Input

Standard input consists of several test cases followed by a line containing 0. Each test case begins with an integer $1 < k \leq 2500$, the number of knights. k lines follow, each giving the name and ability of a knight. The name is a string of lower case letters not longer than 20; the ability is a real number.

Output

The mismatch between knights with abilities a and b respectively is defined to be $(a-b)^2$. For each test case, output the names of the knights to be given a bye such that the sum of all mismatch values for pairs of knights competing in the first round is minimized. If there are several solutions, any will do.

Output an empty line between test cases.

Example

tournament.in	tournament.out
3 gallahad 10 lancelot 11 mccartney 2 0	mccartney

Problem M. Wedding

Input file: wedding.in
Output file: wedding.out

Up to thirty couples will attend a wedding feast, at which they will be seated on either side of a long table. The bride and groom sit at one end, opposite each other, and the bride wears an elaborate headdress that keeps her from seeing people on the same side as her. It is considered bad luck to have a husband and wife seated on the same side of the table. Additionally, there are several pairs of people conducting adulterous relationships (both different-sex and same-sex relationships are possible), and it is bad luck for the bride to see both members of such a pair. Your job is to arrange people at the table so as to avoid any bad luck.

Input

The input consists of a number of test cases, followed by a line containing 0 0. Each test case gives n, the number of couples, followed by the number of adulterous pairs, followed by the pairs, in the form "4h 2w" (husband from couple 4, wife from couple 2), or "10w 4w", or "3h 1h". Couples are numbered from 0 to $n-1$ with the bride and groom being 0w and 0h.

Output

For each case, output a single line containing a list of the people that should be seated on the same side as the bride. If there are several solutions, any one will do. If there is no solution, output a line containing "bad luck".

Example

wedding.in	wedding.out
10 6 3h 7h 5w 3w 7h 6w 8w 3w 7h 3w 2w 5h 0 0	1h 2h 3w 4h 5h 6h 7h 8h 9h