2D

Problem Statement

You are given an $n \times n$ matrix, where $3 \leq n \leq 100$ and all entries are either 0 or 1. Is there a path from the top edge to the bottom edge? Write a C++ program that prints "1" if yes, and "0" if no.

Definitions and Illustrations

Top and Bottom Edge

The "Top" edge is the first row of inputs in the $n \times n$ matrix, and the "Bottom" edge is the last.

Connected

Two elements are connected by 1's if there is an uninterrupted sequence (or path) of neighboring 1's from one element to the other.

Input

The input will come from a text file named "data.txt". This file should come from the same folder or directory that your code is found in. The start of the file will have one number representing the linear dimension, n, followed by a newline. Each row will have 1's and 0's separated by a single space and end with a newline character.

Below is an example of a 10×10 matrix input:

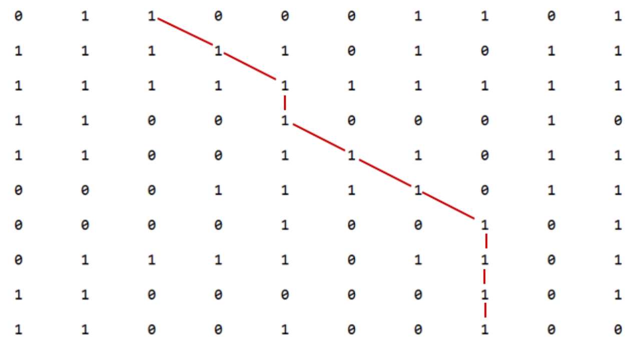
```
10
0 1 1 0 0 0 1 1 0 1
1 1 1 1 1 0 1 0 1 1
1 1 1 1 1 1 1 1 1 1
1 1 0 0 1 0 0 0 1 0
1 1 0 0 1 1 1 0 1 1
0 0 0 1 1 1 1 0 1 1
0 0 0 0 1 0 0 1 0 1
0 1 1 1 1 0 1 1 0 1
1 1 0 0 0 0 0 1 0 1
1 1 0 0 1 0 0 1 0 0
```

Output

This problem should have the following output:

```
1
```

Below is an example of a path that moves from the top edge to the bottom:



```
0    1    1    0    0    0    1    1    0    1
1    1    1    1    1    0    1    0    1    1
1    1    1    1    1    1    1    1    1    1
1    1    0    0    1    0    0    0    1    0
1    1    0    0    1    1    1    0    1    1
0    0    0    1    1    1    1    0    1    1
0    0    0    0    1    0    0    1    0    1
0    1    1    1    1    0    1    1    0    1
1    1    0    0    0    0    0    1    0    1
1    1    0    0    1    0    0    1    0    0
```

3D Problem

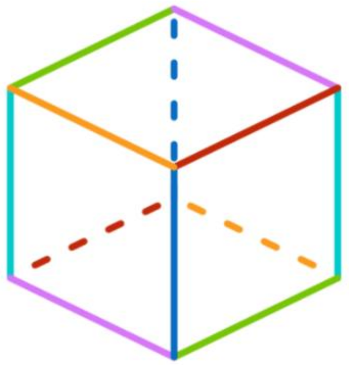
Problem Statement

You are given an $n \times n \times n$ matrix that represents a cube, where $3 \leq n \leq 100$ and all entries are either 0 or 1. How many pairs of opposite edges of the cube are connected by a path of 1's? Write a C++ program that returns the number of connected opposite edge pairs.

Definitions and Illustrations

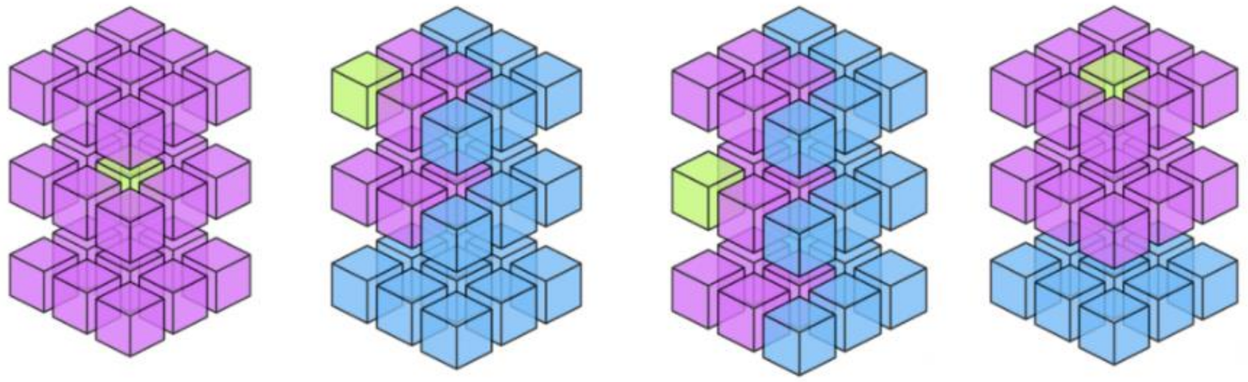
Opposite Edge

A cube has 12 edges. Opposite edges are two parallel edges that are positioned diagonally accross from one another as shown in the picture below. There are a total of 6 opposite edge pairs.



Connected

An element in the middle of a cube has at most 26 neighbors (counting all adjacent elements across, beside, above, below, and diagonal). An element in the corner has 7 neighbors, an element on the edge will have no more than 11 neighbors, and an element on a face will have no more than 17 neighbors.



Two elements are connected by 1's if there is an uninterrupted sequence (or path) of neighboring 1's from one element to the other.

Two edges of the $3D$ matrix/cube are connected if there is at least one element from one edge that connects to an element on the opposite edge.

Since a cube has 6 opposite edge pairs, the solution will be a number from 0 to 6.

Input

The input will come from a text file named "data.txt". This file should come from the same folder or directory that your code is found in. The start of the file will have one number representing the linear dimension, n, followed by a newline. Each row will have 1's and 0's separated by a single space and end with a newline character. Each 2D matrix will be delimited by the # symbol.

Below is an example of a 3×3 matrix input:

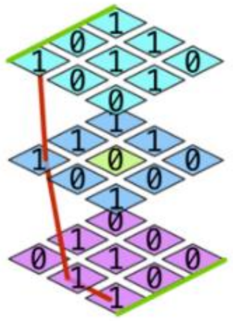
```
3
1 0 1
0 1 1
0 1 0
#
1 1 1
0 0 1
1 0 0
#
0 1 0
1 1 0
1 0 0
```

Output

This problem should have the following output:

4

One connected edge pair of above example:


1
0
1
1
1
0
0
1
0
1
1
1
1
0
0
0
1
0
1
0
0
1
0
0
1
0
1