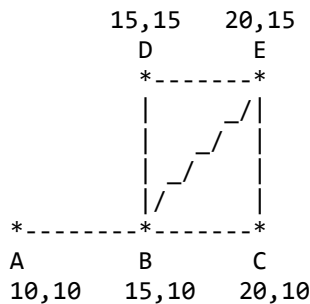# Cow Tours

Farmer John has a number of pastures on his farm. Cow paths connect some pastures with certain other pastures, forming a field. But, at the present time, you can find at least two pastures that cannot be connected by any sequence of cow paths, thus partitioning Farmer John's farm into multiple fields.

Farmer John would like add a single a cow path between one pair of pastures using the constraints below.

A field's `diameter' is defined to be the largest distance of all the shortest walks between any pair of pastures in the field. Consider the field below with five pastures, located at the points shown, and cow paths marked by lines:

```
        15,15    20,15
          D        E
          *-------*
          |      _/|
          |    _/  |
          |  _/    |
          |/       |
  *-------*--------*
  A       B        C
 10,10   15,10    20,10
```

The `diameter' of this field is approximately 12.07106, since the longest of the set of shortest paths between pairs of pastures is the path from A to E (which includes the point set {A,B,E}). No other pair of pastures in this field is farther apart when connected by an optimal sequence of cow paths.

Suppose an additional field on the same plane as above is connected by cow paths as follows:

```
           *F 30,15
           /
         _/
       _/
      /
    *------*
    G      H
  25,10   30,10
```
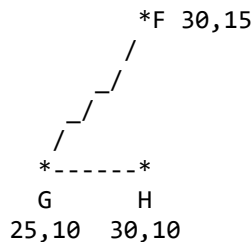
Given just these two fields on his farm, Farmer John would add a cow path between a point in each of these two fields (namely point sets {A,B,C,D,E} and {F,G,H}) so that the joined set of pastures {A,B,C,D,E,F,G,H} has the smallest possible diameter.

Note that cow paths do not connect just because they cross each other; they only connect at listed points.

The input contains the pastures, their locations, and a symmetric "adjacency" matrix that tells whether pastures are connected by cow paths. Pastures are not considered to be connected to themselves. Here's one annotated adjacency list for the pasture {A,B,C,D,E,F,G,H} as shown above:

```
        A B C D E F G H
      A 0 1 0 0 0 0 0 0
      B 1 0 1 1 1 0 0 0
      C 0 1 0 0 1 0 0 0
      D 0 1 0 0 1 0 0 0
      E 0 1 1 1 0 0 0 0
      F 0 0 0 0 0 0 1 0
      G 0 0 0 0 0 1 0 1
      H 0 0 0 0 0 0 1 0
```

Other equivalent adjacency lists might permute the rows and columns by using some order other than alphabetical to show the point connections. The input data contains no names for the points.

The input will contain at least two pastures that are not connected by any sequence of cow paths.

Find a way to connect exactly two pastures in the input with a cow path so that the new combined field has the smallest possible diameter of any possible pair of connected pastures. Output that smallest possible diameter.

PROGRAM NAME: cowtour

INPUT FORMAT

Line 1:	An integer, N (1 <= N <= 150), the number of pastures
Line 2-N+1:	Two integers, X and Y (0 <= X ,Y<= 100000), that denote that X,Y grid location of the pastures; all input pastures are unique.
Line N+2-2*N+1:	lines, each containing N digits (0 or 1) that represent the adjacency matrix as described above, where the rows' and columns' indices are in order of the points just listed.

SAMPLE INPUT (file cowtour.in)

```
8
10 10
15 10
20 10
15 15
20 15
30 15
25 10
30 10
01000000
10111000
01001000
01001000
01110000
00000010
00000101
00000010
```

OUTPUT FORMAT

The output consists of a single line with the diameter of the newly joined pastures. Print the answer to exactly six decimal places. Do not perform any special rounding on your output.

SAMPLE OUTPUT (file cowtour.out)