Search in a row wise and column wise sorted matrix

Given an n x n matrix, where every row and column is sorted in increasing order. Given a number x, how to decide whether this x is in the matrix. The designed algorithm should have linear time complexity.

We strongly recommend that you click here and practice it, before moving on to the solution.

Thanks to devendraiiit for suggesting below approach.

1) Start with top right element
2) Loop: compare this element e with x
....i) if they are equal then return its position
...ii) e < x then move it to down (if out of bound of matrix then break return false) ..iii) e > x then move it to left (i
out of bound of matrix then break return false)
3) repeat the i), ii) and iii) till you find element or returned false

Implementation:

```
#include<stdio.h>

/* Searches the element x in mat[][]. If the element is found,
    then prints its position and returns true, otherwise prints
    "not found" and returns false */
int search(int mat[4][4], int n, int x)
{
   int i = 0, j = n-1;  //set indexes for top right element
   while ( i < n && j >= 0 )
   {
      if ( mat[i][j] == x )
      {
         printf("\n Found at %d, %d", i, j);
         return 1;
      }
      if ( mat[i][j] > x )
        j--;
      else //  if mat[i][j] < x
        i++;
   }

   printf("\n Element not found");
   return 0;  // if ( i==n || j== -1 )
}

// driver program to test above function
int main()
{
  int mat[4][4] = { {10, 20, 30, 40},
                    {15, 25, 35, 45},
                    {27, 29, 37, 48},
                    {32, 33, 39, 50},
                  };
  search(mat, 4, 29);
  getchar();
  return 0;
}
```

Run on IDE

Time Complexity: O(n)

The above approach will also work for m x n matrix (not only for n x n). Complexity would be O(m + n).