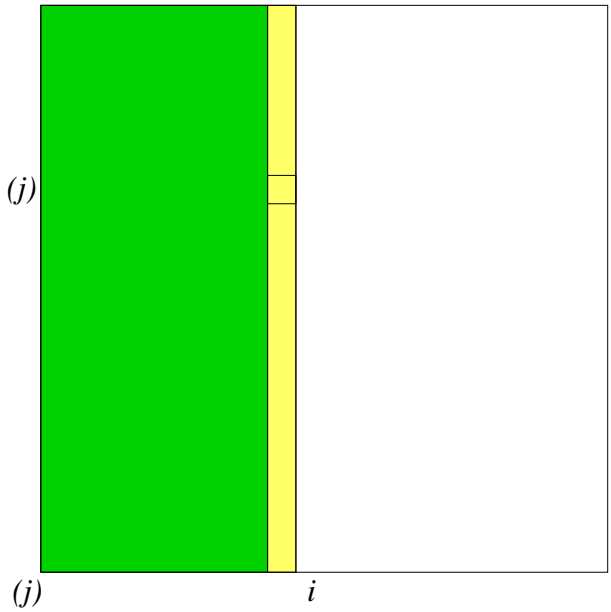
```python
def mat_find1(M, x):
    i = 0
    found = False
    while not found and i < len(M):
        j = 0
        while not found and j < len(M[i]) :
            found = M[i][j] == x
            j += 1
        i += 1
    if found :
        return (i-1, j-1)
    else :
        return None
```



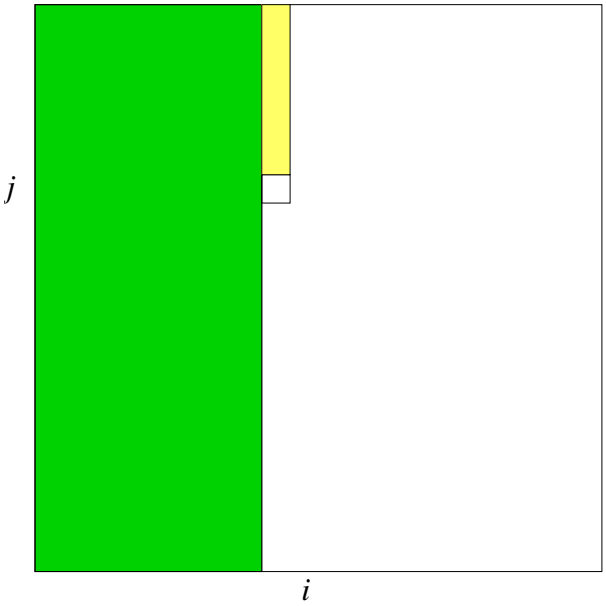
Invariant 1 (Outer loop of `mat_find1`)

(a). $\forall\ a \in [0, i-1), \forall\ b \in [0, m), \texttt{M}[a][b] \neq x$

(b). $\sim$ `found` *iff* $\forall\ b \in [0, m), \texttt{M}[i-1][b] \neq x$

(c). `found` *iff* $\texttt{M}[i-1][j-1] = x$

(d). i is the number of iterations of the outer loop completed.



Invariant 2 (Inner loop of `mat_find1`)

(a). $\forall\ b \in [0, j-1), \texttt{M}[i][b] \neq x$

(b). `found` *iff* $\texttt{M}[i][j-1] = x$

(c). j is the number of iterations of the inner loop completed on the current iteration of the outer loop.

In the worse case, each position in the array is read once, hence $\Theta(m^2)$ or $\Theta(n)$.

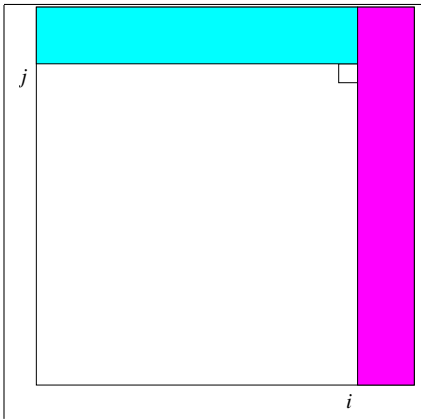
```python
def mat_find2(M, x):
    i = len(M) - 1
    j = 0
    found = False
    while not found and i >= 0  and j < len(M[i]):
        while i >= 0 and M[i][j] > x :
            i -= 1
        while i >= 0 and j < len(M[i]) and M[i][j] < x :
            j += 1
        if i >= 0 and j < len(M[i]) :
            found = M[i][j] == x

    if found :
        return (i, j)
    else :
        return None
```

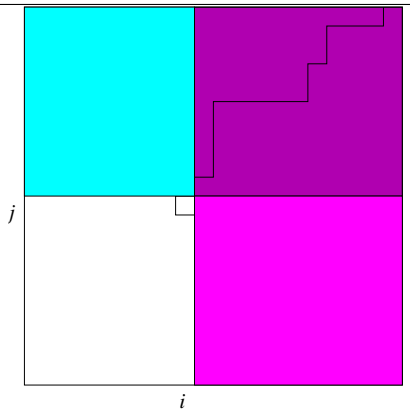
1	2	8	21	43	57	92	103
3	5	9	23	44	61	93	105
17	22	27	30	46	62	95	106
37	39	42	47	48	69	99	108
64	67	71	75	76	77	101	110
73	74	81	88	89	91	107	119
92	96	100	102	103	106	111	121
115	116	126	131	138	146	152	160

```
def mat_find2(M, x):
    i = len(M) - 1
    j = 0
    found = False
    while not found and i >= 0  and j < len(M[i]):
        while i >= 0 and M[i][j] > x :
            i -= 1
        while i >= 0 and j < len(M[i]) and M[i][j] < x :
            j += 1
        if i >= 0 and j < len(M[i]) :
            found = M[i][j] == x

    if found :
        return (i, j)
    else :
        return None
```



After first iteration



Arbitrary case

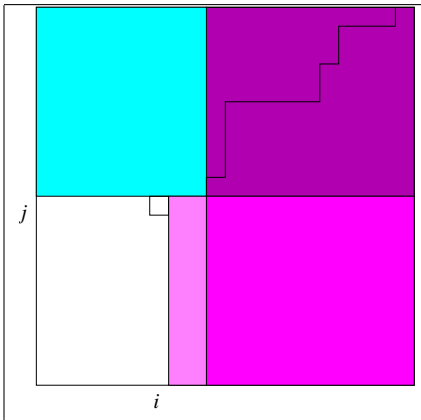
Invariant 3 (Outer loop of `mat_find2`)

(a). $\forall\ a \in (i, m), \forall\ b \in [j, m), \texttt{M}[a][b] > x$

(b). $\forall\ a \in [0, i), \forall\ b \in [0, j), \texttt{M}[a][b] < x$

(c). $\forall\ a \in (i, m), \forall\ b \in [0, j), \texttt{M}[a][b] \neq x$

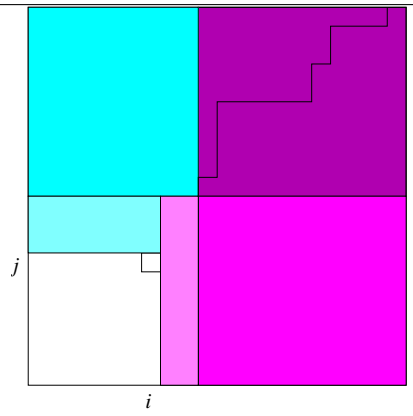
(d). `found` *iff* $\texttt{M}[i][j] = x$



Invariant 4 (First inner loop of `mat_find2`)

(a). $\forall\ a \in (i, m), \forall\ b \in (j, m), \texttt{M}[a][b] > x$

(b). $m - i - 1$ *is the* ***total*** *number of iterations of the first inner loop completed (across all iterations of the outer loop)*



Invariant 5 (Second inner loop of `mat_find2`)

(a). $\forall\ a \in [0, i], \forall\ b \in [0, j), \texttt{M}[a][b] < x$

(b). j *is the* ***total*** *number of iterations of the second inner loop completed (across all iterations of the outer loop)*

On any iteration of the outer loop, at least one of the inner loops must have at least one iteration, or else we have found the item at position (i, j). Thus the number of iterations of the outer loop is less than or equal to the sum of the total number of iterations of the inner loops plus one. Each inner loop will have at most m **total** iterations. Hence worst case $\Theta(m)$ or $\Theta(\sqrt{n})$.