

Todays Content:

1. Leaders in Array
2. Buy & sell stock
3. Ag pairs

Leaders in Array:

Given an $ar[n]$ return no. of leaders in $ar[]$

An $ar[i]$ is said to be leader if

$ar[i] > \text{max of all elements in right } [i+1 \dots n-1]$

Note: $ar[n-1]$ is said to be leader because no element in right

Constraints:

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

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

Ex1: 0 1 2 3 4 5 6 7

$ar[] = \{15 \ 1 \ 7 \ 2 \ 5 \ 4 \ 2 \ 3\}$ ans = 5

Ex2: 0 1 2 3 4 5 i

$ar[] = \{10 \ 7 \ 9 \ 3 \ 2 \ 4\}$ ans = 3

Ideas: For every $ar[i]$

Estimated TC: $O(N^2)$

Iterate $i+1 \dots n-1$ calculate max of chunk if $ar[i]$ is leader or not.

int leaders(int ar[], int n) { TC: $O(N^2)$ SC: $O(1)$

 int c = 0;

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

 int man = INT_MIN;

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

 if (ar[j] > man) {

 man = ar[j];

 }

 if (ar[i] > man) { // leader

 c++;

 }

Table T0DD

i	j

Inner loop :

Outer loop :

Total =

3 return c;

Tracing:

	0	1	2	3	4	5	
$ar[] : \{ 10, 7, 9, 3, 2, 4 \}$ man							if $ar[i] > man : l = i + 1;$
i: 1	j: 2						$ar[0] > 9 : c = c + 1 \checkmark$
i: 1	j: 3						$ar[1] > 9$
i: 1	j: 4						$ar[2] > 9 : c = c + 1 \checkmark$
i: 1	j: 5						$ar[3] > 9$
i: 1							$ar[4] > 9$
i: 1							$ar[5] > 9 : c = c + 1 \checkmark$
i: 1							$i \rightarrow -\infty$
i: 1							$ar[5] > -\infty : c = c + 1 \checkmark$

Observation:

For every $ar[i]$ we are calculating man in right side.

Carry forward Idea:

When we calculate same data from $L \rightarrow R$ or $R \rightarrow L$ multiple times we carry forward q. Calculate data in only 1 iteration.

Note: We iterate from direction we are calculating

Optimization Using Carry forward:

	\downarrow	\uparrow	\downarrow								
$ar[7] : \begin{array}{ccccccc} 0 & 1 & 2 & 3 & 4 & 5 & 6 \end{array}$	10	7	9	3	6	4	5				
	10 > man	7 > man	9 > man	3 > man	6 > man	4 > man	5 > man				
return cij	c_{ij}		c_{ij}		c_{ij}		c_{ij}				
	$man = 10$	$man = 9$	$man = 9$	$man = 6$	$man = 6$	$man = 5$	$man = 5$				

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

```
int man = INT_MIN, c = 0;
```

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

```
// arr[i] is leader > man on right
```

```
if(arr[i] > man) {
```

```
c++;
```

```
man = arr[i];
```

```
}
```

```
return c;
```

```
}
```

Buy & Sell Stocks:

Given an array $ar[n]$, where $ar[i]$ is price of given stock in i^{th} day

Return max profit which can be achieved by exactly 1 transaction

Note: 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:

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

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

Ex1:

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

Ex2: 0 1 2 3 4 5 6

$$ar[] = \{4, 6, 10, 4, 2, 9, 1\}$$

Idea: In Stock we 9

0	1	2	3	4	5	6
<u>i</u>	$\{4, 6, 10, 4, 2, 9, 1\}$	<u>Profit day i^{th}</u>				

Con:

Tc: Sc:

Count Pairs "ab"

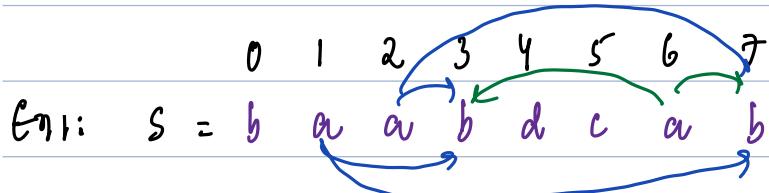
Given a char str[N] calculate no: of pairs indices i, j such that

i & j & s[i] := 'a' & s[j] := 'b'

Constraints:

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

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



Ex: s =

Pairs = (i, j) (1, 3) (1, 7) (2, 3) (2, 7) (6, 7) (6, 3)

i & j

Ideal: Generate all pairs & check if pair forms ab & ignore c.

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

int c=0;

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

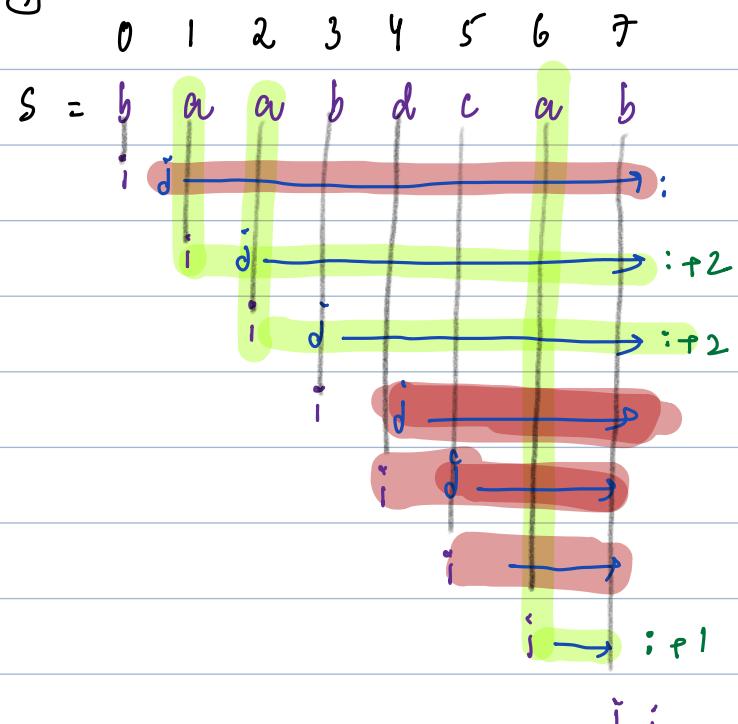
 for (int j=i+1; j<n; j++) {

 if (str[i]== 'a' && str[j]== 'b') {

 c++;

} }

Dry Run:



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

Tc: $O(n^2)$: 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') {
```

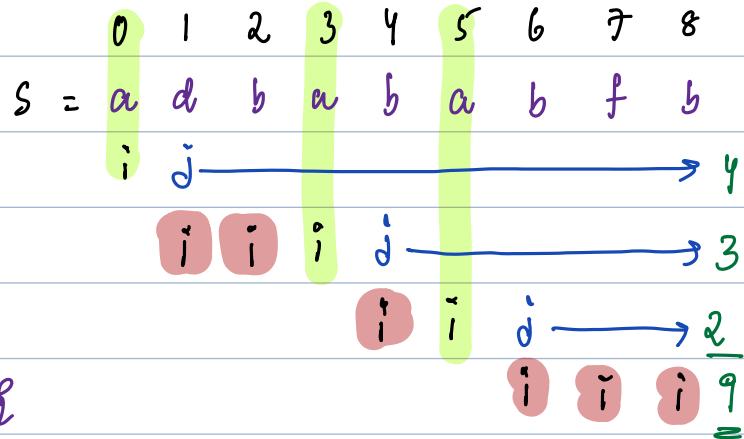
```
            for(int j=i+1; j<N; j++) {
```

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

```
                    c++;
```

```
}
```

```
    return c;
```



Obs: For every $s[i] == 'a'$: iterate in right q counting no of b's.

optimization idea: Using carry forward calculate no's of b's from right to left

	✓	✓	✓	✓	✓	✓	✓	✓	✓	ans=0
$s =$	a	d	b	a	b	a	b	f	b	$cb=0$
	$ans_1 = cb$	$cb+1$	$ans_2 = cb$	$cb+1$	$ans_3 = cb$	$cb+1$	$cb+1$		$cb+1$	
return	$ans = 9$	$cb=9$	$ans = 5$	$cb=3$	$ans = 2$	$cb=2$		$cb=1$		
carry = 9										

```
int pairs(char sc[], int N) { Tc: O(N) Sc: 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;
```

Ideas: 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 $\text{left} \rightarrow \text{right}$

TODO: Dry Run & Code.