

Today's Content:

1. Leaders in Array
2. Buy & sell stocks
3. Adj pairs

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

1

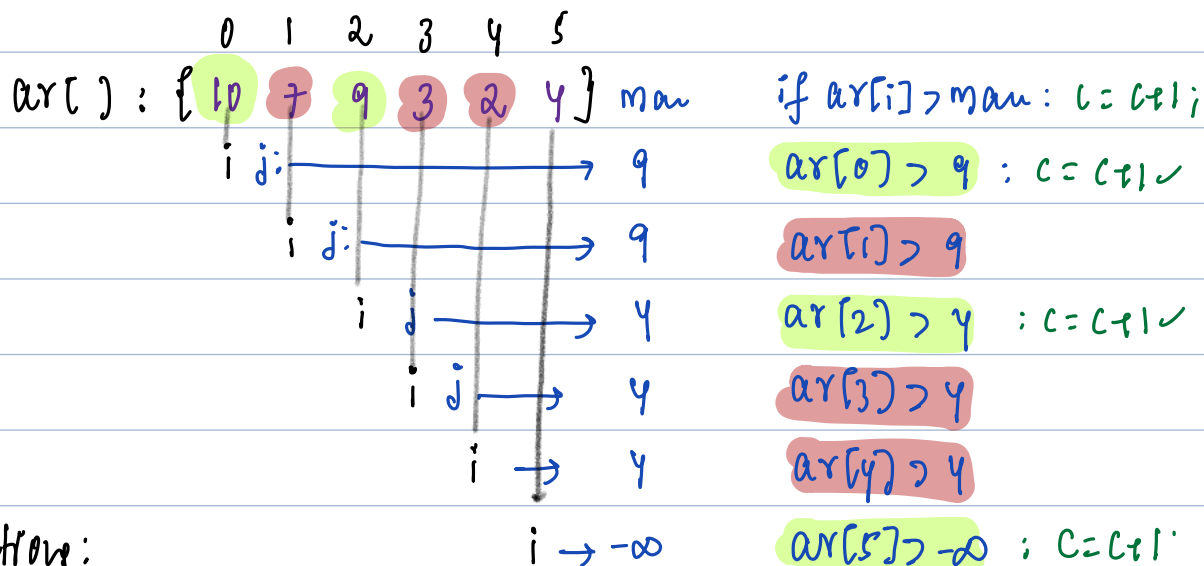
1

1

1

1

Tracing:



Observation:

for every arr[i] we are calculating man on right side.

Carry forward Idea:

When we calculate same data from L → R or R → L multiple times use carry forward & calculate data in only 1 iteration.

Note: We iterate from direction we are calculating

Optimization Using Carry Forward:

arr[]:	0	1	2	3	4	5	6	
	10	7	9	3	6	4	5	man = -∞
	10 > man	7 > man	9 > man	3 > man	6 > man	4 > man	5 > man	c = 0;
return c, j	c++		c++		c++		c++	
	man = 10	man = 7	man = 9	man = 3	man = 6	man = 4	man = 5	

int leaders (int arr[], int N) { TC: $O(N)$ SC: $O(1)$

int max = INT_MIN, c = 0;

for (int i = N-1; i >= 0; i--) {

// arr[i] is leader > max on right

if (arr[i] > max) {

c++;

max = arr[i];

}

return c;

}

Buy & Sell Stocks:

Given an array $arr[N]$, where $arr[i]$ is price of given stock on i^{th} day

Return max profit which can be achieved by exactly 1 transaction

Notes: If we buy a stock on i^{th} day: We can sell on any day $\{i+1, i+2, i+3, \dots, n-1\}$

Note2: If cannot achieve any profit: return 0;

Constraints:

$$1 \leq N \leq 10^5$$

$$1 \leq arr[i] \leq 10^9$$

Ex1:

$$arr[] = \{7, 1, 5, 3, 6, 4\} \quad ans =$$

Ex2:

$$\begin{array}{ccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ arr[] = \{ & 4 & 6 & 10 & 4 & 2 & 9 & 1 \} \end{array}$$

Idea:

In Stock we

9

$$\begin{array}{ccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 \\ \underline{i} & \{ & 4 & 6 & 10 & 4 & 2 & 9 & 1 \} \end{array} \quad \underline{\text{Profit day } i^{th}}$$

Con:

T.C:

S.C:

Count Pairs "ab"

Given a char $s[N]$ calculate no. of pairs indices i, j such that $i < j$ & $s[i] == 'a'$ & $s[j] == 'b'$

Constraints:

$$1 \leq N \leq 10^5$$

$$'a' \leq s[i] \leq 'z'$$

Ex: $s = b \ a \ a \ b \ d \ c \ a \ b$

Pairs: (i, j) $(1, 3)$ $(1, 7)$ $(2, 3)$ $(2, 7)$ $(6, 7)$ $(6, 3)$

Idea: Generate all pairs & check if pairs forms a b & one c.

int pairs(char s[], int N) { TC: $O(N^2)$ SC: $O(1)$

int c = 0;

for (int i = 0; i < N; i++) {

for (int j = i + 1; j < N; j++) {

if ($s[i] == 'a'$ & $s[j] == 'b'$) {
c++;
}

return c;

Dry Run:

0 1 2 3 4 5 6 7
 $s = b \ a \ a \ b \ d \ c \ a \ b$

Idea: Only if $s[i] == 'a'$: We iterate in inner loop.

T.C: $O(N^2)$: Worst case = all a's.

int pairs(String s, int N) {

int c = 0;

for (int i = 0; i < N; i++) {

if (s[i] == 'a') {

for (int j = i + 1; j < N; j++) {

if (s[j] == 'b') {

c++;

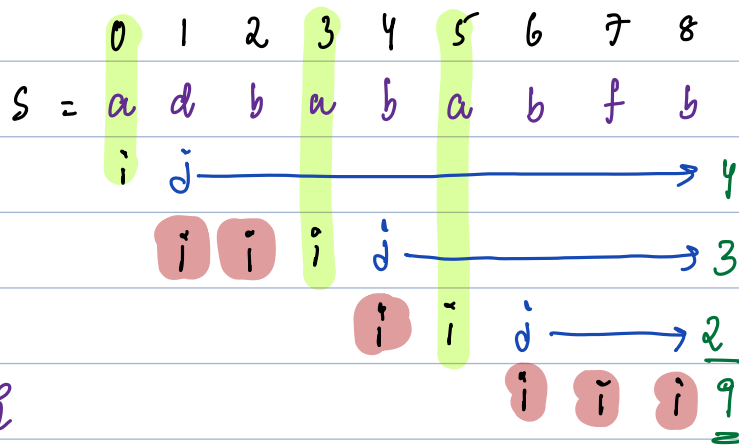
}

}

}

return c;

}



obs: For every $s[i] == 'a'$: Iterate in right & counting no of b's.

Optimization Idea: Using carry forward calculate no's of b's from right to left

	✓0	✓1	✓2	✓3	✓4	✓5	✓6	✓7	✓8	
s =	a	d	b	a	b	a	b	f	b	ans = 0 cb = 0
	ans = cb		cb + 1	ans = cb	cb + 1	ans = cb	cb + 1		cb + 1	
return	ans = 9		cb = 4	ans = 5	cb = 3	ans = 2	cb = 2		cb = 1	
cb = 9										

int pairs(char s[], int N) { T.C: $O(N)$ S.C: $O(1)$

int ans = 0, cb = 0;

for (int i = N - 1; i >= 0; i--) {

if (s[i] == 'b') { cb++; }

if (s[i] == 'a') { ans = ans + cb; }

return ans;

}

Idea3: Other way for above question

obs: For every $s[i] = 'b'$: Iterate on left & counting no of a's

Optimization Idea: Using carry forward calculate no's of a's from left \rightarrow right

TODO: Dry Run & Code.