1Q) Given N array elements, construct a new b[], b[i] indicates the smaller element to the right of A[i]

	0	1	2	3	4	5	6	7	8	9	10	11
A[] :	10	13	18	15	20	25	18	12	15	6	10	8
B[]	6	12	15	12	18	12	12	6	6	-1	8	-1

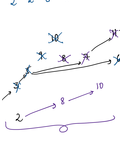
15 > pop 18 = 15

12
10
10

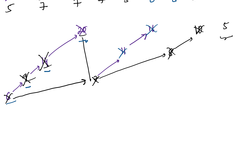
Note: we will insert index into stack

b[] = -1
for(int i = 0; i < N; i++){
  for(int j = i+1; j < N; j++){
    if(ar[j] < ar[i]){
      b[i] = ar[j];
      break;
    }
  }
}
TC: $O(N^2)$
SC: $O(1)$

Ex: 3 5 1 10 8 7 11 13 6 2 8 10
2 2 8 8 7 6 6 6 2 -1 -1 -1



Ex2: 6 7 14 20 7 11 16 8 10 5
5 7 7 7 6 8 5 5 -1



	0	1	2	3	4	5	6	7	8	9	10	11
Ex1:	10	13	18	15	20	18	25	12	15	6	10	8
b[]		12	15	12	18	12	12					

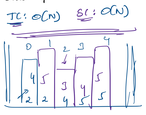
Step1: If stack is empty directly insert index → New Element

Step2: We will comp A[i] with A[st.top]

if (A[i] ≥ A[st.top]){
  push i to stack
}
else while (A[st.top] > A[i]){
  int x = st.top();
  st.pop();
  b[x] = A[i];
}
st.push(i)
}

// b[N] = -1 : Already initialized all value with -1
// stack<int> st;
for(int i = 0; i < N; i++){
  if(st.size() == 0){
    st.push(i);
  }
  else{
    while (A[st.top()] > A[i]){ → new value
      int x = st.top(); → top of stack
      st.pop();
      b[x] = A[i];
      if(st.size() == 0) break;
    }
    st.push(i);
  }
}

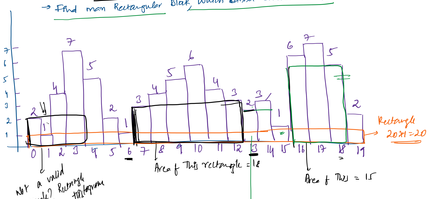
TC: → Every A[i] element will go inside once & come outside once.
TC: $O(N)$ SC: $O(N)$



2Q) Given contiguous blocks of histograms find max histogram area

ar[]: 2 1 4 7 5 2 1 3 4 5 6 4 3 2 3 1 6 7 5 2

→ Find max Rectangular Block which cannot exceed histograms.



Say for rectangle we fix H = 2
6 13 15
We cannot extend to 6
We cannot extend
[Have smaller index on right]
[Have smaller index on left side]
→ Width: 15 - 6 - 1 = 8
Area = 2 * 8 = 16

For a given i
- if FSI on left = P_1
- if ESR on Right = P_2

Area with A[i] has height → $(P_2 - P_1 - 1) * A[i]$

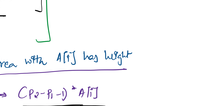
Step1: Each 1st smaller index on left for every element
$P_1[]$ → TC: $O(N)$ → $P_1[] = -1$

Step2: Each 1st smaller index on right for every elem
$P_2[]$ → TC: $O(N)$ → $P_2[] = -1$ ← wrong
$P_2[] = N$

ans =
for(int i = 0; i < N; i++){
  ans = max(ans, A[i] * (P2[i] - P1[i] - 1))
}
return ans;
Overall TC: $O(N)$

TC: $O(N)$

Ex:



These have to be default initializations
$P_1 = -1$
$P_2 = n$
$(P_2 - P_1 - 1) * 2$ → $(-0-1)*2 = -2$
$(2+1-1) * 2$ → 4

3Q) Given a matrix which contains only 1 & 0, find max rectangular area which contains all 0's

N rows M cols

	0	1	2	3	4	5
0	1	1	0	1	0	1
1	0	1	1	1	1	1
2	1	1	1	1	1	0
3	1	0	1	0	0	1
4	1	1	0	0	1	1

1	1	0	1	0	1
0	2	1	2	1	2
1	3	2	3	2	0
2	0	3	0	0	1
3	1	0	0	1	2

This has to get max histogram area
Max Histogram Area by row 4:

TC: We will apply Area of this area N times
N * [M + NM]
Size of Array
Because we have to traverse towards top
→ Overall TC: $N[M + (NM)]$

Time taken to fill entire thing: N^*M
→ For N rows Apply histogram logic
→ Final TC: $(N^*M + N^*M)$ SC: $O(M)$

For every column take pref sum
for start → Iterating M columns
for(int j = 0; j < M; j++){
  pf = 0;
  for(int i = 0; i < N; i++){ → Iterating N rows
    if mat[i][j] = 0
      pf = 0
    else
      pf += mat[i][j]
      mat[i][j] = pf
  }
}

4Q) Double Character Trouble:

Given a string if 2 same characters are adjacent to each other delete them.

Q: a b c ~~a~~ ~~a~~ c ~~a~~ ~~a~~ ~~a~~ ~~a~~ a d

Final String = abcead

Solution:
If str[i] is not same as top → push str[i]
If str[i] is same as top, pop()
If stack is empty, directly push str[i]

5Q) Google: K character trouble:

Given a string if a character comes k consecutive times delete it

k = 3

Ex: ~~a~~ ~~b~~ d ~~d~~ ~~d~~ ~~b~~ ~~b~~ ~~a~~ ~~a~~ ~~a~~ ~~b~~ ~~b~~ ~~b~~ ~~a~~ ~~a~~
Empty string:

How: Stack / push() / pop() → Insert ele, freq to stack

Logic:
If stack empty Insert <ar[i],1>
If ar[i] is not same has top element
→ Insert ar[i],1
If ar[i] is same has top.
→ Increase freq of top element by 1
→ If freq == k, pop the top element

TC: $O(N)$
SC: $O(N)$