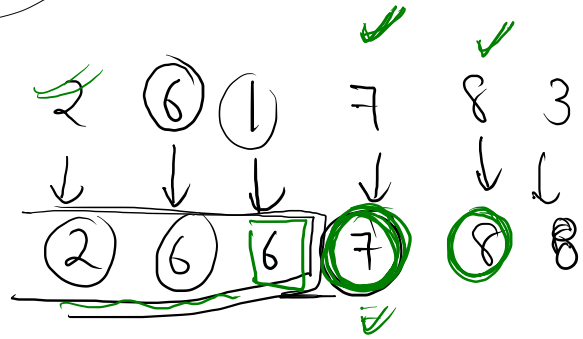
Revision.

2 pointers to prefix array.

- Target Sum
  Reach Target
  Count boats
  } i, j
- reverse.
- rotate

* 3 sum
* 4 sum



i/p → X.

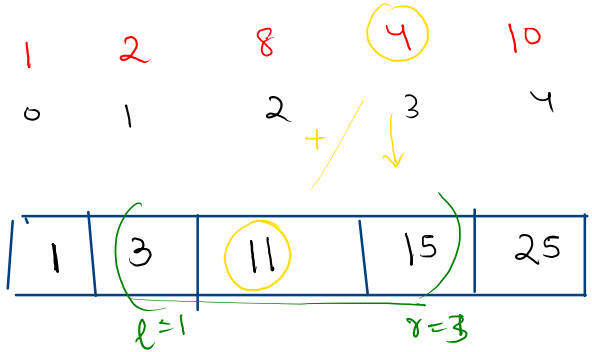
Print Prefix Sum between L and R

Sample Input 0

```
5
1
2
8
4
10
1
3
```

Sample Output

```
3
11
15
```



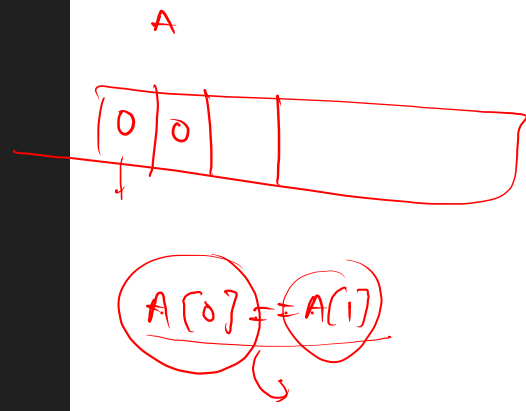
l = 1
r = 3

1. prepare prefix-sum array.
sum till me.

2. print in l to r range

l=1
r=4
↳ 3 11 15 25

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [] A = new int[n];

    for(int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }

    int L = scn.nextInt();
    int R = scn.nextInt();

    //prepare prefix sum array
    int [] prefix = new int[n];
    prefix[0] = A[0];

    for(int i = 1; i < n; i++){
        prefix[i] = prefix[i-1] + A[i];
    }

    //print in range
    while(L <= R){
        System.out.println(prefix[L]);
        L++;
    }
}
```

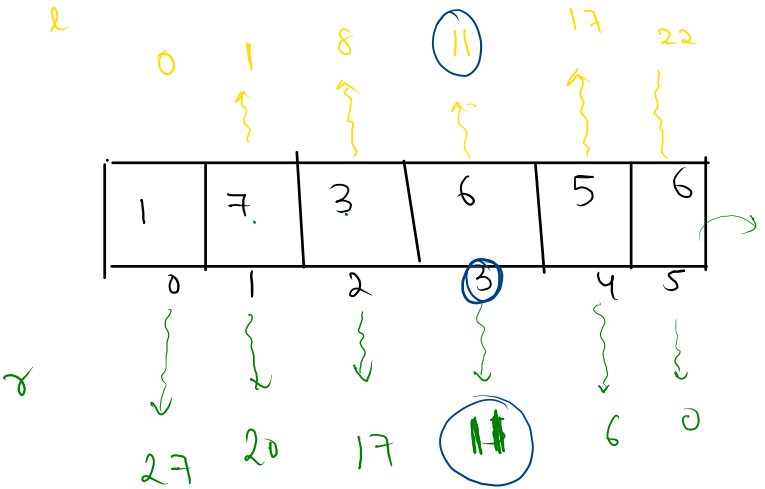


Find Pivot Index.

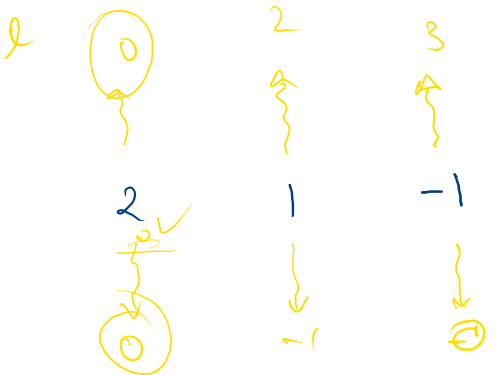
```
6
1 7 3 6 5 6
```

Sample Output 0

```
3
```



eg-2.



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int [] A = new int[n];

    for(int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }

    int [] left = new int[n];
    left[0] = 0;
    for(int i = 1; i < n; i++){
        left[i] = left[i-1] + A[i-1];
    }

    int [] right = new int[n];
    right[n-1] = 0;
    for(int i = n-2; i >= 0; i--){
        right[i] = right[i+1] + A[i+1];
    }

    //compare
    int ans = -1;
    for(int i = 0; i < n; i++){
        if(left[i] == right[i]){
            ans = i;
            break;
        }
    }
    System.out.println(ans);
}
```

Print Freq of Alphabet in String

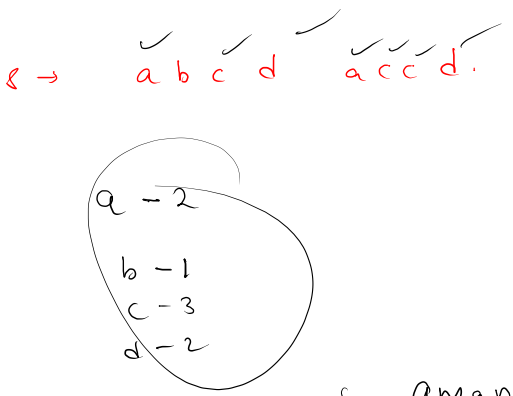
Array has Hashmap.

Sample Input 0

```
abcdaccd
```

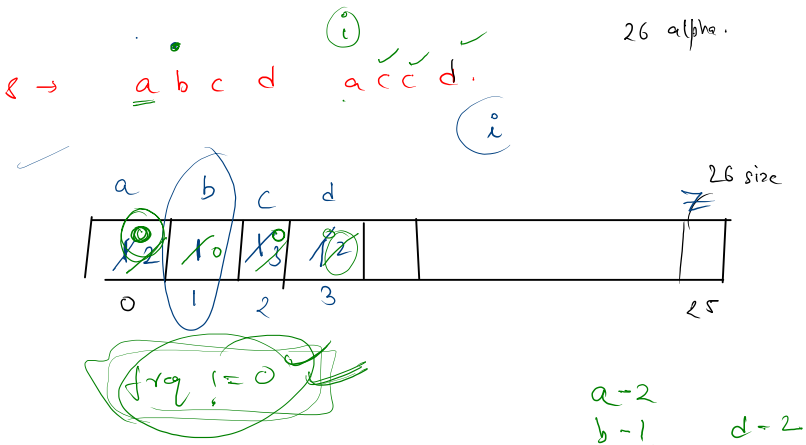
Sample Output 0

```
a-2
b-1
c-3
d-2
```



s → aman
a → 2
m → 1
n → 1

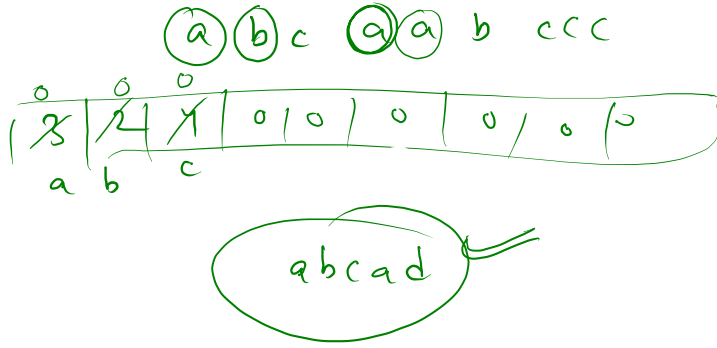
26 alpha.



a-2
b-1
c-3

d-2
a-2

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String s = scn.next();
    //freq arr
    int n = s.length();
    int [] freq = new int[26];

    for(int i = 0; i < n; i++){
        char ch = s.charAt(i);

        freq[ch - 'a']++;
    }

    // print
     for(int i = 0; i < n; i++){
        char ch = s.charAt(i);
         if(freq[ch-'a'] != 0){
             System.out.println(ch+ "-" + freq[ch-'a']);
             freq[ch-'a'] = 0;
         }
     }

}
```



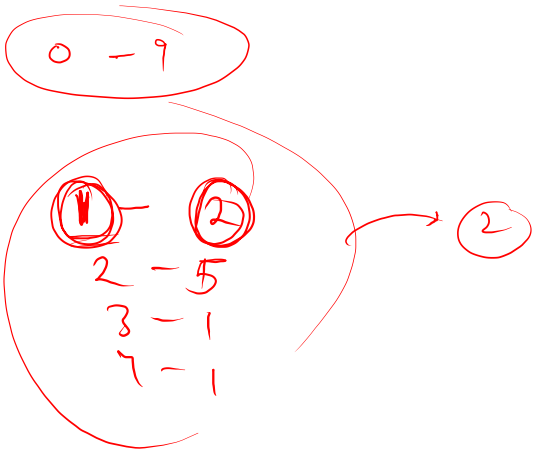
Int with Maximum Freq

Mark is a data analyst who is trying to analyze the customer data of a retail company. One of the tasks he needs to perform is to find the most common digit in the customer IDs. The IDs are represented as an array of **single-digit** integers from **0-9**. Mark needs to find the **digit that occurs the most in the array** in order to identify patterns in customer behavior.

5
1 1 2 3 4

Sample Output 0

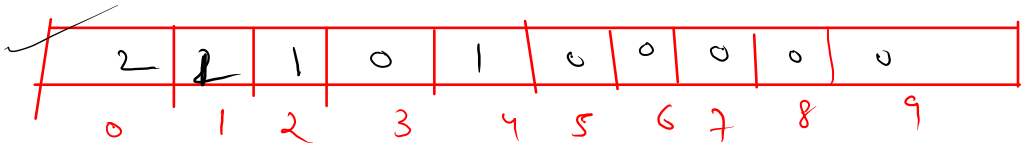
1



0 - 9 → sigle

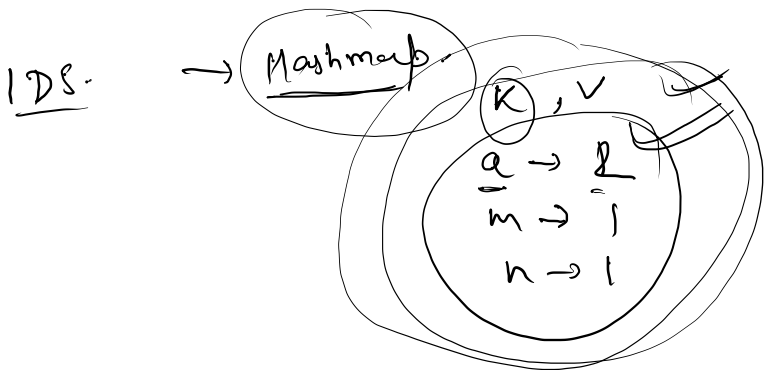
1 0 0 2 1 4

freq:



```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int A [] = new int [n];

    for (int i = 0; i < n; i++){
        A[i] = scn.nextInt();
    }

    //freq arr
    int [] freq = new int[10];
    for(int i = 0; i < n; i++){
        int ele = A[i];
        freq[ele]++;
    }

    int maxFreq = Integer.MIN_VALUE;
    int maxFreqVal = -1;

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

}
```

Array as Hashmap.



K , V

character , freq