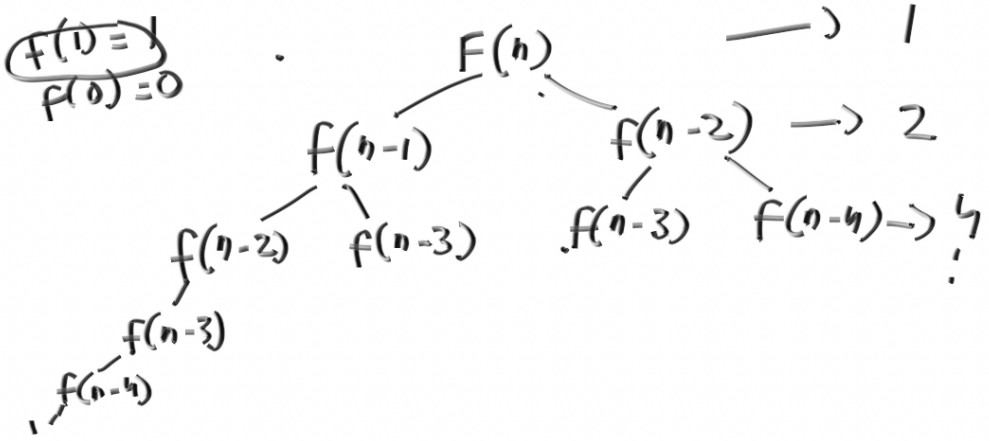
$$F(n) = F(n-1) + F(n-2)$$

$f(1) = 1$
$f(0) = 0$



$n = 1000$

(3)

Big-O

Theta $\Theta(n)$ n^2 $n^{\log n}$

Ω

O

$1 < \log n < n < n\log n < n^2 \text{ ---}$

$\Omega(n)$ $O(n^2)$

2^{2n} 2^n

4^n 2^n

$2^n \times 2^n \times 2^n$ ---- n times

2^{n+1} 2^{2n+1}

2×2^n 2×2^{2n}

n^2 n^3 $2n$ $3n$

$n^2 \times n$

n n^2

$O(f(n) + g(n)) = O(\max(f(n), g(n))$

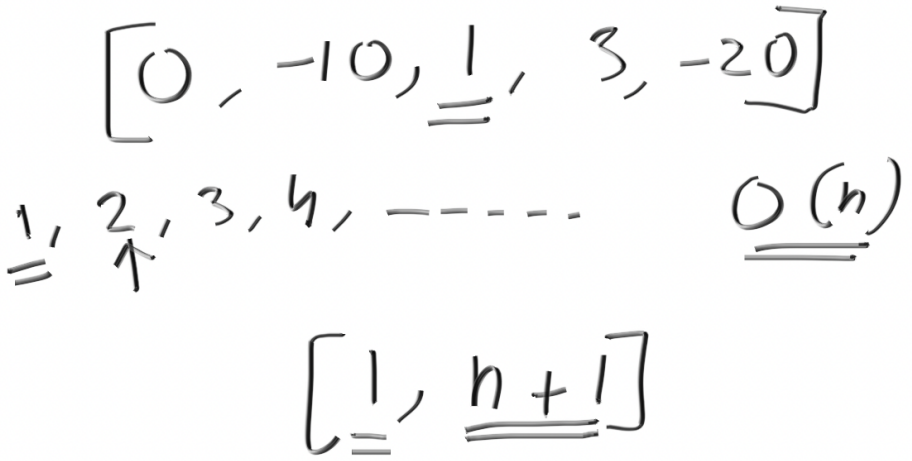
$O(n + n^2) = O(n^2)$

Example 1:

```
Input:
N = 5
arr[] = {1,2,3,4,5}
Output: 6
Explanation: Smallest positive missing
number is 6.
```

Example 2:

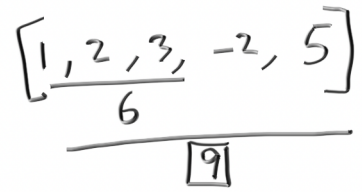
```
Input:
N = 5
arr[] = {0,-10,1,3,-20}
Output: 2
Explanation: Smallest positive missing
number is 2.
```

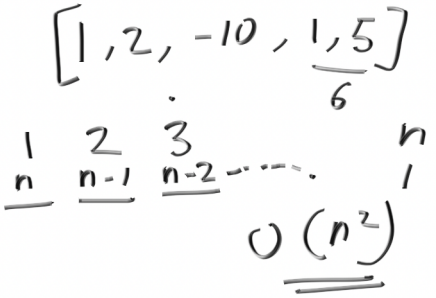


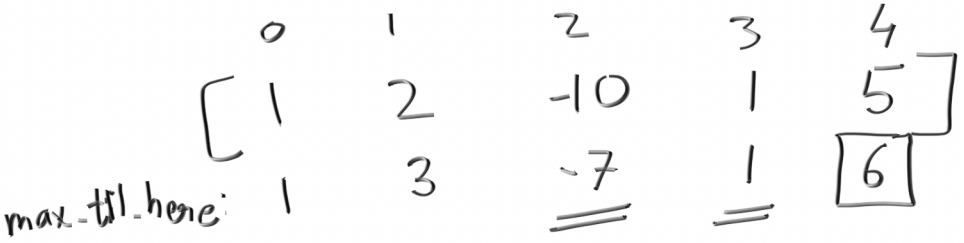
Input:
N = 5
Arr[] = {1,2,3,-2,5}
Output:
9
Explanation:
Max subarray sum is 9
of elements (1, 2, 3, -2, 5) which
is a contiguous subarray.

Example 2:

Input:
N = 4
Arr[] = {-1,-2,-3,-4}
Output:
-1
Explanation:
Max subarray sum is -1
of element (-1)







mth[0] = arr[0]

i >= 1 : mth[i] = mth[i-1] + arr[i]
if mth[i-1] > 0
mth[i] = arr[i] otherwise

mth[i] = maximum sum of any subarray ending EXACTLY at index i.

TC: O(n) Aux space = O(n)

	0	1	2	3	4	5
arr:	1	2	-1	1	3	-2
mth:	1	3	2	3	6	4

```
Input : arr[] = {1, 3, 5, 5, 5, 5, 67, 123, 125}
        x = 5
Output : First Occurrence = 2
         Last Occurrence = 5

Input : arr[] = {1, 3, 5, 5, 5, 5, 7, 123, 125 }
        x = 7
Output : First Occurrence = 6
         Last Occurrence = 6
```

0	1	2	3	4	5	6	7
1	3	5	5	5	5	67	123

x = 5

low = 0 high = 7

mid = $\frac{0+7}{2}$ = 3 → first_occ = 3

arr[mid] == x

low = 0 high = 2

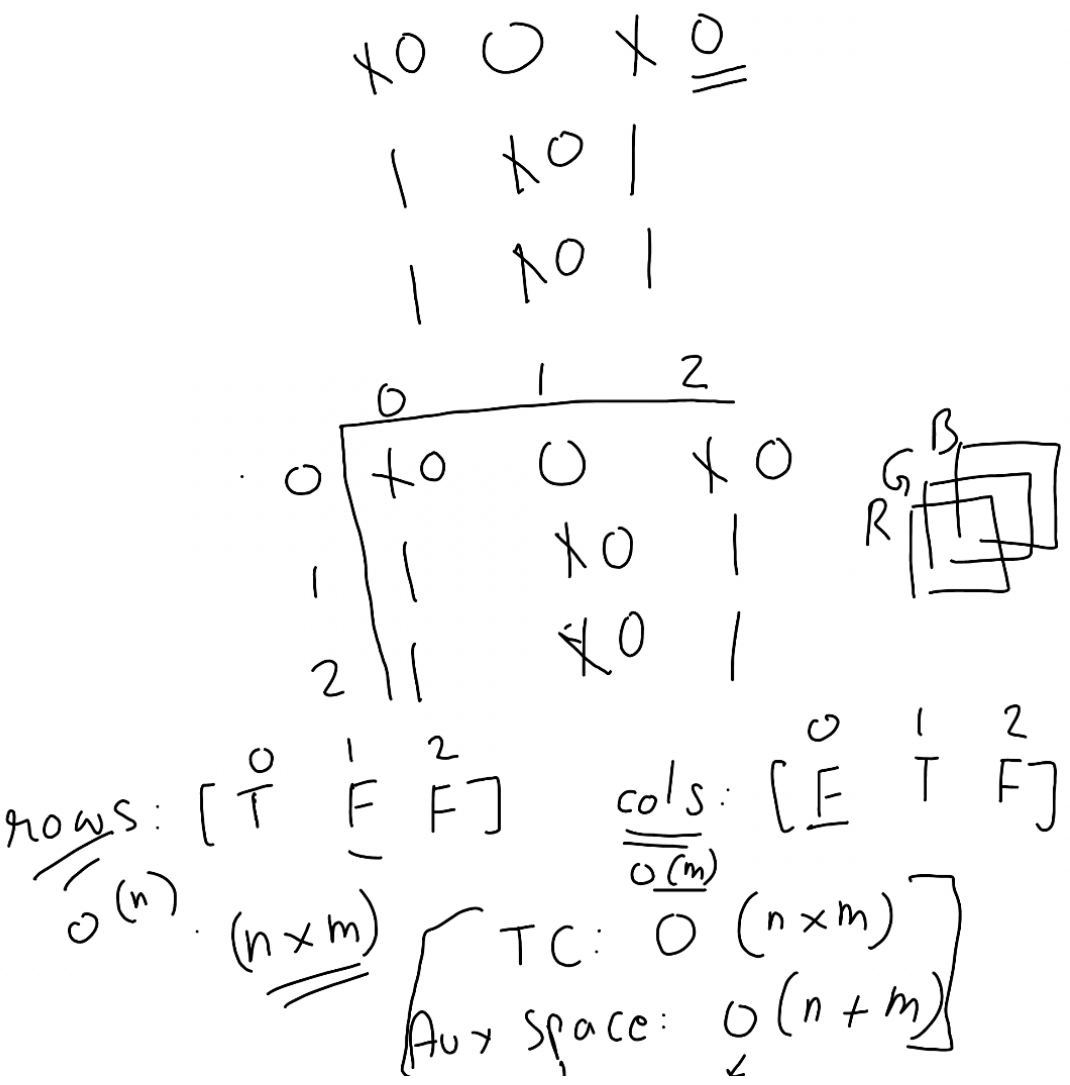
mid = $\frac{0+2}{2}$ = 1 (arr[mid] < x)

low = 2 high = 2

mid = $\frac{2+2}{2}$ = 2 → first_occ = 2

- arr[mid] == x

(low = 2 high = 1) STOP!



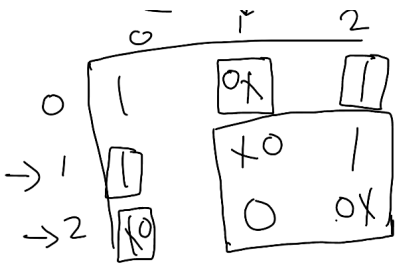
rows: [T F F] cols: [F T F]

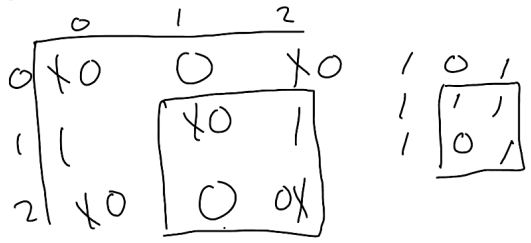
$O(n)$ $O(m)$

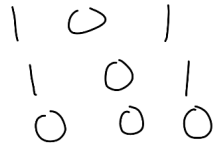
$(n \times m)$

TC: $O(n \times m)$

Aux space: $O(n+m)$







bool first_row = T TC: O(n×m)

bool first_col = F Aux space: O(1)