192 (sorted two-dimensional search) Write a program to find a given item in a given 2-dimensional array in which each row is sorted and each column is sorted. The execution time must be linear in the sum of the dimensions.

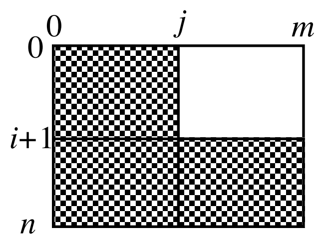
After trying the question, scroll down to the solution.

§ Let the array be A , let its dimensions be n by m , and let the item we seek be x . If our search eliminates items one by one, the time will be $n{\times}m$. For execution time $n{+}m$ we have to eliminate a whole row or column each time we look at an item. Since the rows and columns are sorted, we can do that by looking at an end of a row or column. If we look at the top left corner $A\ 0\ 0$, and we see $x < A\ 0\ 0$, we can eliminate both column 0 and row 0 and, in fact, the whole array; but if $x > A\ 0\ 0$, we cannot eliminate anything but the one item $A\ 0\ 0$. The bottom right corner has the same problem. So let's look at the bottom left corner. If we see $x > A\ (n{-}1)\ 0$, we can eliminate column 0 ; if we see $x < A\ (n{-}1)\ 0$, we can eliminate row $n{-}1$. Then the problem is the same, but one row or column smaller. We could equally well start at the top right corner, but let's start at the bottom left. We'll need integer variables i and j to keep track of the row and column; they start at $i{=}n{-}1 \land j{=}0$, and they finish at or before $i'{=}0 \lor j'{=}m{-}1$ (note: $\lor$). Now we need some way to indicate x is not found, so that can be either i gets too small or j gets too big: $i'{=}{-}1 \lor j'{=}m$.

The problem, except for time, is P , where

$$P \;=\; \mathbf{if}\ x\colon A(0,..n)(0,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi}$$

Now we need to describe the remaining problem Q when we are partway through searching. A picture helps: the remaining search area is the clear part, and $A\ i\ j$ is its bottom left corner.



Define

$$Q \;=\; \mathbf{if}\ x\colon A(0,..i{+}1)(j,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi}$$

Then

$$\begin{aligned}
P \;&\Leftarrow\; i{:=}\ n{-}1.\ j{:=}\ 0.\ Q\\
Q \;&\Leftarrow\; \mathbf{if}\ i{=}{-}1 \lor j{=}m\ \mathbf{then}\ ok\\
&\qquad \mathbf{else\,if}\ A\ i\ j > x\ \mathbf{then}\ i{:=}\ i{-}1.\ Q\\
&\qquad\quad \mathbf{else\ if}\ A\ i\ j < x\ \mathbf{then}\ j{:=}\ j{+}1.\ Q\\
&\qquad\qquad \mathbf{else}\ ok\ \mathbf{fi\ fi\ fi}
\end{aligned}$$

Here is the proof. First the refinement of P .

$$\begin{aligned}
& i{:=}\ n{-}1.\ j{:=}\ 0.\ Q && \text{expand } Q\text{; substitution law twice}\\
=\;& \mathbf{if}\ x\colon A(0,..n)(0,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi}\\
=\;& P
\end{aligned}$$

Now the refinement of Q . We use case analysis.

$$\begin{aligned}
& Q \;\Leftarrow\; (i{=}{-}1 \lor j{=}m) \land ok && \text{expand } Q\text{, mirror}\\
=\;& \quad (i{=}{-}1 \lor j{=}m) \land ok\\
& \Rightarrow\ \mathbf{if}\ x\colon A(0,..i{+}1)(j,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi} && \text{distribution, antidist}\\
=\;& \quad (i{=}{-}1 \land ok\ \Rightarrow\ \mathbf{if}\ x\colon A(0,..i{+}1)(j,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi})\\
& \land\ (j{=}m \land ok\ \Rightarrow\ \mathbf{if}\ x\colon A(0,..i{+}1)(j,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi}) && \text{expand } ok\\
=\;& \quad (i{=}{-}1 \land i'{=}i \land j'{=}j\ \Rightarrow\ \mathbf{if}\ x\colon A(0,..i{+}1)(j,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi})\\
& \land\ (j{=}m \land i'{=}i \land j'{=}j\ \Rightarrow\ \mathbf{if}\ x\colon A(0,..i{+}1)(j,..m)\ \mathbf{then}\ x = A\ i'\ j'\ \mathbf{else}\ i'{=}{-}1 \lor j'{=}m\ \mathbf{fi})\\
& && \text{antecedent context}
\end{aligned}$$

$=$ $\quad (i{=}{-1} \wedge i'{=}i \wedge j'{=}j \;\Rightarrow\; \textbf{if } x{:}\ A(0,..0)(j,..m) \textbf{ then } x = A\ i'j' \textbf{ else } {-1}{=}{-1} \vee j'{=}m \textbf{ fi})$

$\wedge\ (j{=}m \wedge i'{=}i \wedge j'{=}j \;\Rightarrow\; \textbf{if } x{:}\ A(0,..i{+}1)(m,..m) \textbf{ then } x = A\ i'j' \textbf{ else } i'{=}{-1} \vee m{=}m \textbf{ fi})$

Each **if** condition is $\bot$ because the bunch is *null*, and the **else**-part is $\top$.

$=$ $\top$

Next case:

$Q \;\Leftarrow\; i{\ne}{-1} \wedge j{\ne}m \wedge A\ i\ j > x \wedge (i{:}{=}\ i{-}1.\ Q)$ — expand first Q, mirror

$=$ $\quad i{\ne}{-1} \wedge j{\ne}m \wedge A\ i\ j > x \wedge (i{:}{=}\ i{-}1.\ Q)$

$\Rightarrow\ \textbf{if } x{:}\ A(0,..i{+}1)(j,..m) \textbf{ then } x = A\ i'j' \textbf{ else } i'{=}{-1} \vee j'{=}m \textbf{ fi}$ — expand Q, substitution

$=$ $\quad i{\ne}{-1} \wedge j{\ne}m \wedge A\ i\ j > x \wedge \textbf{ if } x{:}\ A(0,..i)(j,..m) \textbf{ then } x = A\ i'j' \textbf{ else } i'{=}{-1} \vee j'{=}m \textbf{ fi}$

$\Rightarrow\ \textbf{if } x{:}\ A(0,..i{+}1)(j,..m) \textbf{ then } x = A\ i'j' \textbf{ else } i'{=}{-1} \vee j'{=}m \textbf{ fi}$ — If $A\ i\ j > x$ and row i is sorted, then $\neg\ x{:}\ A\ i(j,..m)$ and so $x{:}\ A(0,..i)(j,..m) = x{:}\ A(0,..i{+}1)(j,..m)$

$=$ $\top$

Next case:

$Q \;\Leftarrow\; i{\ne}{-1} \wedge j{\ne}m \wedge A\ i\ j < x \wedge (j{:}{=}\ j{+}1.\ Q)$ — just like the previous case

$=$ $\top$

Last case:

$Q \;\Leftarrow\; i{\ne}{-1} \wedge j{\ne}m \wedge A\ i\ j = x \wedge ok$ — expand Q, mirror

$=$ $\quad i{\ne}{-1} \wedge j{\ne}m \wedge A\ i\ j = x \wedge ok$

$\Rightarrow\ \textbf{if } x{:}\ A(0,..i{+}1)(j,..m) \textbf{ then } x = A\ i'j' \textbf{ else } i'{=}{-1} \vee j'{=}m \textbf{ fi}$ — expand ok

$=$ $\quad A\ i\ j = x \wedge i'{=}i \wedge j'{=}j$

$\Rightarrow\ \textbf{if } x{:}\ A(0,..i{+}1)(j,..m) \textbf{ then } x = A\ i'j' \textbf{ else } i'{=}{-1} \vee j'{=}m \textbf{ fi}$ — context $A\ i\ j = x$ makes **if** condition $\top$, and context $A\ i\ j = x \wedge i'{=}i \wedge j'{=}j$ makes **then** part $\top$.

$=$ $\top$

The timing proof is much easier. P becomes $t' \le t{+}n{+}m$ and Q becomes

$$-1{\le}i{<}n \wedge 0{\le}j{\le}m \Rightarrow t' \le t{+}i{+}1{+}m{-}j$$