Manitoba High School Programming Contest

May 24 2013.

12:30-3:30

Rules

- Do not open this problem package until instrcuted to do so.
- All input should be read from standard input (the keyboard) and written to standard output (the screen).
- No text boxes or input windows should be used.
- Any programming language resources and notes are allowed.

UNIVERSITY OF MANITOBA

Department of Computer Science

Problem 1 – Central Binomial Coefficients

The central binomial coefficients are $\binom{2n}{n}$, the largest binomial coefficient in the even rows of Pascal's triangle. Pascal's triangle starts with

```
                1
              1   1
            1   2   1
          1   3   3   1
        1   4   6   4   1
      1   5   10  10  5   1
    1   6   15  20  16  6   1
```

So the first few central binomial coefficients are 1,2,6,20. Even if you don't know Pascal's Triangle, there is a simple formula for the central binomial coefficients:

- $a_0 = 1$
- $a_n = a_{n-1}*(4*n-2)/n$

Write a program that prints all the values of the central binomial coefficients from n=0 to n=14.

Input

This program does not need any input. The program will start, print its output and then quit.

Output

The program should output the values of the central binomial coeffients, each on a separate line. The program should end after writing all 15 numbers.

Problem 2 – Multiply perfect numbers

A number *n* is said to be *multiply perfect* if the sum of all the divisors of *n* is equal to *k*n* for some *k* ≥1 (a divisor of *n* is a number *d* such that *d* divides *n*). For instance, the number 120 is a multiply perfect number since the divisors of 120 are 1, 2, 3, 4, 5, 6, 8, 10, 12, 15, 20, 24, 30, 40, 60 and 120, and the sum of these numbers is 360 which is 3 times 120.

Write a program that takes a number *n* and outputs either "not multiply perfect" if *n* is not multiply perfect or the number *k* if *n* is multply perfect and the sum of its divisors is *k*n*.

Input

The input file consists of several positive numbers, followed by the number zero. Each number will be on its own line. The number zero indicates that you are at the end of the input: do not proces the number of zero as input. Simply terminate when zero is reached.

Output

Sample Input	Sample Output
6 7 120 0	2 not multiply perfect 3

Judging Data for Problem 2

```
1
2
3
4
5
6
7
8
9
10
100
120
360
1000
30240
123456789
523776
23569920
17363022
0
```

Problem 3 – Gold, Silver, Bronze

You are responsible for determining the medal winners for your school's track meet. You are given a list of finish times for each runner, and need to determine the gold, silver and bronze medallists. The times appear in the order they come in from different heats, so they are not in order.

Input

Each line of input consists of the results from a single event. The first integer in the line, *n*, gives the number of competitors in the event. The next *n* floating point values give the race times for each competitor, in seconds and fractions of a second. There are always at least three competitors per race. You are also guaranteed that each finishing time is unique.

Output

For each race, list the three medal-winning times as follows:
"Race X
GOLD: nnn.nn
SILVER: nnn.nn
BRONZE: nnn.nn"

Races should be numbered starting at 1. Do not output any blank lines between the race results. Output the same number of decimal places as in the input data.

Sample Input	Sample Output
5 10.1 10.2 10.3 10.4 10.5 3 9.5 9.2 9.6	Race 1 GOLD: 10.1 SILVER: 10.2 BRONZE: 10.3 Race 2 GOLD: 9.2 SILVER: 9.5 BRONZE: 9.6

Judging Data for Problem 3

(*Note: all inputs for one test case are on one line. Recall that the integer at the beginning of the line gives the number of race results to process.*)

```
21 9.57 64.33 107.66 103.96 100.19 85.66 32.96 42.98 81.40 62.62 118.98 88.95 122.09
53.07 30.01 57.39 53.82 93.75 88.59 89.74 119.25
14 35.34 73.36 110.42 88.23 87.02 104.83 62.51 33.99 66.83 89.61 52.06 64.96 14.99 26.51
30 86.55 57.57 116.16 55.21 21.82 94.72 94.50 69.38 62.27 32.94 87.14 88.63 30.55 86.49
93.47 67.71 17.54 56.95 60.41 56.24 89.36 16.94 34.62 49.41 75.40 103.56 109.58 27.49
80.38 75.15
10 36.94 40.49 85.38 57.30 45.99 34.34 57.59 37.62 98.25 115.75
22 49.21 102.34 49.74 66.01 46.58 39.86 21.08 97.43 108.81 23.63 57.03 85.67 13.15 13.05
57.79 15.99 124.57 14.56 17.52 28.36 36.31 99.66
23 89.62 49.64 23.65 30.08 100.09 99.50 104.34 17.37 90.32 30.42 23.50 47.31 111.97 77.54
37.39 95.74 108.39 98.99 43.21 18.54 94.98 99.59 82.47
15 42.19 58.59 108.48 93.61 93.10 47.20 60.71 52.35 111.72 90.88 76.66 58.14 44.62 112.98
34.26
10 104.08 37.22 90.42 67.29 69.83 74.72 18.04 39.29 8.74 42.38
28 52.00 112.52 82.99 11.75 99.54 98.29 104.80 111.36 56.87 103.69 63.00 59.85 96.24
65.51 111.92 81.60 31.15 21.10 38.54 15.41 101.83 71.17 101.76 48.45 61.51 15.80 116.55
108.02
11 102.11 86.75 97.71 94.36 33.84 92.38 87.07 52.35 74.80 40.53 69.19
24 25.68 29.14 42.87 54.45 105.75 93.65 22.39 107.83 56.35 93.20 100.14 81.28 42.23 67.71
89.96 76.74 96.58 68.59 57.11 81.74 14.20 113.22 42.35 55.37
21 98.63 95.15 34.29 23.66 63.56 106.44 88.45 60.82 75.49 90.11 121.82 82.90 38.78 61.28
46.42 34.62 43.60 97.46 52.70 119.00 114.80
13 34.82 85.87 71.36 64.08 71.83 22.05 112.44 59.41 30.87 20.95 62.88 74.72 22.38
25 91.30 33.09 59.31 28.69 87.15 35.09 73.52 61.43 47.28 59.79 83.42 85.56 79.63 86.57
59.47 120.10 102.54 29.00 114.36 92.79 68.33 37.47 80.53 98.25 73.05
30 21.94 74.07 59.68 124.44 7.17 68.63 67.85 33.06 76.43 23.55 95.62 10.65 48.77 43.13
66.70 73.47 120.00 68.16 66.60 65.64 38.02 44.76 86.64 87.02 60.74 26.61 93.72 73.09
28.53 93.34
23 12.64 103.44 82.13 84.24 87.54 82.71 25.03 46.18 90.40 88.23 82.15 100.49 94.81 33.53
74.55 46.27 32.03 33.93 98.49 38.55 19.95 52.00 30.67
16 61.75 78.92 72.48 44.98 34.50 31.71 31.70 53.87 72.89 96.15 55.62 109.22 21.93 85.08
51.52 60.70
19 51.53 108.42 103.66 77.13 64.66 29.06 70.37 69.68 123.02 47.97 9.80 111.11 61.82 92.39
112.59 82.08 64.39 27.71 97.97
18 70.88 54.03 86.63 56.67 67.85 75.31 17.12 100.55 108.86 99.13 107.63 17.99 86.39 38.60
59.79 60.40 29.92 25.42
15 121.22 22.91 80.14 79.61 110.82 95.51 103.48 52.52 78.56 122.65 86.45 47.00 103.69
59.33 58.66
```

Problem 4 – Pebbles

Bernard and Frederick have invented a very easy two-person boardgame they call Pebbles. The game works like this: the board is divided into a number of squares that form a single track. Each player has a set of playing pebbles that they place on squares on the board. The action of a player on a square is determined by the number of pebbles on that square:

- if there is an odd number of pebbles on the square, the player goes forward by the number of pebbles on the square.
- if there is an even number of pebbles on the square, the player goes backwards by the number of pebbles on the square.

There are no dice in the game: all of the movement is determined by the number of pebbles on a square. The players play several rounds of placing the pebbles on the board and taking turns being the "active player". The active player starts on the first square of the board and moves according to the rules to figure out where he or she ends up. If the active player arrives at the last square of the board (or beyond), he or she wins the round. If the active player tries to move before the first square of the board or gets caught in a loop, he or she loses the round.

For example, suppose the board looks like this:

3	2	2	1	4	7	1	2	0	1
1	2	3	4	5	6	7	8	9	10

The number of pebbles in a square is written inside the box. The number below the square is the number of the square (from 1 to 10).

An active player starting in the first square would then move forward (3 is odd) to square 4. There is one pebble in this square, so the player moves forward to square 5. In this square, there is an even number of pebbles, so the active player moves backwards that number of squares, to square 1. But now the active player is stuck in an infinite cycle, so he or she will never win.

Your job is, given the number of pebbles on each square of the board, to determine if the active player wins the round (by getting to the end of the board or beyond) or loses (by failing to make it to the end).

Input

The first line of input is the number of test cases.

Each test case consists of two lines. On the first line is *N*, the number of squares on the board. The number *N* is guaranteed to be at least one.

The second line contains N numbers, each separated by one space. This represents the pebbles on each square of the board.

Output

Output one of two answers for each board: "`Active player wins.`" or "`Active player loses.`" according to whether or not the player reaches the last square of the board (or beyond) after starting on the first square in the board and following the rules.

Sample Input	Sample Output
2 10 3 2 2 1 4 7 1 2 0 1 5 1 1 1 1 1	Active player loses. Active player wins.

Judging Data for Problem 4

```
18
1
0
1
1
1
2
2
0 1
2
1 0
2
1 1
2
2 0
2
3 0
10
3 2 2 1 4 7 1 2 0 1
5
1 1 1 1 1
10
4 2 3 5 10 7 2 5 13 14
8
0 1 2 3 4 5 6 7
7
1 2 3 4 5 6 7
8
1 3 5 7 9 11 13 15
8
1 3 5 7 8 11 13 15
8
1 3 5 7 4 11 13 15
1000
21 8 12 25 2 13 10 26 17 24 25 14 29 10 14 21 10 10 5 18 20 3 2 5 12 9 17 17 12 16 4 8 9
28 4 19 4 34 18 15 25 27 20 11 25 20 10 14 20 10 22 20 4 30 8 29 14 16 0 4 4 34 7 6 12 11
6 24 0 30 23 9 27 17 1 16 8 26 33 9 34 14 28 19 27 26 1 15 33 2 22 4 7 24 22 7 12 4 15 20
31 15 18 30 16 31 27 26 3 16 3 4 7 18 26 28 5 9 1 21 3 24 5 24 27 6 23 28 7 1 12 3 31 19
17 30 30 23 30 9 9 33 8 17 18 21 17 11 9 20 22 2 27 3 9 17 12 24 18 3 5 2 26 34 30 10 0 6
9 4 25 4 29 9 26 31 11 1 22 24 8 0 0 9 14 15 33 5 2 22 6 30 5 1 33 30 28 1 29 12 7 7 30
21 0 15 11 17 2 34 4 23 32 16 13 20 11 3 4 31 17 20 4 0 16 31 8 1 4 19 7 13 32 2 10 15 11
4 17 8 17 5 7 27 11 15 12 16 0 26 16 21 12 15 31 2 18 30 15 32 19 18 18 16 34 16 28 14 10
12 17 2 4 24 25 4 21 2 22 9 12 21 12 0 29 29 10 6 27 1 27 14 0 12 34 12 4 31 27 8 6 18 17
13 29 15 25 25 34 15 31 25 1 31 14 2 0 27 11 8 11 11 8 31 29 1 30 24 14 18 6 22 33 26 29
27 33 11 0 24 12 9 2 9 11 22 32 27 12 23 3 2 6 14 21 0 13 16 17 23 23 25 20 23 22 14 23 3
9 2 16 4 20 26 22 7 5 22 25 4 29 11 2 10 11 2 18 6 17 30 10 4 33 21 30 9 29 34 30 21 21 6
1 28 21 21 14 11 10 30 11 11 31 18 19 2 22 29 34 32 26 20 33 19 22 34 27 7 18 32 4 22 18
7 15 18 12 18 15 23 31 26 7 5 0 21 15 5 8 26 27 27 17 0 3 20 11 16 30 1 2 5 9 4 4 34 30
10 11 23 5 24 7 5 13 29 9 14 6 17 4 16 28 10 14 29 2 24 15 31 29 22 18 18 33 25 14 18 34
7 25 21 28 16 24 9 8 32 11 16 2 20 29 21 20 9 7 18 33 7 25 31 29 18 6 19 17 3 21 31 8 6 1
15 4 7 4 32 24 18 18 12 24 18 26 9 11 9 21 3 31 11 4 29 10 34 9 20 32 10 33 16 13 7 10 1
11 8 5 2 14 26 13 16 23 2 5 30 1 18 19 0 24 23 0 34 28 33 6 34 15 22 18 9 8 24 6 27 5 33
4 20 15 16 2 27 23 1 0 9 29 29 25 17 16 33 14 27 13 14 19 25 0 20 17 27 22 5 1 12 16 3 0
7 3 9 21 6 21 30 21 2 24 31 6 33 31 30 28 21 29 2 31 6 4 1 23 28 31 3 6 29 27 13 25 31 34
20 27 29 7 20 28 3 8 28 13 26 12 11 31 21 21 2 13 5 3 29 34 28 2 6 19 3 9 9 18 9 10 10 22
28 22 29 6 15 34 31 16 5 19 31 25 29 23 6 32 9 8 20 19 6 31 22 19 13 30 2 31 14 18 15 5 1
```

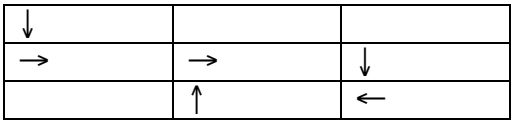
19 9 24 10 23 24 22 20 12 15 6 23 22 2 11 9 23 15 32 1 29 30 3 5 29 5 16 29 4 33 7 15 14
8 19 16 34 8 32 3 25 1 16 12 13 33 33 4 0 21 13 6 8 14 3 8 9 23 9 29 26 8 20 13 28 20 2
28 12 21 5 4 7 9 2 1 23 12 15 29 18 22 25 10 31 3 29 6 5 7 11 32 17 11 18 13 11 28 9 5 4
23 5 15 9 19 33 27 2 11 27 29 20 10 7 13 29 12 3 26 17 32 9 19 18 26 12 33 17 15 7 12 0 5
9 13 8 28 26 1 9 22 21 18 7 17 26 14 12 0 33 0 24 29 16 9 31 16 11 28 1 22 32 18 24 31 1
12 11 19 20 6 14 16 22 1 33 21 9 12 8 1 14 5 11 14 8 30 19 4 31 9 11 0 17 26 24 12 8 19
33 31 28 18 33 8 20 10 33 30 18 13 11 30 28 2 34 9 12 31 29 13 1 33 6 31 10 30 21 16 0 32
6 3 10 12 16 23 4 12 14 7 32 8 8 22 24 30 18 16 25 18 33 29 32 23 27 21 21 18 13 19 14 16
25 6
1000
3 170 12 5 42 94 17 7 7 57 37 171 184 100 9 11 191 150 149 106 12 47 131 183 19 190 13
142 143 177 199 98 3 117 85 150 32 129 172 17 77 122 136 114 160 94 22 49 138 104 92 60
36 196 182 76 19 4 89 14 29 151 108 5 4 176 0 57 172 98 143 149 170 90 43 23 118 29 146
172 185 152 110 114 9 76 176 138 73 107 2 172 106 91 73 69 125 136 29 0 93 42 11 8 7 130
68 15 157 56 9 73 159 199 116 196 184 72 159 26 185 102 108 89 179 15 109 31 97 20 157 30
103 190 141 47 98 67 155 81 34 149 188 63 115 32 51 95 56 98 17 99 113 160 38 18 31 63 37
65 95 17 26 18 0 29 89 169 17 125 147 184 81 42 191 178 74 107 79 32 138 58 32 31 114 134
144 24 178 182 196 195 3 107 168 41 106 133 88 151 55 80 21 68 12 146 165 78 50 192 143
173 59 3 153 168 42 96 125 10 56 4 130 45 10 35 43 143 143 26 159 35 64 184 176 105 43 67
165 127 24 170 62 23 100 31 25 9 138 68 104 67 13 108 121 188 84 93 0 76 116 61 177 142
78 66 88 177 101 61 91 63 106 76 110 155 129 46 27 47 191 140 11 116 145 26 104 135 133
165 160 126 56 19 140 28 70 127 56 35 188 30 92 39 192 155 165 78 99 38 65 89 183 88 101
20 65 26 152 62 1 135 69 113 103 96 53 198 49 53 14 122 71 192 5 192 47 125 102 20 3 165
192 23 159 74 19 189 62 6 31 11 6 126 104 76 46 129 109 31 84 89 95 66 18 70 147 131 68
42 168 194 58 62 60 196 122 86 150 59 97 144 35 87 26 89 32 166 140 155 15 182 97 137 117
23 75 93 141 89 70 33 99 108 45 31 197 196 10 112 148 181 5 169 170 107 95 23 177 162 86
19 30 104 56 14 118 186 39 183 145 142 54 146 100 106 14 57 61 19 60 109 79 132 53 105 37
82 155 101 141 140 105 6 29 105 77 164 13 71 179 121 162 79 107 173 39 13 118 160 8 23 96
66 5 128 153 87 89 64 184 189 77 67 40 121 92 156 113 102 63 160 144 175 66 81 138 155
176 67 135 62 142 122 190 50 146 163 174 169 89 11 35 24 84 87 162 169 85 198 58 88 72
114 24 142 21 113 170 49 21 103 110 64 49 88 92 150 121 41 36 196 89 170 6 16 77 33 26 17
143 20 66 40 132 180 60 191 160 172 23 80 184 152 194 151 71 85 179 36 131 132 99 36 93
154 19 186 103 45 6 50 48 141 178 87 115 186 24 185 141 143 88 21 111 4 194 175 193 100
38 63 102 5 190 129 85 128 137 142 163 125 6 58 136 179 146 7 171 20 103 29 8 197 51 123
117 182 39 134 97 172 111 193 80 11 21 73 40 24 182 130 14 0 49 69 195 83 121 10 59 25
195 153 199 155 86 19 11 86 17 94 69 50 39 128 130 158 82 70 31 90 7 175 171 175 109 44
75 184 151 80 50 154 181 36 176 119 39 185 71 178 106 58 153 13 183 41 2 132 74 151 142
149 61 15 123 101 191 22 79 164 49 179 146 31 40 146 21 75 96 113 106 27 139 92 80 59 61
159 5 163 174 79 28 62 158 58 9 81 171 46 10 29 70 104 187 21 175 199 133 44 68 19 65 29
160 67 160 120 180 183 13 40 158 192 81 62 21 117 24 167 153 22 63 26 137 113 129 179 109
72 136 119 187 184 101 187 191 83 24 3 151 110 156 179 31 33 119 20 172 95 160 185 77 20
32 60 43 10 45 59 28 148 64 178 134 3 80 75 158 103 157 116 109 33 153 73 52 47 47 53 13
143 147 69 186 42 185 129 90 133 193 75 129 102 26 79 65 146 185 190 180 99 9 169 18 123
153 103 136 25 152 95 72 144 180 153 115 122 63 129 89 141 107 166 3 199 88 1 69 42 57 8
170 155 177 79 192 127 136 20 150 82 89 165 41 35 147 44 107 135 180 43 173 28 2 193 151
89 178 110 183 3 166 159 71 189 39 69 26 133 51 136 52 148 34 185 74 163 173 2 145 31 193
76 9 180 132 81 121 5 97 96 95 190 128 65 162 176 58 184 31 47 168 87 137 5 154 142 34 97
125 176 37 19 14 80 112 46 15 67 23 63 180 176 107 13 115 97 141 38 31 85 88 35 104 47
187 8 130 114 113 31 72 46 154 62 76 40

Problem 5 – Do the Fubbler.

You've been swept up in the newest dance craze, called the FBLR (read "Fubbler"). Dancing the FBLR is really easy: you divide a dancing region into a grid of sections, then you are given a sequence of letters F, B, L and R. Starting at the north-west section of the grid and facing south, you follow the sequence of letters (Forward, Backwards, Left and Right).

Your path is given as a sequence of characters F, B, L, R, representing Forward, Backwards, Left and Right. If you are facing a certain direction, these instructions tell you which of the neighbouring sections you should end up standing on for your next dance step. Note that these instructions (F, B, L, R) depend on what direction you are currently facing. If you are facing North, an F will move you to the next section to the north (an leave you facing North) but if you are facing East, an L will move you to the next section to the north (leaving you facing North). In this way, each letter represents both a turn and a move.

For instance, if you had a 3-by-3 grid and your sequence was FLFRRR, then your dance path would be



Your final section in the grid would be the center space.

You've found a local park that's allowed you to use a field to do the FBLR, but after a few weeks, there's been some concern about the grass in the field being trampled by all the dancing. So you have been asked to submit information about all different dances you are doing. For each dance sequence, you are asked to determine if a grid section is visited once, more than once, or not at all.

The section in the north-west corner of the grid is considered to be visited before you start your dance (so even if the dance has zero steps, the north-west corner space is visited once).

Input

The input consists of several test cases, and a final line indicating the end of the test cases.

The first line of each test case is the size of the grid in the park, given by two integers one one line, h and w. The integers satisfy $1 < h,w \leq 50$.

The next line is a sequence of the letters F,B,L and R. The length of the sequence may be zero, in which case a blank line appears between two grid sizes.

The inputs are guaranteed to never take you out of the grid area.

The final line of input is `0 0`. This represents the end of the input and should not be processed.

Output

Output a map of the area representing the status of the grass after the dance is complete. Each grid section is represented by one character, with no spaces between the grid sections. Print a period (.) if the grass is never visited, a capital letter x (X) if the grass is visited once and a pound symbol (#) if the grass is visited more than once.

Print a blank line after every map.

Sample Input	Sample Output
3 3 FLFRRR 3 4 FBRB 0 0	X.. X#X .XX #X.. X... ....

Judging Data for Problem 5

```
2 2
F
2 2
L
2 2
FL
2 2
LR
2 2
FLLL
2 2
FLLLLL
2 2
FLLLLLL
3 3
FLFRRR
3 4
FBRB
10 10
FBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBBB
10 18
FFFFFFFFFBFFFFFFFFRRLRLRLRLRLRLRLLRLRLRLRLRLRLRLRRFFFFFFFF
7 9
LFFFFFFFBFFFLFFFFFRFFFBFFFFFFF
10 6
FFFFFFFFFBFFFRRLRLRLRLBRLRLRLRLRRLRLRLRLR
11 10
FFFFFFFFFFLFFFFFFFFBFFFFFFFFRFFFFRFFFFBFFFFRFFFFRFFFFFFFF
30 30
LRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLBLLRLFBFRFRFLRRRFBLFRRFFBFLBBBBRRBFFBLR
50 50
LRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRRFFLFBLFBFFBLRBLLFFLLLFFRRLFBLLLFRRFFFR
FRBLBFRBRLRFRFBRLLLRLLLRBFBFLBRRFFFRFFRFFLFLRBFLRLLBBRLRFFBFLBBBLBBLRBBBBFRBBRLRLLRBLBRLR
FFFRFLLBLLRRRRRRRLFLBRBLBBLFRRBFL
50 50
LRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRLRRBFLBFFRLBRBLFFRBRRLRRFLLFRRBFBRBFRRFRF
RFFFLRBRFBBBRFRRBFRLRBBFRRFFFRFLFBBBRBFLRFFFFRBFLLBLLLRBLFRBFLRBLLFRBRBLRBBBFBRFFLLBFBFBL
BRRLRLLLLRLBLLLBFRFRLFFRLBBBFBRRLFL
2 2

0 0
```

Problem 6 – Public Consultations

Ron is the director of the parks and recreation department in a small midwestern US city. Ron doesn't like his job and he strives to do as little work as possible. One of the jobs Ron likes the least is attending public consultation meetings. As a result, he gets his deputy director, Leslie, to schedule as many as possible on one day, and then he tries to visit some of them as quickly as possible and go back to sitting in his office, undisturbed. Ron doesn't need to visit all the consultations: he just needs to make it look like he cares about the public by visiting some of them.

However, Ron can't simply visit some of the consultations and then backtrack to his office. At each consultation, Ron makes an excuse for leaving, and then needs to steer clear of that venue for the rest of the day.

Given all the public consultations in a day, and the road connections between them, determine if there is a way for Ron to visit some set of public consultations and then return to his office, all while not travelling on the same road twice.

Input

The input starts with the number of test cases (`N`) on its own line.

A single test case starts with a line with two numbers on it: `n`, the number of public consultations, and `m`, the number of road connections between the consultations. The consultations are numbered `0` to `n-1`. The number `0` represents the starting and ending point, Ron's office.

After the first line, there are `m` lines of input, representing the `m` road connections. Each line has two integers `i` and `j` ($0 \leq i,j < n$), representing that Ron can travel between public consultation `i` and public consultation `j` or from `j` to `i`.

Output

For each case, output only "`true`" if there is a path from Ron's office (`0`) back to itself that does not use the same road connection twice. Output "`false`" otherwise.

Sample Input	Sample Output
2 5 4 0 1 0 2 0 3 0 4 6 7 0 1 1 2 2 3 1 4 4 5 5 0 2 1	false true

Judging Data for Problem 6

(*Note: for ease of reading, a blank line has been inserted between each test case below. These blank lines are **not** in the true judging data.*)

```
13

5 4
0 1
0 2
0 3
0 4

6 7
0 1
1 2
2 3
1 4
4 5
5 0
2 1

2 1
0 1

3 2
0 1
0 2

4 2
0 1
2 3

5 4
0 1
1 2
2 3
3 4

4 3
0 1
0 2
0 3

4 4
0 1
0 2
0 3
1 2

4 4
0 1
0 2
0 3
1 3
```

```
7 7
0 1
1 2
2 3
0 4
4 5
5 6
3 6

6 7
0 1
1 2
1 3
0 5
2 4
3 4
4 5

2001 2001
0 1
1 2
2 3
3 4
[lines "j j+1" deleted for j=4..998]
999 1000
0 1001
1001 1002
1002 1003
1003 1004
[lines "j j+1" deleted for j=1004..1998]
1999 2000
1000 2000

2001 2001
0 1
1 2
2 3
3 4
[lines "j j+1" deleted for j=4..998]
999 1000
0 1001
1001 1002
1002 1003
1003 1004
[lines "j j+1" deleted for j=1004..1998]
1999 2000
```