Input File: –
Output File: –
Source File: `square.pas/.c/.cpp`

100 Points
Time limit: 1 s
Memory limit: 16 MB

Solution

The following algorithm does not need more than $3 \cdot N$ queries and is thus within the limits.

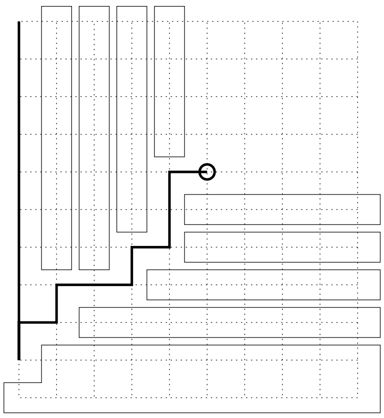
First, we have to understand the information that each path from $(1,1)$ to a given node v has the same length. We will denote $D(v)$ as this distance from $(1,1)$ to v. If $D(v) > L$, we know that neither the nodes to the right of v nor the nodes below v can be a solution, as their distances from $(1,1)$ are even greater.

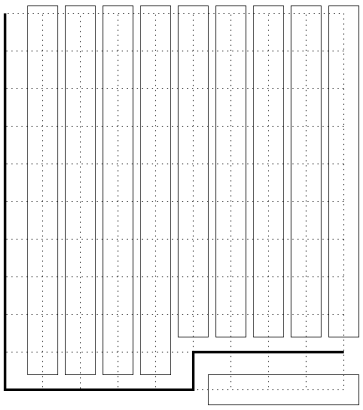
We begin by exploring the graph at node $(1,1)$ and move down until we reach a node v whose distance is $D(v) \geq L$. If $D(v) = L$, we have found a solution. If $D(v) > L$ we know that neither the current node nor any of the nodes to the right of the current node or below the current node can be a solution. Furthermore we know that the node above had a too short distance. So we reach the next node that we have to examine by moving up and right.

Now we can iteratively proceed to the right neighbor every time the distance of the current node is too small or to the upper neighbor every time the distance is too long. Each time we can subtract a group of nodes from the set of possible solutions, as you can see in the figure below.

If we have to go in a direction where there are no further nodes, we are sure that there is no solution for this testcase.

We have forgotten one case at the very beginning: What happens if we cannot reach a node of distance $\geq L$ by moving down? – The answer is easy: We continue by moving to the right and proceed as described above as soon as we have found a node v with $D(v) > L$.





It is obvious that you do not need more than $3 \cdot N$ queries: You will move at most $N-1$ steps down, $N-1$ steps to the right and $N-1$ steps up.