Road Trip-

-5 1 5 0 -7 - original

0 + (-5) = -5 + 1 = -4 + 5 = 1 + 0 = 0

0 + (-7) = -7

[0, -5, -4, 1, 0, -7] → prefix

TC → O(n)

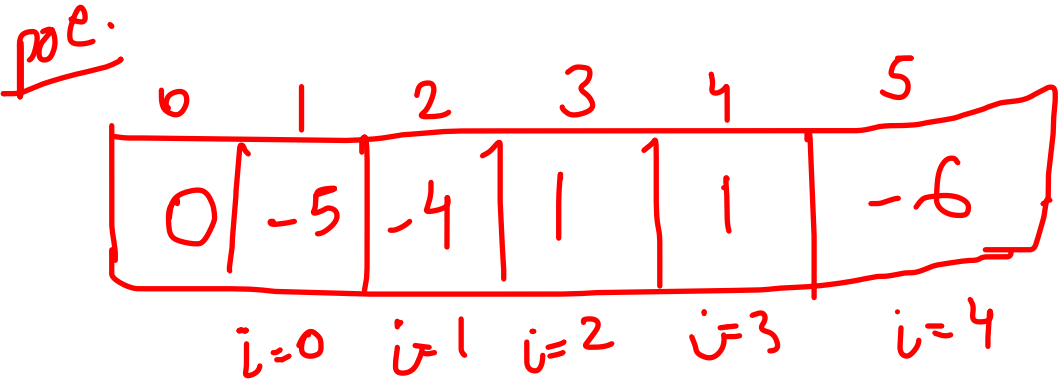
SC → O(n)

```java
public static int altitude(int arr[], int n){

    int prefixSum[] = new int[n+1];
    prefixSum[0] = 0;

    for(int i = 0; i <n; i++){
        prefixSum[i+1] = prefixSum[i] + arr[i];
    }

    // find highest altitude in prefix

    int highest = prefixSum[0];
    for(int i = 0; i <= n; i++){
        if(prefixSum[i] > highest){
            highest = prefixSum[i];
        }
    }
    return highest;
}
```

n

n

[-5, 1, 5, 0, -7] n = 5
 0 1 2 3 4



high = 1

0 > 0 X

-5 > 0 X

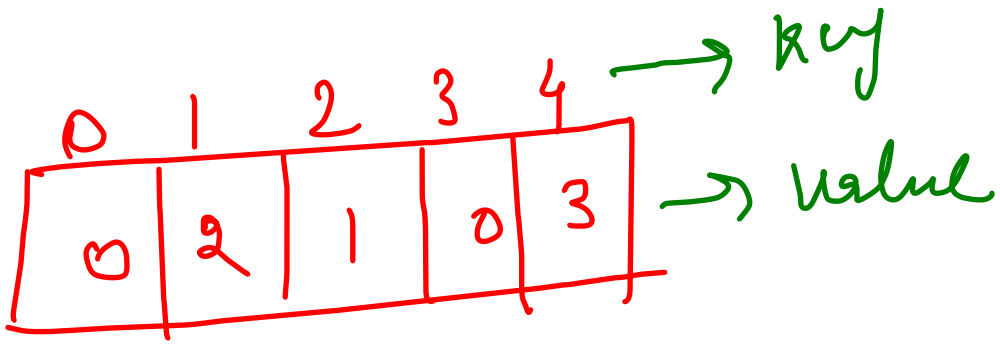
-4 > 0 X

1 > 0 ✓

1 > 1 X

-6 > 1 X

Arrays as Hashmaps / Frequency arrays.



K	V
0	0
1	2
2	1
3	0
4	3

1234

In case of numbers. → size 10 (0-9)

strings → size 256

/

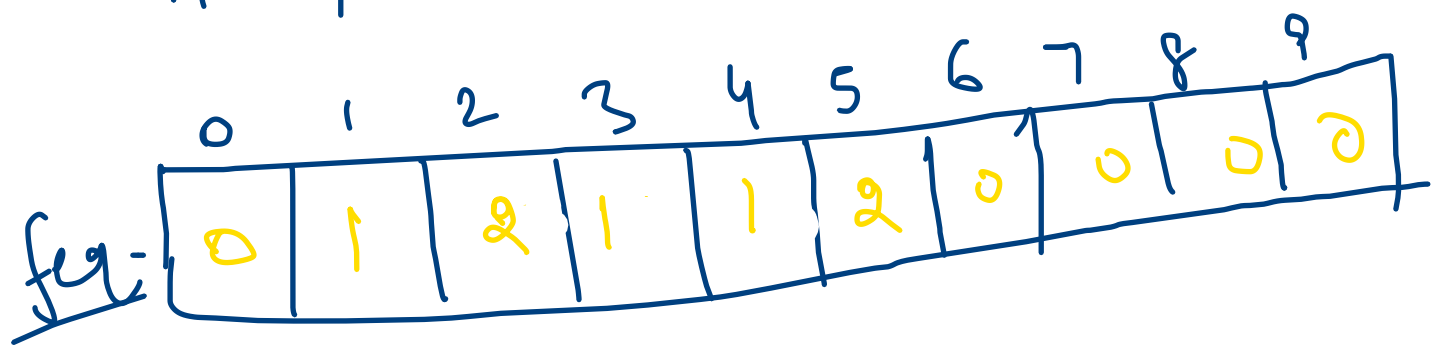
(alphabets) → size 26

```java
Scanner s = new Scanner(System.in);
int n = s.nextInt();

int freq[] = new int[10];

// calculate freq of each no.
while(n > 0){
    int digit = n % 10;
    freq[digit]++;
    n /= 10;
}

// digit with max freq

int maxfreq = 0;
int maxfreqdigit = 0;
for(int i = 0; i < 10; i++){
    if(freq[i] > maxfreq){
        maxfreq = freq[i];
        maxfreqdigit = i;
    }
}
System.out.println(maxfreqdigit);
}
```

TC - O(n)

SC - O(n)

n = 223554



0 > 0 X

1 > 0 ✓

2 > 1 ✓

1 > 2 X

2 > 2 X

0 > 2 X

maxfreq = 0 → 2

maxdigit = 0 → 2

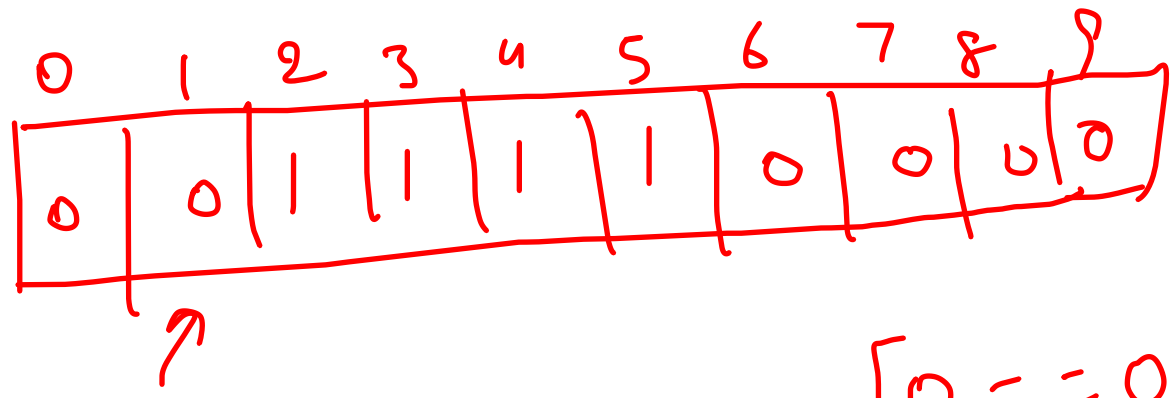
o/p → 2.

Lucky Number -

```
    int freq[] = new int[10];

    for(int i = 0; i < n; i++){
        freq[arr[i]]++;
    }

    int largest = -1;
    for(int i = 1; i < 10; i++){
        if(freq[i] == i && i > largest){
            largest = i;
        }
    }
    Syso(largest);
}
```

largest = -1

arr [3, 2, 5, 4]



freq[1] = 0 == 1 X

1 == 2 X

1 == 3 X

[0 == 0 ✓

0 > -1 ✓