UNIVERSITY OF MANITOBA

DEPARTMENT OF COMPUTER SCIENCE

Manitoba High School Programming Contest 2014

23 May 2014 12:30 – 3:30

Contest Rules:

- Do not open this package until instructed to do so.
- All solutions must be entered completely during the contest. No electronic copies of pre-written code are permitted.
- You may submit as many solutions as you like to each problem; however, incorrect solutions will be assessed a time penalty.
- Contest score is based on the most problems successfully solved; ties are won by shortest total time taken, including any penalties.
- A correct submission must solve the given problem and produce correct output for the given test data within a reasonable time.
- Programming style will not be considered during judging.
- Any programming language resources and notes are allowed.
- No other Internet access is allowed during the contest.

Submission Requirements / Pre-submission Checklist:

- All input must be read from standard input (in Java, `System.in`). Only Problem 1 does not require input. Do not open input files.
- All output must be written to standard output (in Java, use `System.out.print` or `println`).
- Your output format must follow the problem requirements *exactly*.
- Submit the source code file (`.java`, `.c`, etc. – *NOT* `.class`, `.exe`, etc.).
- Java programs must be in a single file and not placed in a package (no `package` statements – `import` statements are of course OK).
- Java programs must be complete; they require a `main` method.

Problem 1 – Box Volumes

For this question, write a program that outputs the volume of a series of twenty boxes. These boxes each have a length (L), height (H) and width (W) that are integers, and satisfy H = L + 1 and W = H + 1.

The lengths of the boxes go from L = 1 to L = 20. So the smallest box has dimensions L = 1, H = 2, W=3 and a volume of 6. The last box has dimensions L=20, H = 21 and W = 22 and a volume of 9,240.

Input

This problem does not have any input. The program will begin, write the output, and then quit.

Output

Output twenty integer outputs, one on each line, representing the volumes of the twenty boxes with lengths from 1 to 20.

Sample Output
6 *(…eighteen more lines of output not shown…)* 9240

Problem 2 – The balance game

The balance game is played with two players and an old-fashioned balance scale, the kind used to measure weight. Players take turns placing rocks on their side of the balance while the other player's back is turned, trying to guess how heavy a rock they need to even out the scales. Every time a player balances the scales, they receive a point. The player with the highest score after all the rocks are played wins.

For this computer simulation of the balance game, you are given a sequence of integers indicating the weight of the rock being played, starting with player 1, followed by player 2, and then alternating between the two players. The signs of the numbers are alternating too; all of player 1's numbers are positive, and all of player 2's are negative (the magnitude of the number is the weight of their rock).

Every time the total weights played on the two sides are equal, the player whose rock was most recently played scores a point.

Input

The first line of the input has a single integer *n*, which is the number of rocks to play. The next *n* lines each contain one integer representing the weight of one rock, starting with a positive value for player 1 and alternating with negative values for player 2.

Output

Print the total number of times the scales were balanced for each player (their score).

Sample Input	Sample Output
12 15 -20 8 -3 12 -14 2 -1 7 -12 6 -4	Player 1 scored 2 Player 2 scored 1

Judging Data for Problem 2

(input data values are one per line but shown in columns to save paper)

256	10	-7	2	-10	8	-4
15	-9	5	-9	6	-3	8
-20	6	-5	4	-5	3	-4
8	-2	4	-10	11	-6	3
-3	2	-8	10	-2	2	-8
12	-8	3	-1	7	-6	6
-14	2	-10	11	-6	6	-2
2	-6	8	-9	6	-7	8
-1	8	-11	11	-4	6	-10
7	-5	6	-6	11	-4	12
-12	2	-10	10	-2	2	-9
6	-12	6	-9	5	-11	6
-4	6	-2	5	-6	11	-3
3	-10	1	-4	7	-5	4
-8	8	-10	5	-11	7	-12
10	-9	6	-11	3	-4	8
-4	2	-10	1	-3	2	-3
11	-6	11	-3	6	-4	5
-3	12	-3	6	-9	6	-3
9	-4	8	-7	10	-6	12
-5	4	-7	9	-11	7	-2
9	-9	8	-3	2	-12	7
-2	10	-2	10	-8	6	-8
8	-9	3	-9	11	-6	9
-8	4	-8	7	-1	5	-8
6	-6	4	-2	10	-7	9
-3	8	-6	9	-8	10	-8
4	-9	5	-4	1	-7	9
-6	6	-8	5	-3	4	-10
11	-8	9	-6	11	-11	4
-7	1	-2	6	-5	2	-6
2	-10	11	-1	6	-3	10
-11	4	-1	9	-9	11	-1
6	-12	10	-7	6	-10	5
-9	6	-11	3	-8	9	-8
5	-3	6	-9	8	-1	
-8	7	-8	8	-1	5	

Problem 3 – Actual Retail Price

Come on down! You are part of a game show where contestants guess the cost of a prize, and the host then reveals the "actual retail price". The contestant that guesses closest to the actual retail price, without going over, wins the prize.

For instance, if you have four contestants whose guesses are $800, $801, $700, and $900, and the actual retail price is $899, the winner is the contestant that guessed $801.

Write a program that takes the guesses of the contestant and the actual retail price and gives the winning guess. If someone correctly guesses the actual retail price, print this out. If all contestants bid over, print out a message indicating that.

Input

The first line of the data is an integer *n*, indicating the number of test cases. Following this, there are n lines of input. Each line of input starts with an integer *k*, the number of contestants. The integer *k* will always be greater than zero. Following this integer, there are *k*+1 floating point numbers. The first *k* floating point numbers are the contestant bids, and the last floating point number on each line is the actual retail price. All the bids will be different from all other bids in the same test case.

All floating point numbers will have two (non zero) decimal points.

Output

For each test case, first output "Case X:" where X is the case number, starting from 1. Then output a single space and one of three messages:

- "Closest bid: Y" where Y is the closest bid without going over.
- "Correct bid: Y" where Y is the bid that is equal to the actual retail price.
- "Everyone went over." when no bids were under the actual retail price.

Sample Input	Sample Output
3 4 1.01 2.02 3.03 4.04 2.55 3 10.99 11.99 12.99 9.99 2 1.01 1000.01 1000.01	Case 1: Closest bid: 2.02 Case 2: Everyone went over. Case 3: Correct bid: 1000.01

Judging Data for Problem 3

```
15
4 1.01 2.02 3.03 4.04 2.55
3 10.99 11.99 12.99 9.99
2 1.01 1000.01 1000.01
1 1.01 1.01
1 1.01 1.02
1 1.01 1.00
2 2.99 2.02 2.01
2 2.99 2.02 2.02
2 2.99 2.02 2.03
2 2.99 2.02 2.99
2 2.99 2.02 3.01
4 4.04 2.02 3.03 1.01 2.55
4 3.03 1.01 4.04 2.02 2.55
100 25.61 63.87 70.61 71.02 35.65 71.64 22.18 22.89 38.69 55.93 29.16 5.49 87.51 18.26
61.38 78.17 66.55 39.01 80.41 8.38 74.77 69.88 13.88 93.44 97.18 68.63 45.42 66.48 78.65
36.75 86.51 52.88 78.91 55.04 33.62 5.06 4.87 61.01 11.03 45.86 46.39 24.85 54.83 69.83
72.99 34.43 43.76 48.53 58.35 32.82 32.66 80.48 65.54 25.97 61.27 61.77 24.09 87.86 46.06
56.91 21.98 93.92 78.06 81.49 55.29 65.15 74.93 10.67 82.55 52.87 99.32 21.74 73.22 17.09
66.88 53.37 56.24 11.84 47.47 96.82 18.17 31.56 14.64 37.15 64.64 41.08 0.67 41.64 43.25
88.67 53.79 61.31 88.76 91.43 5.97 66.43 81.07 37.25 44.15 67.38 70.61
1000 9294.46 3549.27 3306.93 2181.14 9120.64 4944.06 9114.03 8764.41 5185.39 1101.38
3374.08 9115.36 337.02 5681.09 5252.02 6579.49 957.33 8705.85 935.83 459.15 7174.09
1501.78 375.68 1164.13 6497.85 4332.71 2067.96 2756.42 3918.72 3178.71 8385.41 942.67
9622.58 812.64 6270.46 4237.92 3822.74 5583.58 4732.07 1687.12 3296.96 602.69 3433.63
1062.47 4510.74 303.94 7892.01 6424.02 4551.81 8646.03 9974.83 1727.53 393.73 8110.81
3768.46 8658.85 9185.76 6169.86 9077.96 8693.79 4984.97 2669.31 1599.01 575.57 6898.13
3680.06 4225.93 2028.06 2083.66 6021.63 1984.14 8333.94 2763.62 5764.96 9050.17 5466.27
6041.67 6515.12 3210.23 1173.38 534.73 8220.15 1204.78 9160.68 8794.25 6953.45 1633.08
3447.56 7726.81 3478.37 9806.94 6355.48 7633.41 5604.83 3819.11 3932.28 6449.15 5363.59
5497.03 1067.86 1670.61 9428.14 7309.51 7416.31 5781.92 5885.25 6642.87 2529.22 8793.63
8721.96 255.29 2203.45 2362.34 6842.46 74.26 55.99 2543.38 5212.77 8431.69 174.54 5003.66
7067.94 2701.87 4316.94 4353.75 3050.83 9537.27 8266.62 5127.35 3510.37 780.08 8422.37
5298.02 5034.99 6173.65 7975.42 6068.26 3955.88 4081.39 845.28 1149.65 501.55 8798.47
5121.64 5034.12 6725.13 9002.88 1244.98 9202.71 3613.58 9698.29 6833.77 6640.97 9592.16
5645.11 2508.89 5693.81 9894.23 6716.57 2800.56 6236.76 3679.59 5890.91 7887.65 7394.74
8393.39 4689.61 4044.86 2670.61 8338.81 6861.38 8218.22 1196.89 6084.37 6536.15 6116.06
4245.74 4270.76 9792.08 3708.66 8955.41 2486.08 7366.22 5012.41 4009.42 2277.41 2951.61
59.21 186.45 1898.17 4634.61 8626.77 2837.82 9748.86 483.55 1786.42 4396.53 8238.61
3578.03 4992.34 7539.09 5970.32 5046.77 8597.53 679.18 1206.82 5347.44 244.32 6647.37
1791.13 6168.05 4137.17 1515.01 6520.25 8680.35 7110.57 2487.23 4191.18 7389.31 5315.82
7485.31 4078.99 263.08 1487.66 9222.91 5368.84 1356.79 7476.79 3791.08 9457.41 18.58
9521.89 309.52 6730.97 6057.62 7398.04 7870.34 2343.89 9620.67 171.78 1727.51 1484.93
495.37 167.51 4884.85 7461.09 8500.64 8944.35 1233.84 3249.79 9491.74 6776.64 3542.96
6138.77 9494.29 3925.75 9449.91 6165.51 8428.21 8328.66 5294.02 1735.33 5618.89 5448.98
2687.38 780.41 3875.11 5558.91 5604.53 9601.99 8130.45 883.51 5031.17 1389.67 8243.04
3660.39 995.61 8914.96 3258.95 3340.27 3854.29 6507.15 3232.98 5814.81 7104.33 947.72
9067.12 6397.46 2925.91 8379.13 3154.71 5369.94 779.59 9364.38 308.08 3256.89 8464.96
3426.26 1266.92 7094.67 1382.29 9531.23 5186.11 956.93 28.44 5604.59 9255.54 1571.35
9058.02 6521.03 6929.62 2882.74 7629.62 976.78 372.66 4329.09 6078.75 8397.42 6676.07
3259.85 5472.68 1898.69 2482.89 2639.67 7056.76 9504.11 4818.54 8400.78 7812.56 3257.83
1974.43 6042.23 6231.84 2384.64 2345.29 1495.29 9937.96 5683.06 3141.65 1533.18 484.53
8358.89 9777.29 9283.11 7085.47 5633.87 4285.42 4622.49 1351.97 5447.34 899.35 4691.97
6482.73 9493.82 4600.71 2351.41 6694.93 6593.71 3213.97 5225.04 3774.24 7187.73 3003.33
```

(line continues on next page…)

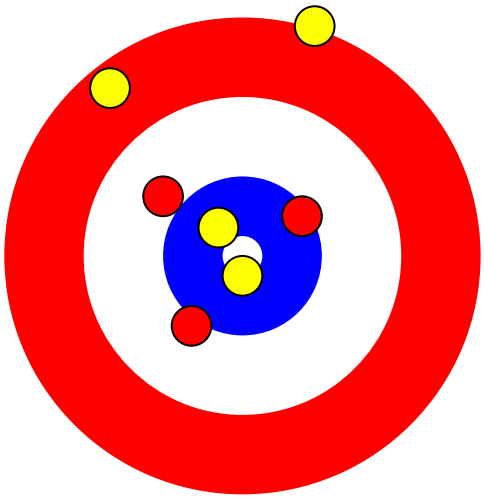
4734.47 3039.14 2326.44 1424.28 6670.51 3390.11 547.89 6599.58 3590.35 4242.36 3507.94
7389.86 389.36 3495.78 129.86 6897.41 5064.98 1555.45 5543.39 8407.38 5875.23 7950.91
6567.56 5119.72 6241.53 5707.57 8768.51 4748.41 182.88 7206.75 7850.88 2907.85 4391.86
2545.65 7966.31 1834.91 4323.29 3383.34 5440.83 3836.21 7805.02 7244.42 7901.88 6037.68
2065.29 7807.11 6280.09 3223.31 1784.92 6934.71 2599.39 1764.63 9839.44 4493.11 4489.35
8645.38 7822.15 8087.89 4606.16 4717.79 1085.01 6622.62 2039.88 8961.63 9564.71 1543.66
9733.31 8209.67 8011.71 2967.32 8264.83 575.44 9854.63 5235.44 7554.19 6881.73 230.92
2396.89 3714.73 9930.15 2614.62 8824.87 8202.31 762.41 333.53 548.13 9644.28 6367.93
3423.52 8229.25 7788.13 920.66 1228.27 6688.09 2198.48 2446.05 9225.42 1253.35 6907.52
500.97 8651.51 5863.64 4274.59 4200.15 9479.23 4820.52 8603.51 8718.66 7433.74 7476.93
1316.86 3609.87 8118.37 2731.35 6428.85 5842.05 5090.69 4544.96 8711.08 9123.68 9327.72
6412.73 216.48 6779.08 9187.71 6175.82 6998.62 9770.44 3437.88 7880.86 6925.26 7113.69
3149.49 6959.86 3934.43 7446.57 7488.84 4822.11 3243.91 8859.18 3284.97 5556.74 5277.55
826.96 4818.73 7410.14 1633.71 2203.29 1971.25 4090.17 4400.49 4958.85 5592.53 3808.88
4558.39 8542.42 3541.92 3671.45 3408.76 2761.48 7135.85 5729.04 9791.78 4088.94 9767.89
184.09 316.63 2463.63 324.74 2887.06 9495.08 7875.66 3591.67 7935.37 7860.06 1566.37
7318.11 3624.64 7662.57 5295.64 8350.03 7661.13 5456.36 6068.66 5587.25 7367.89 3457.25
8634.56 2931.98 1981.43 2912.99 6260.28 7021.31 3678.23 9176.42 2895.74 2788.22 6657.09
9923.09 6181.22 9351.59 9908.24 3419.47 9632.91 8361.68 8047.69 3623.99 6212.49 520.27
5214.01 9515.71 8467.24 3591.21 7933.42 5294.01 8540.55 3708.97 4956.24 6450.32 2892.23
2702.01 7017.98 6594.62 1683.74 5852.51 3103.97 9543.45 9623.61 8688.06 5069.43 4822.56
5510.26 8062.52 1957.97 1234.56 4897.79 1398.69 9377.07 6335.28 3863.78 2998.91 8706.82
760.52 6498.41 7407.36 5212.38 1704.37 1470.24 9919.44 6873.18 967.14 2771.83 2985.46
9762.56 3939.49 8510.35 9678.31 4430.85 8500.41 1116.03 6554.36 8562.57 5418.72 1798.82
7073.25 8133.61 7651.28 7140.43 6898.64 8799.43 5680.37 3757.18 5009.27 7702.16 2596.35
9170.83 735.66 5126.63 9101.81 5673.81 1554.56 6981.79 1700.21 3899.44 3357.29 3467.49
3206.47 8344.09 1470.97 7730.35 6270.25 4584.71 4494.05 5278.01 4675.68 5408.45 5548.35
2750.45 6783.48 3541.74 4760.96 8477.28 8181.67 3924.55 5829.42 6226.15 7823.36 2103.43
9867.32 3756.24 2633.62 1039.13 1119.41 3786.15 6399.18 7053.09 9296.25 8293.96 9282.01
3639.74 1011.79 6925.18 3154.59 1351.58 3759.48 4234.59 2574.57 5546.08 7834.67 7462.56
8393.71 5556.75 3842.27 8051.25 5535.98 4997.89 5953.19 1271.25 441.91 8713.94 5074.33
6356.05 1158.99 9534.79 1613.59 4476.19 7.45 8179.98 4483.35 3197.17 8483.13 9783.36
5345.77 5408.33 4768.33 8907.71 3088.63 4928.86 3026.76 2839.19 2397.39 9018.88 9172.75
8383.86 9531.08 6083.09 1370.28 6576.12 1009.06 428.24 8194.14 2619.93 6269.36 6496.78
1920.02 5624.85 2140.35 7697.24 8857.21 2485.16 7710.68 2455.54 1167.28 8565.43 1468.48
1814.11 1681.49 4003.97 4832.82 4262.17 2032.69 3871.61 226.09 949.04 6193.75 3004.62
9153.79 8698.89 3868.52 2826.34 5440.88 4002.25 7172.18 8585.88 8003.15 8793.34 6.51
891.24 9962.93 8828.93 2844.65 1258.56 4394.35 9260.88 4237.33 5014.83 7549.05 7523.12
6820.74 5764.46 5585.71 8235.39 9142.92 6257.47 135.32 2982.79 4745.89 5880.37 1098.94
8066.15 5979.38 4277.05 8527.27 9713.86 5994.72 4419.26 7292.29 4129.41 9125.65 6234.08
9714.92 6016.17 3036.14 2905.91 3704.57 4583.17 9304.95 7189.15 7548.91 5244.67 6341.22
5.84 6595.81 1895.02 980.01 8035.16 7210.56 969.09 4925.47 3873.33 5807.68 9030.88
3591.44 2998.26 4988.63 237.37 9504.48 6613.89 4624.12 4292.65 1017.69 6362.21 313.88
7296.91 475.34 9648.03 9493.02 1725.18 2617.29 4453.02 1234.74 8880.38 3733.64 4643.47
2964.94 4436.19 2010.73 886.63 6293.91 1220.46 2880.29 2753.54 7695.72 4916.27 6445.21
8864.05 29.41 7815.33 9975.05 8474.84 4531.34 9823.05 2334.65 6255.87 6551.45 3403.15
3092.56 3190.68 3874.41 7618.12 5136.01 2655.23 5529.62 3618.25 892.56 8145.89 5683.42
204.25 2227.09 2345.32 8648.06 1934.96 690.95 7794.98 5716.47 5143.18 1088.68 9702.57
9015.41 1562.83 5732.76 4230.27 5810.72 9535.32 5059.68 6153.34 5149.76 9389.71 6118.65
3028.94 5808.55 7069.11 2225.54 8752.23 5468.94 1742.17 7232.97 9650.74 4665.47 112.95
4975.45 8274.18 9831.86 9422.77 8669.34 5857.84 489.17 741.81 332.64 1615.52 7650.23
1808.76 5830.81 937.26 8555.36 3659.25 2188.72 6725.35 4818.06 6040.22 3619.46 3024.61
9600.96 3331.03 2690.02 4672.16 6030.42 9240.03 4849.93 6914.37 7386.13 6342.45 6630.69
8699.95 296.37 8545.13 4339.84 8371.16 9235.08 2928.35 2217.44 2912.11 6989.61 231.92
2447.45 4956.83 8544.21 7622.28 4886.57 2567.09 6572.83 6519.03 7495.04 9154.76 2023.31
8827.45 4381.11 9185.56 1314.98 9327.59 21.05 3142.94 2094.78 4376.48 3891.01 8793.59
1209.34 2705.36 2734.52 2703.88 4154.32 981.97 9308.79 336.85 1546.98 8200.33 5237.61
1955.31 7184.23 4883.19 2748.13 5959.49 9935.65 8640.75 8493.64 7630.18 476.13 5277.61

Problem 4 – Curling Scoring

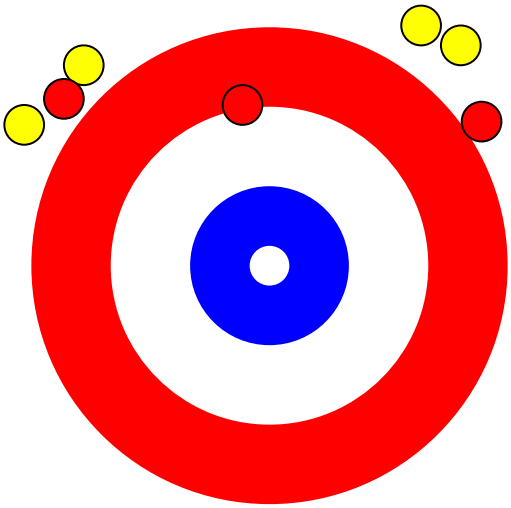
Jennifer is a great curler, but is trying to devise a way to automatically score a curling game based on photos taken of the game.

Here's how curing is scored. Scores are calculated for each end (a game typically has ten ends). After each team throws eight rocks in an end, there are rocks from both of the teams in the scoring region of the sheet of ice (called the house). The team with the closest rock to the centre of the house scores for that end. The other team scores zero points for that end (that is, only one team can score for each end). The scoring team gets one point for every rock that is in the house, until the nearest rock from the non-scoring team is encountered. Only rocks that are in contact with a circle of 6 feet (72 inches) in diameter from the center of the house are scored. Each curling stone has a radius of 5.72 inches. Recall that the distance between two points (x1,y1) and (x2,y2) is given by the formula $\sqrt{(x2-x1)^2+(y2-y1)^2}$.

For example, consider the following diagram, which shows two ends of curling to be scored. In each diagram, the house is represented by the coloured rings and the rocks are represented by the two coloured circles (yellow and red). The colour of the rings in the house is not crucial – only the outer edge of the red ring is important to scoring, as this represents the edge of the 6 foot circle from the centre of the house. In End 1, the yellow team scores 2 points as the team has two yellow rocks closer than any red rock. In End 2, the red team scores two points, as only two red rocks are in contact with the 6-foot radius circle.



End 1: yellow scores 2



End 2: red scores 2

Write a program that calculates the score of an end based on the positions of the rocks.

Input

The input consists of several test cases. Each test case represents the position of rocks in an end. There are three lines of input per test case. The first line of input has two integers (each between 0 and 8 inclusive), which gives the number of rocks in the house for each of the two teams. The integers are *n1* and *n2*, where *n1* is the number of rocks for team 1 and *n2* is the number of rocks for team 2.

The next two lines give the position of the rocks for each team. The first line will have 2**n1* integers and the second line will have 2**n2* integers. For each line, the numbers are pairs of (*x*,*y*) coordinates of the centre of each rock. The coordinates are given in inches, and (0,0) represents the centre of the house. Negative coordinates represent "down" in the y-direction and "left" in the *x*-direction (that is, the coordinates are laid out like the standard Cartesian plane).

The final line of input is -1 -1. This should not be processed as an input batch. As you can see, there are no ties in the input data. One team will always have a rock closer to the centre of the house than the other if there are any rocks present.

Output

For each end, output whether the first team or the second team scored points and how many points were scored. Output should be formatted as "End #X: team Y scores Z" where X is the number of the test case (starting at 1), Y is either 1 or 2 (depending on whether the first team or the second team in the output scored points) and Z is the number of points scored. If no team scores in the end, write "End #X: no score".

Sample Input	Sample Output
4 3 0 -12 -14 14 -60 62 30 65 24 24 -20 -24 -34 20 4 3 -75 47 -58 63 47 75 58 68 -60 51 -5 48 62 43 -1 -1	End 1: team 1 scores 2 End 2: team 2 scores 2

Judging Data for Problem 4

In the data below, [blank line] indicates a line that contains no characters (other than the newline symbol) in the input.

```
0 0
[blank line]
[blank line]
1 0
0 0
[blank line]
0 1
[blank line]
0 0
1 1
0 0
0 12
1 1
0 80
81 0
2 1
0 0 -20 0
0 15
1 2
0 0
0 78 78 0
2 2
-10 0 10 0
0 12 24 0
2 2
20 0 31 31
6 0 15 15
4 1
0 1 10 11 22 -23 70 70
-71 -71
4 3
0 -12 -14 14 -60 62 30 65
24 24 -20 -24 -34 20
4 3
-75 47 -58 63 47 75 58 68
-60 51 -5 48 62 43
8 8
0 0 12 0 -12 0 0 -12 0 12 0 24 24 0 -24 0
36 0 0 36 -36 0 0 -36 36 36 -36 -36 36 -36 -36 36
-1 -1
```

Problem 5 – Driveway Melting

Your parents have given you the job of shoveling your driveway, but since it is spring, you know the snow will start to melt and you can avoid shoveling it. So instead of shoveling, you've decided to measure the depth of the snow on your driveway and calculate how much it will melt. You've determined two things:

1. the depth of the snow (in cm) decreases by one cm per time unit. The time unit depends on the temperature, but you've eliminated that by measuring the time in abstract "time units".
2. however, when snow is next to bare driveway, it melts faster because of the heat absorbed by the driveway surface. Snow next to bare driveway will melt an additional centimetre per time unit.

You've divided your driveway (a rectangle) into several square regions, each of identical size. Each region has a consistent depth of snow and you've measured the depth in cm. Because of your observation #2, any region that is next to a bare region (even diagonally) will melt at a rate of 2 cm per time unit. That is, if the center region in the diagram is bare, all of the striped regions will melt at a rate of 2 cm per time unit.

If a region with snow is next to several bare regions, the rate is increased by 1 cm of melting per time unit for each bare region.

As an example, if the snow on your driveway is represented by the grid on the left below, then after one time unit has elapsed, the snow depth would be given by the grid on the right below.

Before

4	2	1	1
4	3	0	1
4	5	3	1
4	3	0	1
5	5	5	5

After one time unit

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

Write a program that demonstrates the amount of snow that is left on the driveway after one time unit.

Input

The input consists of several cases, each given by the parameters of the case, and then the data for the driveway.

The parameters for each test case are two integers, *m n*, on a line. *m* and *n* are the dimensions of the driveway, in terms of the number of regions that have been measured. *m* and *n* will always be positive integers between 1 and 10.

The next *m* lines of the input file contain *n* integers, representing the depth of snow in centimeters. Each integer will be separated from the next by a single space.

The final line of input will be 0 0. It should not be processed.

Output

For each test case, output "CASE X" where X is the number of the case (starting with 1), followed by the snow in the driveway after the time steps. After each case, output "END CASE X".

Sample Input	Sample Output
5 4 4 2 1 1 4 3 0 1 4 5 3 1 4 3 0 1 5 5 5 5 0 0	CASE 1 3 0 0 0 3 1 0 0 3 2 0 0 3 1 0 0 4 3 3 3 END CASE 1

Judging Data for Problem 5

```
1 1
1
2 2
1 2
3 4
2 2
0 1
2 3
3 2
1 2
3 4
5 6
5 4
4 2 1 1
4 3 0 1
4 5 3 1
4 3 0 1
5 5 5 5
3 3
9 9 9
9 0 9
9 9 9
3 3
0 0 0
0 9 0
0 0 0
3 3
7 0 7
0 9 0
7 0 6
```

(data continues on next page)

```
10 10
17 22 12 17 22 6 15 20 11 18
17 10 14 9 21 9 1 19 22 22
5 3 10 15 15 25 12 13 17 0
14 1 20 19 1 9 17 12 11 22
1 2 5 0 3 13 19 5 24 23
18 7 21 11 1 0 23 18 6 17
2 7 18 10 15 14 3 5 24 1
19 23 22 18 2 15 3 7 13 4
19 11 11 9 22 5 19 24 7 0
18 13 9 9 8 13 19 8 13 14
10 10
47 31 51 39 88 68 24 4 76 23
61 39 79 92 26 20 92 4 88 20
92 66 89 80 2 44 67 9 7 83
39 41 12 22 86 20 68 59 90 26
4 97 31 21 68 65 71 0 39 83
83 27 25 93 73 27 81 58 45 4
31 56 73 64 3 40 6 13 10 81
52 64 32 20 17 73 43 16 36 21
72 66 49 49 96 51 26 3 36 11
38 19 25 66 49 31 50 89 9 7
0 0
```

Problem 6 – Naming Rivers

You are part of an exploratory mission to a new earth-like planet, and have been appointed as the hydrologist for the trip, despite the fact that your training is in computer science. How did you end up being the hydrologist? Let us never speak of that.

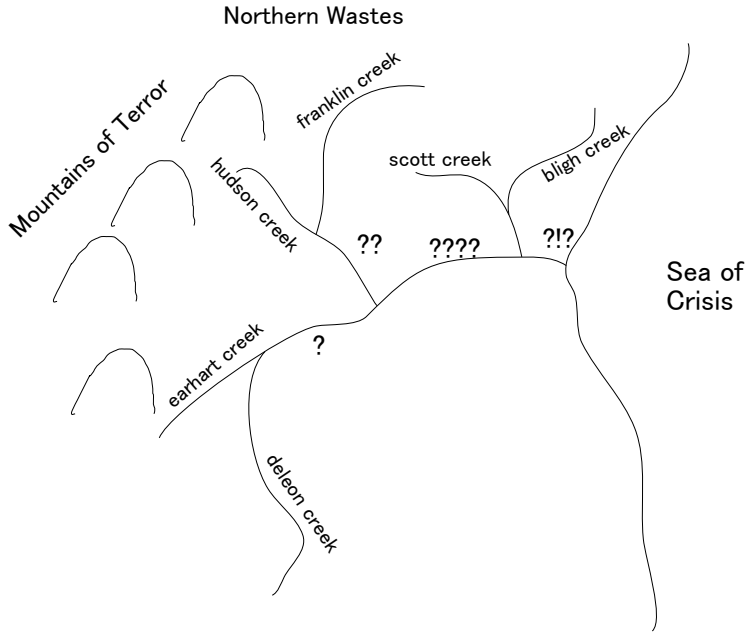
As hydrologist, you are responsible for naming all of the planet's rivers. Everyone has decided that each of the streams (rivers that have a beginning and no other rivers flowing into them) on the planet will be named after a crew member, but there have been significantly heated arguments about how rivers will be named when they join (which is called a *confluence*). The arguments are so intense that you're worried you will end up like the last hydrologist. But let's not talk about that.

As a computer scientist, you've decided to name the rivers based on Strahler number, which calculates the complexity of a river based on how many other rivers flow into it.

The Strahler number is defined for a river R as follows:

1. if R is a stream, it has Strahler number 1. Put another way, if no other rivers flow into R, then its Strahler number is 1.
2. if two rivers with Strahler number X join and become river R, then R has Strahler number X+1.
3. if two rivers with Strahler number X and Y (with X>Y) join and become river R, then R has Strahler number X.

For instance, you've found this rough map drawn by the previous hydrologist.



The Strahler number of each creek is 1. The river indicated by "?" in the map would have Strahler number 2 (both earhart creek and deleon creek have Strahler number 1). The river indicated by "????" would have Strahler number 3 (both rivers flowing into it will have Strahler number 2) and the river indicated by "?!?" would also have Strahler number 3.

To name rivers, if two rivers join, then the river with the higher Strahler number transfers its name to the new (joined) river. However, if the two rivers have the same Strahler number, then the new (joined) river is given the name of the two rivers that comes alphabetically first. (This rule is consistent, but Ensign Zzzzaran is clearly unhappy about it.) For example, in the map above, "??" should be replaced by "franklin creek" while "?!?" should be replaced with "deleon creek".

Write a program that takes the description of a river system and outputs the names of the rivers.

Input

The input has several test cases. Each test case has several lines. The first line is n ($n > 0$), the number of lines in the test case. The next n lines each start with an id number, corresponding to one of the confluences of two rivers **or** a stream.

If the line is a confluence, then the line has the format "ID1 ID2 ID3".
That is, after the first id number are two more id numbers, each describing another confluence or stream in the input. If the line is a stream, the line has the format "ID NAME". That is, after the id number is the name of the stream.

All lines in a test case appear in ascending order, starting from id number 1 up to n. No id numbers are missed. The names of streams do not contain spaces. All id numbers and names are separated by exactly one space.

The final line of the input is the number 0 on its own line.

Output

For each test case, start with a single line that says "CASE #X" where X is the case number, starting from 1. Then output one line per id number, of the form "ID NAME STRAHLER" where ID is an id number, starting from 1 and going up to n. NAME is either the name of a stream (if the id number corresponds to a stream) or the name of the river formed by the confluence (if the id number corresponds to a confluence). STRAHLER is the Strahler number of either the stream or the river formed by the confluence, depending on the case.

Sample Input	Sample Output
11 1 2 3 2 4 5 3 6 9 4 bligh 5 scott 6 7 8 7 franklin 8 hudson 9 10 11 10 deleon 11 earhart 0	Case #1 1 deleon 3 2 bligh 2 3 deleon 3 4 bligh 1 5 scott 1 6 franklin 2 7 franklin 1 8 hudson 1 9 deleon 2 10 deleon 1 11 earhart 1

Judging Data for Problem 6

```
1
1 Winnipeg
3
1 2 3
2 Winnipeg
3 Whiteshell
5
1 2 3
2 Winnipeg
3 4 5
4 Whiteshell
5 Souris
7
1 2 5
2 3 4
3 Winnipeg
4 Souris
5 6 7
6 Rouseau
7 Pembina
7
1 2 3
2 Assinniboine
3 4 5
4 Berens
5 6 7
6 Churchill
7 Fisher
11
1 2 3
2 4 5
3 6 7
4 Assinniboine
5 8 9
6 Berens
7 10 11
```

(data continues on next page)

```
8 Churchill
9 Fisher
10 Grass
11 Hayes
15
1 2 3
2 4 5
3 6 7
4 Antler
5 8 9
6 Bird
7 10 11
8 12 13
9 Caribou
10 Fox
11 14 15
12 Rat
13 Seal
14 Turtle
15 Swan
17
1 2 3
2 4 5
3 Antler
4 6 7
5 8 9
6 Berens
7 Carrot
8 Dauphin
9 10 11
10 Echimamish
11 12 13
12 14 15
13 Fox
14 Grass
15 16 17
16 Hayes
17 Keewatin
```
(data continues on next page)

199
1 2 3
2 a
3 4 5
4 aa
5 6 7
6 aaa
7 8 9
8 aaaa
9 10 11
10 aaaaa
11 12 13
12 aaaaaa
13 14 15
14 aaaaaaa
15 16 17
16 aaaaaaaa
17 18 19
18 aaaaaaaaa
19 20 21
20 aaaaaaaaaa
21 22 23
22 aaaaaaaaaaa
23 24 25
24 aaaaaaaaaaaa
[lines deleted of the form
    2n-1 2n 2n+1
    2n a^n
 for n between 13 and 98]
195 196 197
196
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
197 198 199
198
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
199 b
0