There are *n* zombies in Seattle, and Liv and Ravi are trying to track them down to find out who is creating new zombies — thus preventing an apocalypse. Other than the patient-zero zombies (who became so by mixing MaxRager and tainted Utopium), new people only become zombies after being scratched by an existing zombie; for this reason, zombiism is transitive. This means that if zombie *0* knows zombie *1* and zombie *1* knows zombie *2*, then zombie *0* is connected to zombie *2*. A zombie *cluster* is a group of zombies who are directly or indirectly linked through the other zombies they know (such as the one who scratched them or supplies them with brains).

Complete the *zombieCluster* function in your editor. It has *1* parameter: an array of binary strings (i.e., composed of *0*s and *1*s) named *zombies* that describes an *n × n* matrix of known connected zombies; if *zombies[i][j] = 0*, then the *ith* and *jth* zombies *do not* know one another (otherwise, the cell contains a *1* and they *do* know one another). Your function must return an integer denoting the number of zombie clusters Liv and Ravi have identified in Seattle.

**Note:** Method signatures may vary depending on the requirements of your chosen language.

**Input Format**

The locked stub code in your editor reads the following input from stdin and passes it to your function:

The first line contains an integer, *n*, describing the base size of your zombie association matrix. Each of the *n* subsequent lines contains a binary string of length *n* describing a row in the matrix.

**Constraints**

* *1 ≤ n ≤ 300*
* *0 ≤ i < n*
* *|zombies| = n*
* Each *zombies[i]* contains a binary string of *n* zeroes and ones.
* *zombies[i][i] = 1*, where *0 ≤ i < n*.
* *zombies[i][j] = zombies[j][i]*, where *0 ≤ i < j < n*.

**Output Format**

Your function must return a single integer denoting the number of different zombie clusters in Seattle. This is printed to stdout by the locked stub code in your editor.

**Sample Input 0**

4

1100

1110

0110

0001

**Sample Output 0**

2

**Sample Input 1**

5

10000

01000

00100

00010

00001

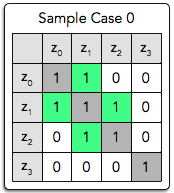
**Sample Output 1**

5

**Explanation**

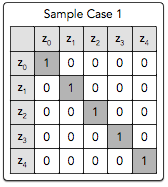
In the diagrams below, the squares highlighting a known connection between two different zombies are highlighted in green. Because each zombie is already aware that they are personally a zombie, those are highlighted in grey.

*Sample Case 0:*



We have *n = 4* zombies numbered *Z0* through *Z3*. There are *2* pairs of zombies who directly know each another: *(Z0*, *Z1)* and *(Z1*, *Z2)*. Because of zombiism's transitive property, the set of zombies *{Z0, Z1, Z2}* is considered to be a single zombie cluster. The remaining zombie, *Z3*, doesn't know any other zombies and is considered to be his own, separate zombie cluster (*{Z3}*). This gives us a total of *2* zombie clusters, so we print *2* on a new line.

*Sample Case 1:*



No zombie knows who any other zombie is, so they each form their own zombie clusters: *{Z0}, {Z1}, {Z2}, {Z3}, and {Z4}*. This means we have *5* zombie clusters, so we print *5* on a new line.