Array Interview problems

1.1 - Drone Jump

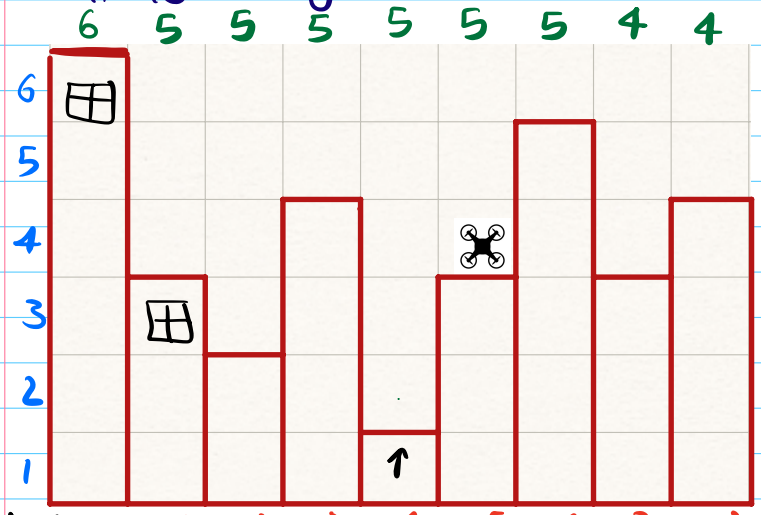
1.2 - Rain Water trap

2 - Intervals 1

3 - Intervals 2

4 - first missing integer

P1.1 Given n buildings, with height of each building. there is a small drone (quadcopter) on top of each building which can only move to up & right. Find how high at max a drone can go and land from current location.



h: { 6[0], 3[1], 2[2], 4[3], 1[4], 3[5], 5[6], 3[7], 4[8] } ← input 1D array

maxR: { 6 5 5 5 5 5 5 4 4 }

Q1 ans: { 0 2 3 1 4 2 0 1 0 }

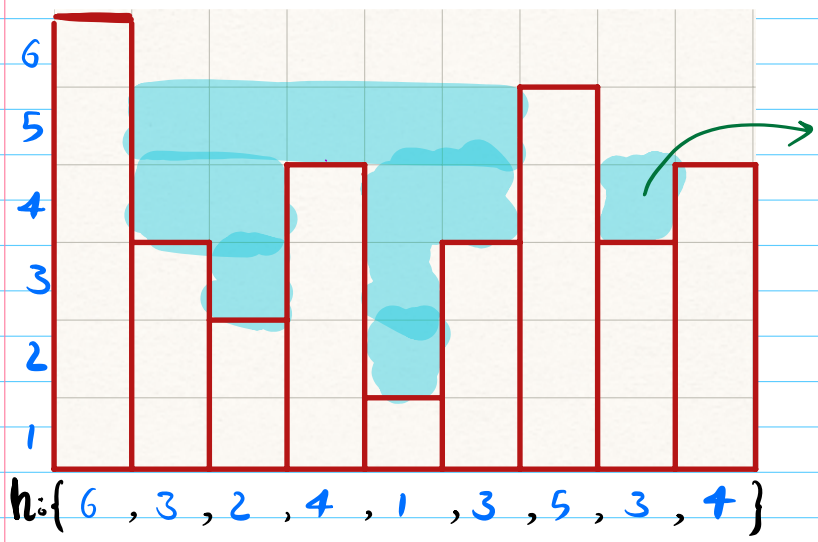
TC: O(n)
SC: O(1)

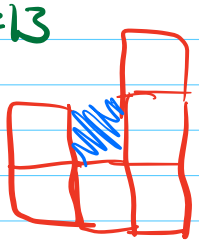
```
int maxHeight(int a[]){
    n = a.Len;
    maxR = a[n-1]; ans = 0;
    for(i = n-2; i >= 0; i--)
    n|    maxR = MAX(maxR, a[i])
          ans = MAX(maxR - a[i], ans)
    }
    ret ans
}
```

P1.2 Rain water

Similar input format as last problem.

Find the amount of rain water trapped between the buildings.





Q2

TC: O(n)

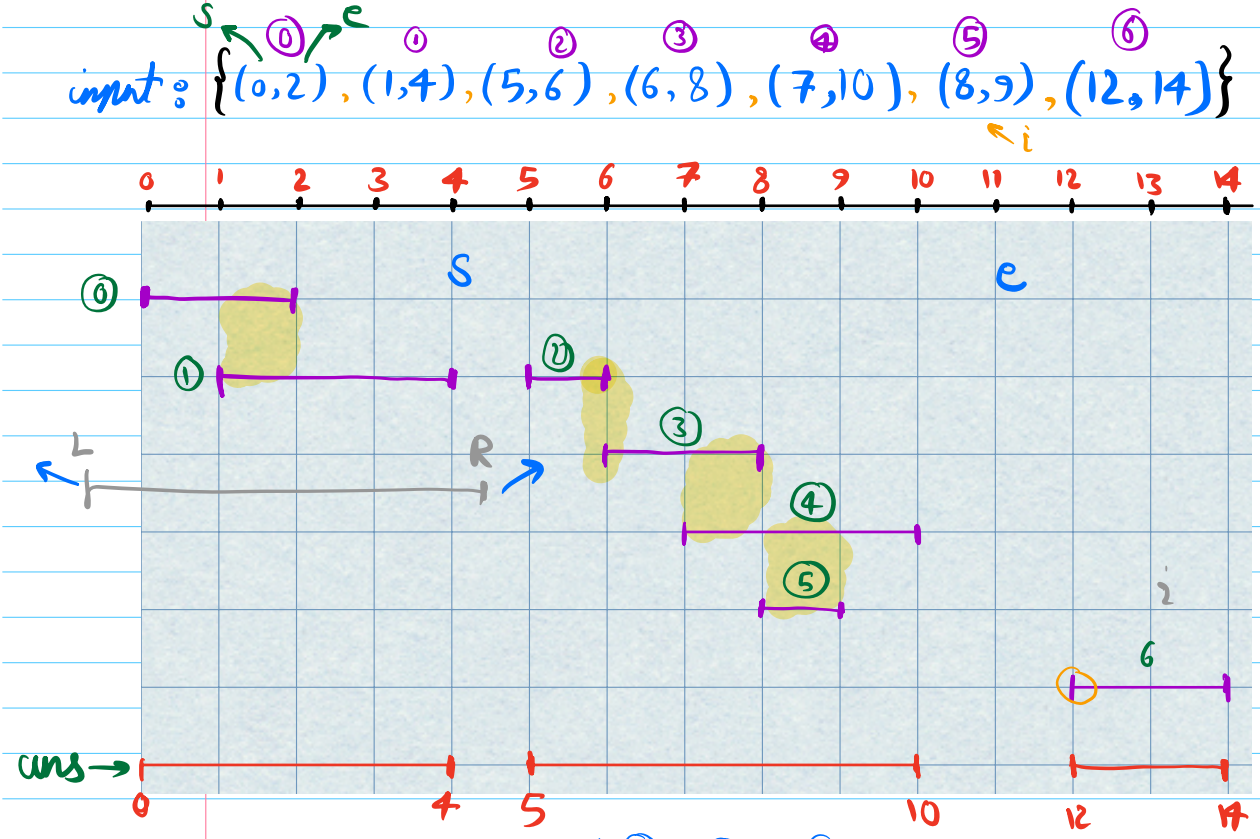
SC: O(n)

```
int totalRainWater(int a[]){
    n = a.Len
    maxR[n], maxL[n]
    maxR[n-1] = a[n-1]
    for(i=n-2; i>=0; i--)   maxR[i] = MAX(maxR[i+1], a[i])
    maxL[0] = a[0]
    for(i=1; i<n; i++)      maxL[i] = MAX(maxL[i-1], a[i])
    ans=0
    for(i=0; i<n; i++){
        ans+= MIN(maxL[i], maxR[i]) - a[i]
    }
    ret ans.
}
```

Can be merged

P2 Given a sorted list of overlapping interval, sorted based on start, Merge all overlapping intervals & return the sorted list.

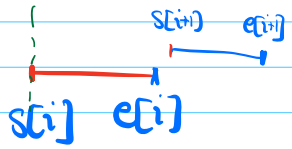
input: {(0,2), (1,4), (5,6), (6,8), (7,10), (8,9), (12,14)}



Observation:

(i)	0	1	2	3	4	5	6
(L,R)	(0,7)	(0,4)	(5,6)	(5,8)	(5,10)	(5,10)	(12,14)

① if interval i & j overlap
all (i, i+1, i+2, ..., j) will overlap



② start from left, move right, merge

③ Interval i, i+1 overlap? $S[i] \le S[i+1]$ $S[i+1] \le e[i]$ → then true

④ i, i+1 overlap → merged result

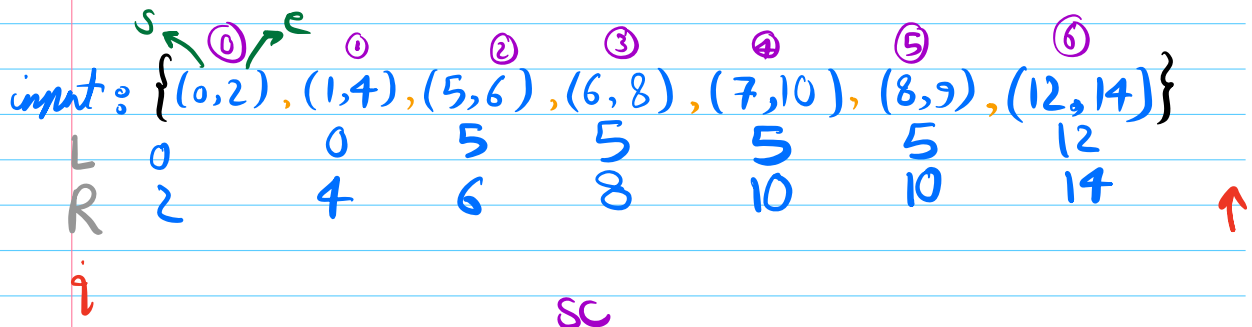
$$\begin{cases} S = \min(S[i], S[i+1]) \rightarrow S[i] \\ e = \max(e[i], e[i+1]) \end{cases}$$

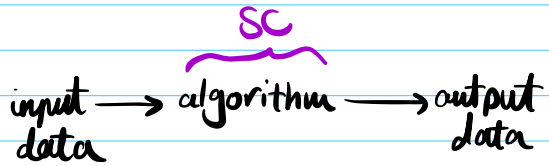
```
List<Pair<int,int>> mergeIntervals(int s[], int e[]){
    n = S.Len; //==e.Len.
    output = new List<Pair<int,int>>();
    L = S[0]   R = e[0]
    for( i=1; i<n; i++){
        if( s[i]<=R){ R = max(R, e[i]) }
        else{ output.add( Pair(L,R))
              L = S[i];  R = e[i]
        }
    }
    output.add( Pair(L,R));
    ret output
}
```

Q3

TC: O(n)

SC: O(1)





output (0,4), (5,10), (12,14)

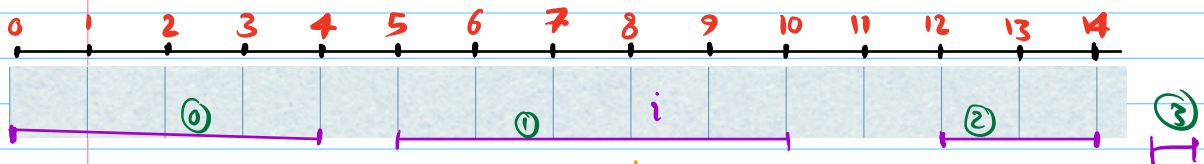
P3 Given a set of sorted & non overlapping intervals, insert a new interval such that the final list of intervals is also sorted & non-overlapping

input

ex (0,4), (5,10), (12,14)

new (8,12)

output list of pairs



ans:

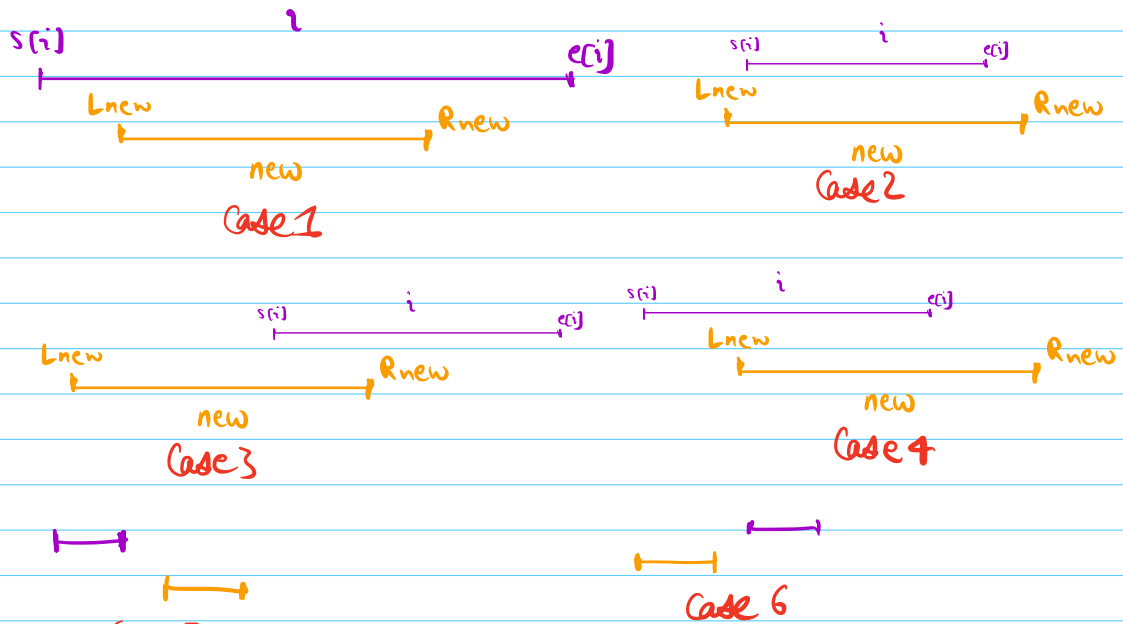
Lnew new Rnew

TC:

SC:

s[i]
e[i]

if(s[i]<=Rnew && e[i]>=Lnew) then merge



Code assignment

(not sorted, no duplicates)

P4 Given an array of integers, find the first missing positive(+) integer. Constraint SC: O(1)

ex a: {-5, -3, 10, 8, 1, 2, 4, -3}   1, 2, 3, ...
ans = 3   (3 → ans)

idea1: Brute Force $O(n^2)$ { for(i=1; i<=n+1; i++)
search(i, a) → O(n)

Q4 → n+1
[1, 2, 3, 4, 5, 6]

idea2: a: {-5, -3, 10, 8, 1, 2, 4} → O(n log n)
TC: O(n log n)   {-5, -3, 1, 2, 4, 8, 10}   while(-) i++
SC: O(1)   O(n)   j=3   j=1   for(; i<n ; i++)
ans ← ret   if(a[i] != j) ret j
} j++

idea3: a: {-5, -3, 10, 8, 1, 2, 4, -3}

MEX

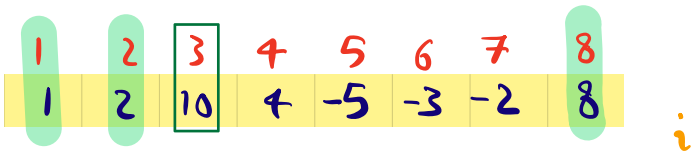
bool exists{ T T F T   T }   (indices 0 1 2 3 4 5 6 7 8 9)
→ ans

O(n) ← SC   a[i] <= 0 ignore
a[i] >= n+1 ignore
exists[a[i]] = true ,

O(n) ← TC   check if items are unique

for(i=0; i<n; i++) → swap
while(....)
if (1 <= a[i] <= n) swap2(i, a[i])
(a[a[i]], a[i])

TC: a: { 1, 2, 10, 4, 5, -3, -2, 8 }   (indices 1 2 3 4 5 6 7 8)
SC:



-5, -3, 10, 8, 1, 2, 4, -2