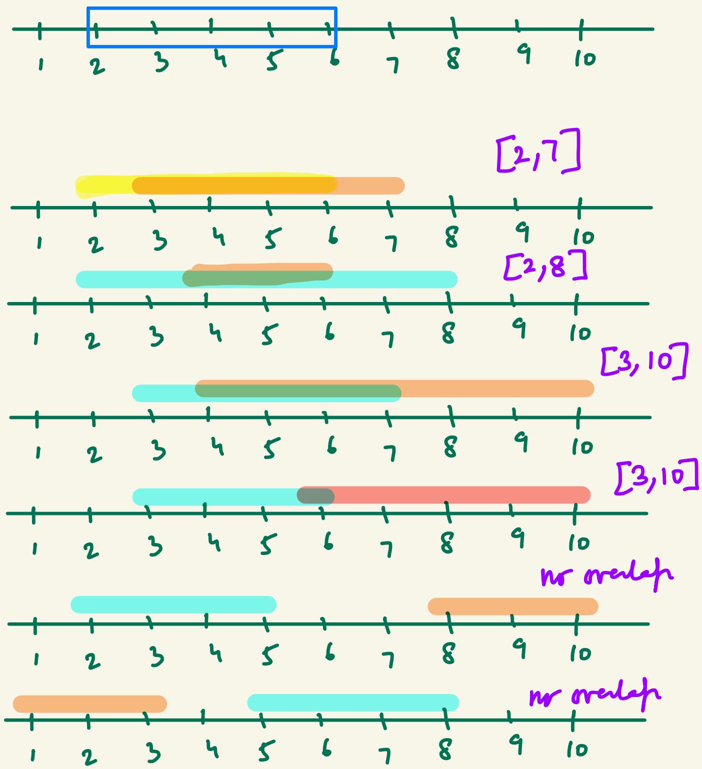
"If you want to shine like a sun,
first burn like a sun"

– Dr. APJ Abdul Kalam

Interval → range
[start, end]

$I_1 = [2,6]$



I_1 (s_1, e_1)	I_2 (s_2, e_2)
[2,6]	[3,7]
[2,8]	[4,6]
[3,7]	[4,10]
[3,6]	[6,10]
[2,5]	[8,10]

$(e_1 < s_2)$

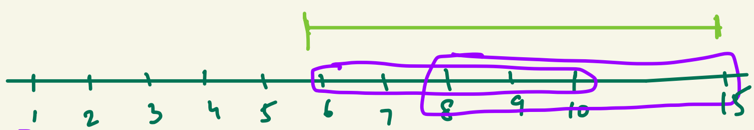
[5,8]	[1,3]

$(e_2 < s_1)$

if $(e_1 < s_2 \,||\, e_2 < s_1)$ ⟶ no overlap.

else
merged_interval → $[\min(s_1, s_2), \max(e_1, e_2)]$

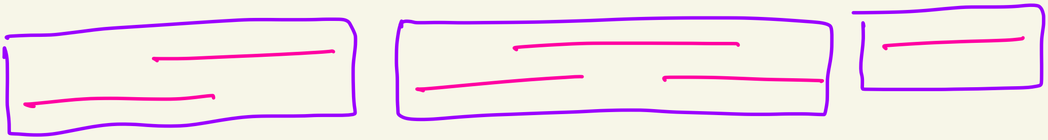
[6,10] [8,15]



$[\min(6,8), \max(10,15)]$
[6, 15].

Q1) Given a collection of intervals A in a 2-D array format, where each interval is represented by a pair of integers [start, end]. The intervals are sorted based on their start values.

Merge all overlapping intervals and return the resulting set of non-overlapping intervals.



eg:-

{ {0, 2}, {1, 4}, {5, 6}, {6, 8}, {7, 10}, {8, 9}, {12, 14} }

I_1	I_2			Answer Interval List
{0, 2}	{1, 4}	→	Overlapping	{0, 4}
{0, 4}	{5, 6}	→	Non-over.	{0, 4}, {5, 6}
{5, 6}	{6, 8}	→	Overlapping	{0, 4}, {5, 8}
{5, 8}	{7, 10}	→	Overlapping	{0, 4}, {5, 10}
{5, 10}	{8, 9}	→	Overlapping	{0, 4}, {5, 10}
{5, 10}	{12, 14}	→	Non-over.	{0, 4}, {5, 10}, {12, 14}

{ {0, 4}, {5, 10}, {12, 14} }

→ Create an array to store the merged intervals

→ If current and new (i^{th}) intervals overlap, then merge them. The merged interval becomes the current interval

→ Else, insert the current interval into the answer array and make the new interval as the current interval

```
int currS = A[0][0], currE = A[0][1]
for (i→1 to n-1) {
    if (A[i][0] > currE) {
        ans.insert({currS, currE})
        currS = A[i][0]
        currE = A[i][1]
    }
    else {
        currE = max(currE, A[i][1])
    }
}
ans.insert({currS, currE})
```

O(n) T.C

O(1) S.C.

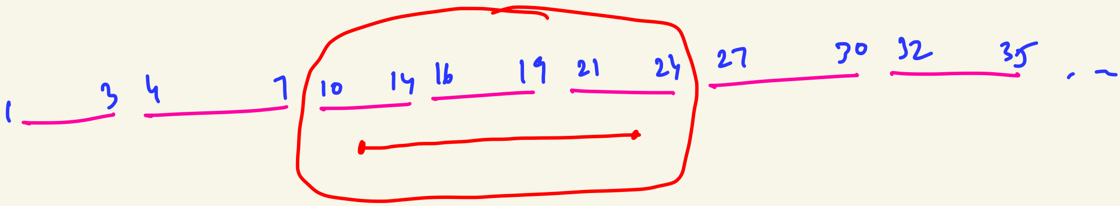
Q2) You have a set of non-overlapping intervals. You're given a new interval [start, end]. Insert this new interval into the set of intervals (merge if necessary)

initially sorted based on start time.

$n = 9$

{ {1,3}, {4,7}, {10,14}, {16,19}, {21,24}, {27,30}, {32,35},
{38,41}, {43,50} }

New interval → {12, 22}



ith	new interval	Overlapping	Print
{1,3}	{12,22}	✗	{1,3}
{4,7}	{12,22}	✗	{4,7}
{10,14}	{12,22}	✓ {10,22}	–
{16,19}	{10,22}	✓ {10,22}	
{21,24}	{10,22}	✓ {10,24}	
{27,30}	{10,24}	✗	{10,24}
{27,30}	–		{27,30}
{32,35}	–		{32,35}
{38,41}	–		{38,41}
{43,50}	–		{43,50}

n = 5

{{1, 5}, {8, 10}, {11, 14}, {15, 20}, {21, 24}}

new interval → {12, 24}

ith		new interval	overlaps?	Print
{1, 5}	<	{12, 24}	✗	{1, 5}
{8, 10}	<	{12, 24}	✗	{8, 10}
{11, 14}		{12, 24}	✓ {11, 24}	
{15, 20}		{11, 24}	✓ {11, 24}	
{21, 24}		{11, 24}	✓ {11, 24}	{11, 24}

```
ans = [ ]     // vector<vector> n   List<List<Integer>>
L, R → new interval
for (i → 0 to n-1) {
    if (A[i][1] < L)
        ans.insert({A[i][0], A[i][1]})
    else if (R < A[i][0]) {
        ans.insert({L, R})
        for (j → i to n-1) {
            ans.insert({A[j][0], A[j][1]})
        }
        return ans
    }
    else {
        L = min(L, A[i][0])
        R = max(R, A[i][1])
    }
}
ans.insert({L, R})
return ans
```

O(n) T.C.

O(1) S.C.

Q3) Given an unsorted list of integers, find the first missing natural number.

arr → {1, 2, 5, 8, 6, 4, 3, 10}   ans → 7

{3, -2, 1, 2, 7}   ans → 4

{-9, 2, 6, 4, -8, 1, 3}   ans → 5

{5, 2, 1, 3}   ans → 4

{-4, 8, 3, -1, 0}   ans → 1

{5, 3, 1, -1, -2, -4, 7, 2}   ans → 4

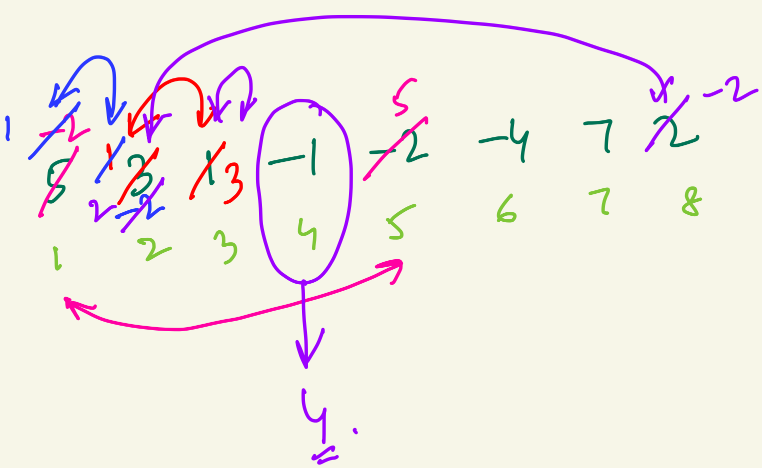
arr[n]

ans → [1, n+1]

⇓

For each i in [1, n+1], search i in the array arr.
First i that is ∉ arr is the answer.

T.C. → $O(ans * n) \sim O(n^2)$.



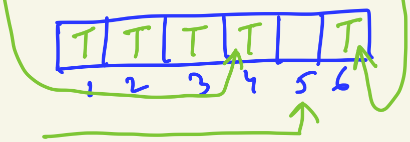
ans → [1, n+1]

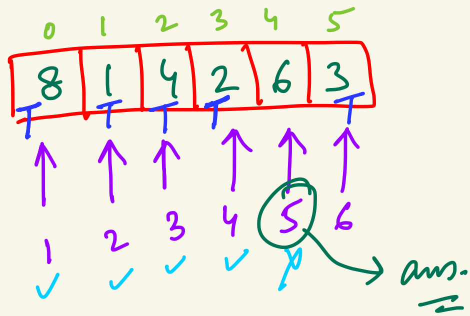
Assume → All the elements are +ve.

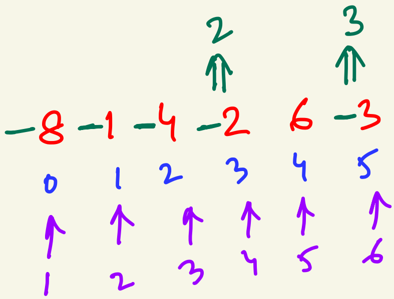
{8, 1, 4, 2, 6, 3}

ans → [1, 7]

n = 6








```
for (i -> 0 to n-1) {
    x = abs(A[i])
    if (x >= 1 && x <= n) {
        A[x-1] = (-1) * abs(A[x-1])
    }
}
for (i -> 1 to n) {
    if (A[i-1] > 0)
        return i
}
return (n+1)
```

O(n) T.C.

O(1) S.C.

a[i] can be +ve, -ve, or 0.

{4, ~~0~~, 1, ~~-5~~, ~~-10~~, -8, 2, -6}

-10 -10 10

1 2 3 4 5 6 7 8

ans = 3.

n = 8

[ans <= 9]

```
for(i -> 0 to n-1){
    if(A[i]<=0){
        A[i] = n+2
    }
}
for(i -> 0 to n-1){
    x = abs(A[i])
    if(x>=1 && x<=n){
        A[x-1] = (-1) * abs(A[x-1])
    }
}
for(i -> 1 to n){
    if(A[i-1] > 0)
        return i
}
return (n+1)
```

$\boxed{\begin{array}{l} O(n)\ T.C. \\ O(1)\ S.C \end{array}}$