# Advanced DSA: Arrays 1

22 May 2023 20:30

**Q1** Given an integer array of size N. Find the max. subarray sum.

Amazon Microsoft ZoomCar

Eg: A: { 1, -4, 5, 7, 11, 13, -59 }

Ans = 36

**Soln** 1) Brute force

⇒ ∀ subarrays, calculate the sum & take max.

**Code**

```
int ans = INT_MIN;
for (i=0; i<N; i++) {
    sum = 0;
    for (j= i; j<N; j++) {
        // Subarray from i to j
O(N) ⇐ 1) Iterate from i to j
        2) Prefix Sum ⇒ T.C. = O(1)
                        S.C. = O(N)
        3) Carry forward ⇒ O(1)
           sum = sum + A[j];
           ans = max (ans, sum);
    }
}
```

T.C. = $O(N^2)$

↓

O(N) ???

2) Optimise ??

**Q** What will be the max subarray sum if :

1) All elements are +ve ??

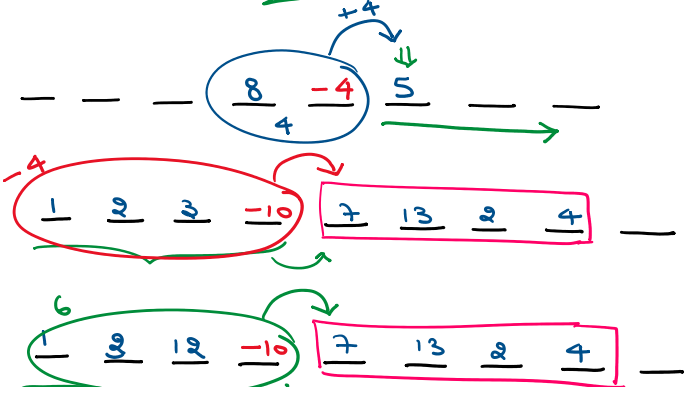
⇒ All elements.

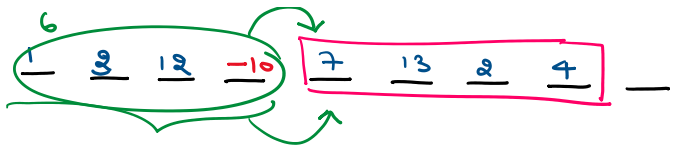
2) All elements are -ve

A = { -3, -1, -5, -7, -20, -6 }

max

Ans






ans = INT_MIN

## Kadane's Algorithm. (Carry forward)

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|---|---|
| A: | 5, | 6, | 7, | -3, | 2, | -10, | -12, | 8, | 12, | 21 |
| Sum: | 5, | 11, | 18, | 15, | 17, | 7, | -5, | 8, | 20, | 41 |
| ans: | 5, | 11, | 18, | 18, | 18, | 18, | 18, | 18, | 20, | 41 |

T.C. = O(N)

S.C. = O(1)

Ans

## Code

```
ans = INT_MIN;
Sum = 0;
for (i=0; i<N; i++) {
        Sum = Sum + A[i];
        ans = max (ans, Sum);
        if (Sum < 0) {
            Sum = 0;
        }
}
```

## Code_2 (find Subarray)

```
ans = INT_MIN;
start = -1;
end = -1;
Sum = 0;
s = 0;
for (e=0; e<N; e++) {
        Sum = Sum + A[e];
        if (Sum > ans) {
            ans = sum;
            start = s;
            end = e;
        }
        if (Sum < 0) {
            Sum = 0;
            s = e+1;
        }
}
```

41

ans = 7 8 20

start = 0 4 4 4

end = 0 4 5 6

A = { 7, -3, -2, -

0 1 2

T.C. = O(

S.C. = O

---

## Beggars Outside Temple (Amazon, MS, Google)

Q Given an Array of size N.

Initially ∀i, A[i] = 0;

sum = 0 0 26 41

s = 0

0

-15

s = 8

3 | 4 5 6 7

-10, 8 12, 21, -56

N)

(1)

Initially $\forall i$, $A[i] = 0$;

Q queries ⇒ index, value

| Queries | | A: | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|---|---|
| index | value | | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| | | | | +3 | +3 | +3 | +3 | +3 | +3 |
| 1 | 3 | | 0 | 3 | 3 | 3 | 3 | 3 | 3 |
| | | | | | | | +2 | +2 | +2 |
| 4 | 2 | | 0 | 3 | 3 | 3 | 5 | 5 | 5 |
| | | | | | +1 | +1 | +1 | +1 | +1 |
| 2 | 1 | | 0 | 3 | 4 | 4 | 6 | 6 | 6 |

Ans

Soln 1) Brute force:

⇒ ∀ queries, iterate from index to end & add the value to each element.

Code

```
int ans[N] = {0};                          // Output

for (i=0; i<Q; i++) {                      // O(Q)
    (input (index, value);
    for (j= index; j<N; j++) {             // O(N)
        ans[j] += value;
    }                                      > Optimise ??
}
return ans;
```

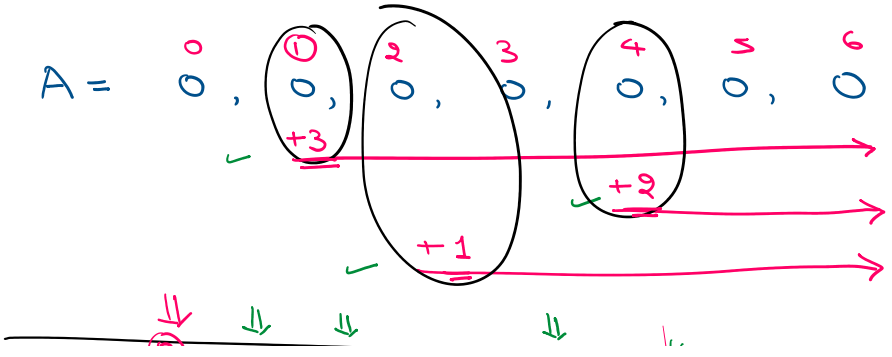
T.C. = O(N×Q)

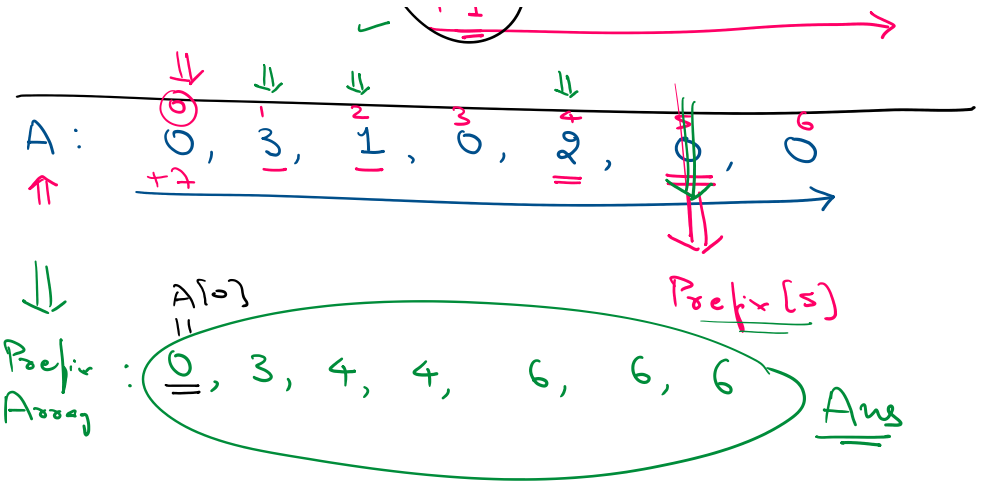
S.C. = O(1)

↑ (TLE)

$1 \le N \le 10^5$ , $1 \le Q \le 10^5$

2) Optimise



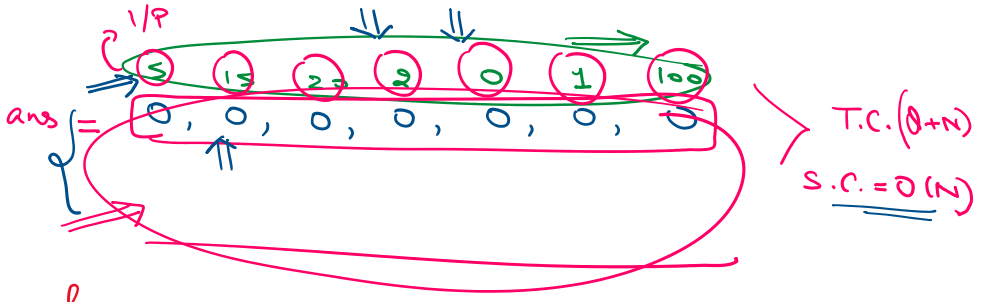



## Code

```
int ans[N] = {0};
for (i=0; i<Q; i++) {
    (input (index, value));
    ans[index] += value;
}

for (i=1; i<N; i++) {
    ans[i] = ans[i-1] + ans[i];
}
```

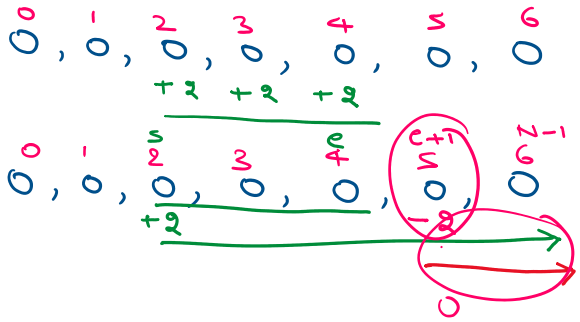
T.C. = O(N+Q)
S.C. = O(1)




## Google Variation

Query ⇒ start, end, value

Query
s, e, v
2 4 2



## Code

```
int ans[N] = {0};
for (int i=0; i<Q; i++) {   (10 times)
    (input (s, e, value));
    ans[s] += value;
    if (e < N-1) {
        ans[e+1] -= value;   (If e == N-1 ??)
    }
}

for (i=1; i<N; i++) {
    ans[i] = ans[i-1] + ans[i];
}
```

---

## Doubt

```
i = 0;
j = 31;
while (i < j) {
    swap (A, i, j);
    i++;
    j--;
}

swap (A, i, j) {
    Implementation
}
```

int ⇒ signed_int

unsigned_int

---

(i & i<<1)

[1, $2^A$]

$2^5 = 32$

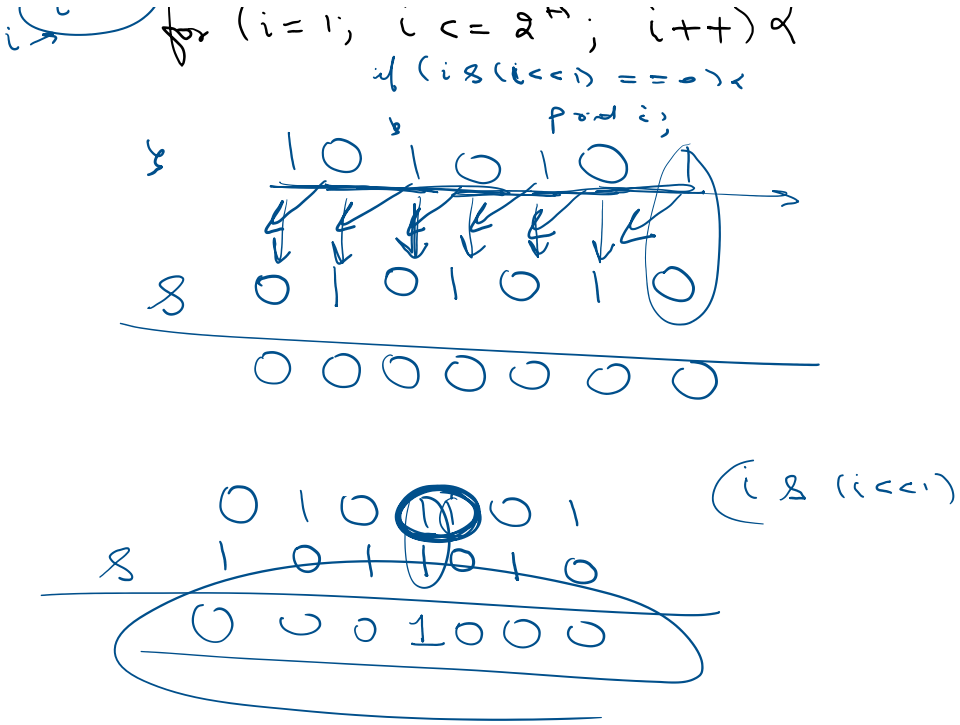
[1, 32]

0 & 0 = 0
0 & 1 = 0
1 & 0 = 0
1 & 1 = 1

i<<1

i →

```
for (i=1; i <= 2^A; i++) {
    if (i & (i<<1) == 0) {
```


for (i = 1; i <= 2^n; i++)
if (i & (i<<1) == 0)
print i;


i & (i<<1)