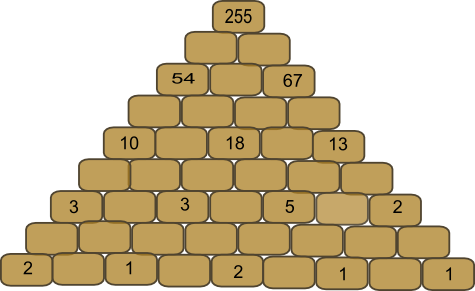
|  |
| --- |
| Add bricks in the wall |

This in not ``*another brick in the wall*", it's just a matter of adding numbers. Suppose you have a wall with the shape of a triangle, like the one shown below. The wall has 9 rows and row ***i*** has exactly ***i*** bricks, considering that top row is the first one and bottom row is the ninth. Some bricks are labeled with a number and other ones are blank. Notice that labeled bricks appear only on odd rows and they occupy odd positions within the row. The problem you must solve is finding a suitable number for each blank brick taking into account one simple rule: the number of a brick is obtained by adding the numbers of the two bricks below it. Obviously, this rule does not apply to the ninth row. Numbers are supposed to be integers.



**Input**

The first line of the input contains an integer ***N***, indicating the number of test cases. This line is followed by the lines corresponding to the test cases. Each test case is described in five lines. These five lines correspond to odd rows of the wall, from top to bottom, as described above. Line ***i*** contains the numbers corresponding to odd bricks on row ***i*** of the wall (that is, non blank bricks), enumerated from left to right and separated with a single space. It is supposed that each test case is correct, that is, there exists a solution to the problem that the case describes.

**Output**

For each test case, the output should consist of nine lines describing the numbers of all bricks of the wall. So, line ***i*** should contain the numbers corresponding to the ***i*** bricks on row ***i*** of the wall, enumerated from left to right and separated by a single space.

**Note:** Here we have an example with two test cases. The first one corresponds to the wall depicted above.

**Sample Input**

2

255

54 67

10 18 13

3 3 5 2

2 1 2 1 1

256

64 64

16 16 16

4 4 4 4

1 1 1 1 1

**Sample Output**

255

121 134

54 67 67

23 31 36 31

10 13 18 18 13

5 5 8 10 8 5

3 2 3 5 5 3 2

2 1 1 2 3 2 1 1

2 0 1 0 2 1 1 0 1

256

128 128

64 64 64

32 32 32 32

16 16 16 16 16

8 8 8 8 8 8

4 4 4 4 4 4 4

2 2 2 2 2 2 2 2

1 1 1 1 1 1 1 1 1

6

255

54 67

10 18 13

3 3 5 2

2 1 2 1 1

256

64 64

16 16 16

4 4 4 4

1 1 1 1 1

512

512 0

512 0 0

512 0 0 0

512 0 0 0 0

6561

729 2916

81 324 1296

9 36 144 576

1 4 16 64 256

2274

525 525

81 168 81

9 36 36 9

1 4 16 4 1

678

195 195

81 30 81

36 9 9 36

16 4 1 4 16

255↵\r\n

121 134↵\r\n

54 67 67↵\r\n

23 31 36 31↵\r\n

10 13 18 18 13↵\r\n

5 5 8 10 8 5↵\r\n

3 2 3 5 5 3 2↵\r\n

2 1 1 2 3 2 1 1↵\r\n

2 0 1 0 2 1 1 0 1↵\r\n

256↵\r\n

128 128↵\r\n

64 64 64↵\r\n

32 32 32 32↵\r\n

16 16 16 16 16↵\r\n

8 8 8 8 8 8↵\r\n

4 4 4 4 4 4 4↵\r\n

2 2 2 2 2 2 2 2↵\r\n

1 1 1 1 1 1 1 1 1↵\r\n

512↵\r\n

512 0↵\r\n

512 0 0↵\r\n

512 0 0 0↵\r\n

512 0 0 0 0↵\r\n

512 0 0 0 0 0↵\r\n

512 0 0 0 0 0 0↵\r\n

512 0 0 0 0 0 0 0↵\r\n

512 0 0 0 0 0 0 0 0↵\r\n

6561↵\r\n

2187 4374↵\r\n

729 1458 2916↵\r\n

243 486 972 1944↵\r\n

81 162 324 648 1296↵\r\n

27 54 108 216 432 864↵\r\n

9 18 36 72 144 288 576↵\r\n

3 6 12 24 48 96 192 384↵\r\n

1 2 4 8 16 32 64 128 256↵\r\n

2274↵\r\n

1137 1137↵\r\n

525 612 525↵\r\n

219 306 306 219↵\r\n

81 138 168 138 81↵\r\n

27 54 84 84 54 27↵\r\n

9 18 36 48 36 18 9↵\r\n

3 6 12 24 24 12 6 3↵\r\n

1 2 4 8 16 8 4 2 1↵\r\n

678↵\r\n

339 339↵\r\n

195 144 195↵\r\n

123 72 72 123↵\r\n

81 42 30 42 81↵\r\n

54 27 15 15 27 54↵\r\n

36 18 9 6 9 18 36↵\r\n

24 12 6 3 3 6 12 24↵\r\n

16 8 4 2 1 2 4 8 16↵\r\n