[1, 2, 3, 4, 5, 6]

4 bits.

6×4 = 24 bits

O(1)?

arr+3

arr+3

O(1)

---

arr+5 s×4 = 20

Bucketing

[1 2 3 4 5 6]

[0, -10, 1, 3, -20]

[-20, -10, 0, 1, 3]

TC : O(n log n)

0 1 2 3 4

[-20, -10, 0, 1, 3]

n=5

⇒ [0, 0, 0, 0, 0]

[1, 2, 3, 4, 5]

O(n²)

n+1

[1, n]

arr[i]-1

0 1 2 3 4 5

[-1, -3, -2, 0, 4, 8]

n = 6

[1, n]

arr[i]-1

4+1 = 5

0 1 2 3 4

[-10, -10, 0, 1, 3] n=5

-6 6 -6

i+1=2.

0 1 2

[-1, -2, -2]

[1, 2, 2] [-1, 3, -2]

0 1 2

[-1, -3, -2]

[1, n]

(0, n-1)

```
// Step-2: mark the buckets negative for the elements present in the array.
for (int i = 0; i < n; i++) {
    if (arr[i] <= n) {
        int bucketIndex = arr[i] - 1;
        if (arr[bucketIndex] > 0) {
            arr[bucketIndex] *= -1;
        }
    }
}
```

0 1 2

[-1, -3, -2]

n = 3

LI = 0 2 -1

2-1 = 1

---

0 1 0

1 1 0

1 1 0

⇒

0 0 0

1 0 1

0 0 0

n×m

0 1 2

0 1 0 1 0

1 1 0 1 1

2 0 1 0 1 0

now : [1, 0, 1]

col : [0, 1, 0]

n

m.

TC = O(n m)

AS = O(n+m)

0 1 2 3

0 1 0 0 1 0

1 1 0 1 0 1 1 0

2 0 1 0 1 0 1 0

0 1 2 3

rows: T F F T

cols: F T T F T

---

1 1 0 1 1

1 0 1 1 1

0 1 1 0 1

1 1 1 1 1

1 1 0 1 0

1 0 1 1 1

0 1 1 0 1

1 1 1 1 1

⇒

1 0 0 0 0

0 1 1 0 1 0

0 1 1 0 0 1 0

1 1 0 1 0 1 0

[1, -2, 3, -5, -6, 1]

1, -1, 2, -3, -9, -8

[1, -2, 3, -5, -6, 1]

max_sum [1, -1, 3, -2, -6, 1]

TC: O(n)

AS = O(n)

[max_sum[i] = max(arr[i], arr[i] + max_sum[i-1])]

[1, 2, 3] ⇒ [1, 2, 3, 1, 2, 3]

12