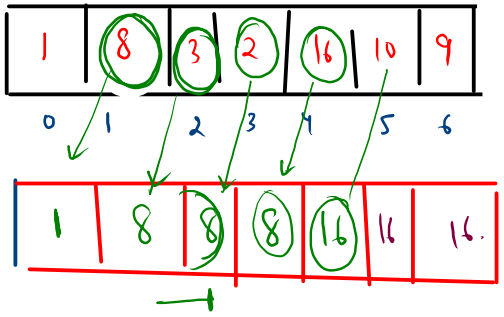
Greatest Till Me

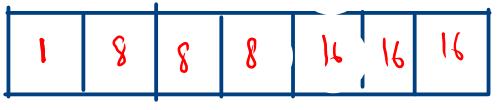
Find the maximum from left till that index

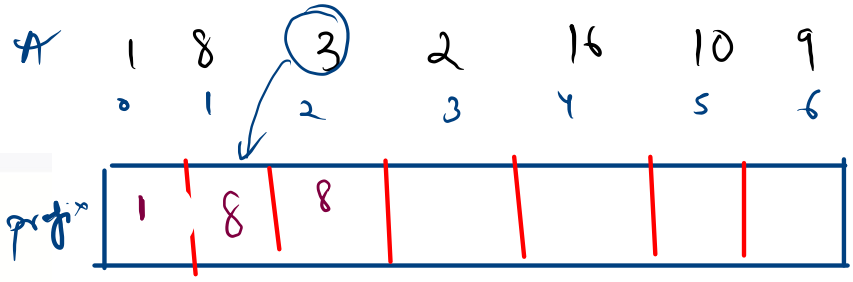
Problem Statement

Make a prefix array of size **N** such that at the **kth** index of the prefix array store the greatest element from the left till the **kth** index of the given array.

```
7
1
88
3
2
16
10
9
```








```java
import java.util.*;
public class Main {
    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 [] prefix = new int[n];
        prefix[0] = A[0];

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

        for(int i = 0; i < n; i++){
            System.out.println(prefix[i]);
        }
    }
}
```

prefix[i] = (A[i], prefix[i-1])

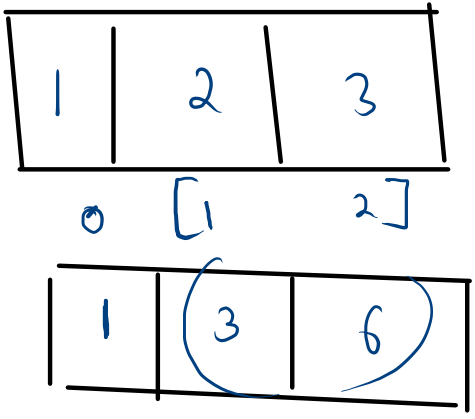
prefix[2] = 8,3

Print Prefix Sum between L and R

Given array prepare a prefix sum array and print the idxs from L to R.

Problem Statement

Take an integer input **l** and **r** such that **l,r<array.length**. Given an array. Make a prefix sum array from this. Then print the sum of the elements inside the array starting from the **l-index** till the **r-index(l and r both inclusive)**.



Test Case 1

Input:

```
3
1
2
3
1
2
```

Output:

```
3
6
```

Test Case 2

Input:

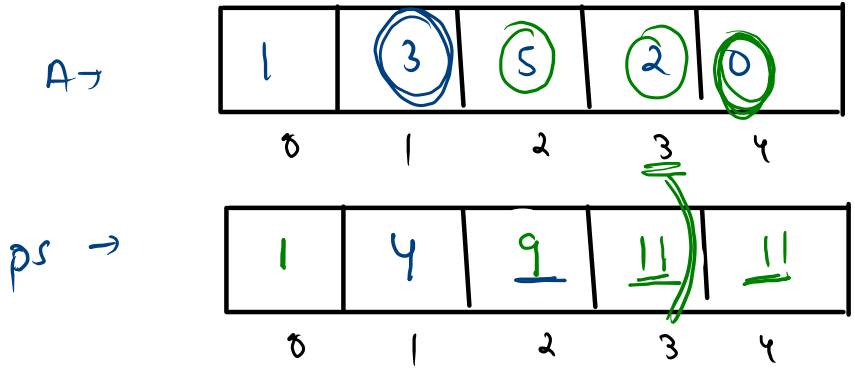
```
5
1
3
5
2
0
0
4
```

5 → N; 1, 3, 5, 2, 0 → ele; 0 → l; 4 → r

Output:

```
1
4
9
11
11
```

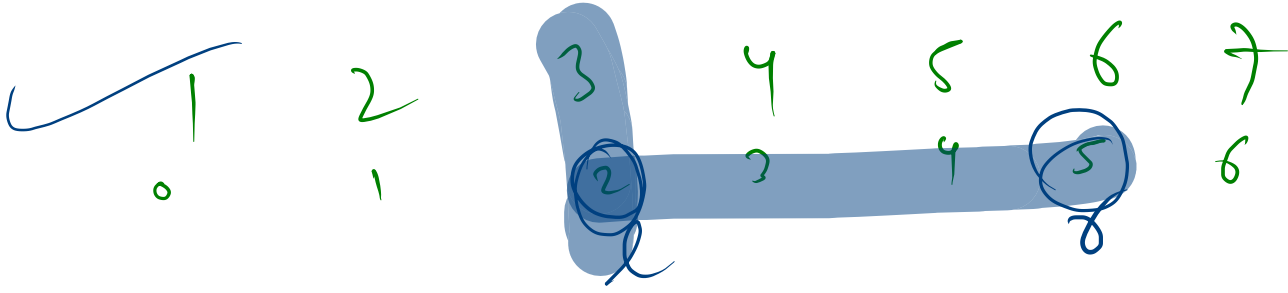
ps[i] = sum of elements of A till i idx



l = 0

r = 4

ps[i] = A[i] + ps[i-1]



$l = 2$

$r = 5$

```java
import java.util.*;

public class Main {
    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 = scn.nextInt();
        int right = scn.nextInt();

        //prefix sum array -> ps array
        int [] ps = new int[n];
        ps[0] = A[0];
        for(int i = 1; i < n; i++){
            ps[i] = A[i] + ps[i-1];
        }
        //left to right print
        for(int i = left; i <= right; i++){
            System.out.println(ps[i]);
        }
    }
}

```

Find Pivot Index

Find Pivot Index

Problem Statement

Given an array of integers **nums**, calculate the pivot index of this array.

The **pivot index** is the index where the sum of all the numbers strictly to the left of the index is equal to the sum of all the numbers strictly to the index's right.

If the index is on the **left** edge of the array, then the **left** sum is **0** because there are no elements to the left. This also applies to the right edge of the array.

Return the leftmost pivot index. If no such index exists, **return -1**.

Test Case 1

Input:

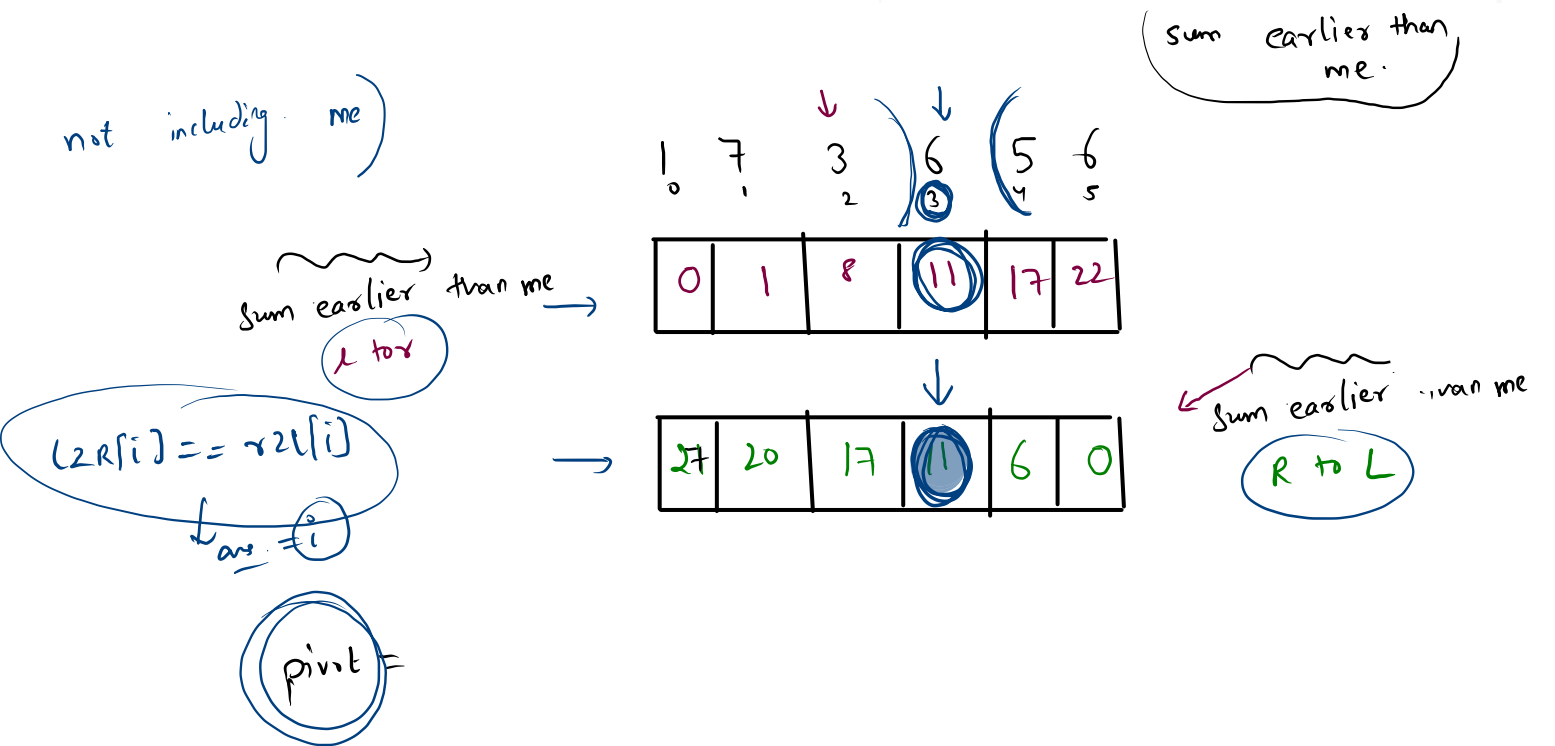
```
6
1 7 3 6 5 6
```

Output:

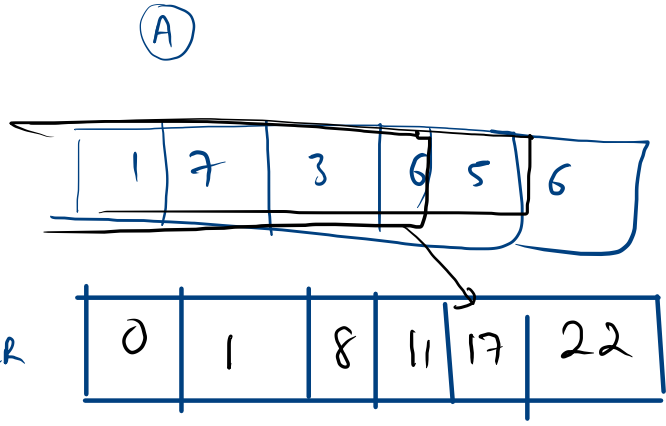
```
3
```

Explanation:

The pivot index is 3. Left sum = nums[0] + nums[1] + nums[2] = 1 + 7 + 3 = 11 Right sum = nums[4] + nums[5] = 5 + 6 = 11

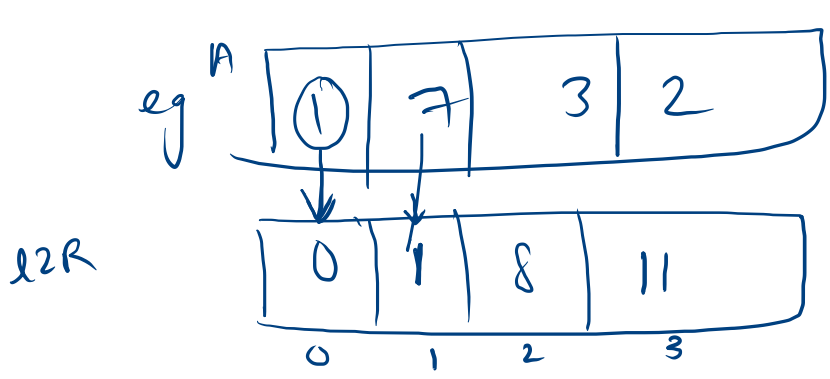


```
4
5       public static void main(String[] args) {
6           Scanner scn = new Scanner(System.in);
7           int n = scn.nextInt();
8           int [] A = new int[n];
9           for(int i = 0; i < n; i++){
10              A[i] = scn.nextInt();
11          }
12          int [] l2r = findL2R(A);
13          int [] r2l = findR2L(A);
14
15          int pivot = find(l2r, r2l );
16          System.out.println(pivot);
17      }
18  }
19
```



```java
public static int [] findL2R(int [] A){
    int [] l2r = new int[A.length];
    l2r[0]=0;
    for(int i = 1 ; i < A.length; i++){
        l2r[i] = A[i-1] + l2r[i-1];
    }
    return l2r;
}
```

2 mins.

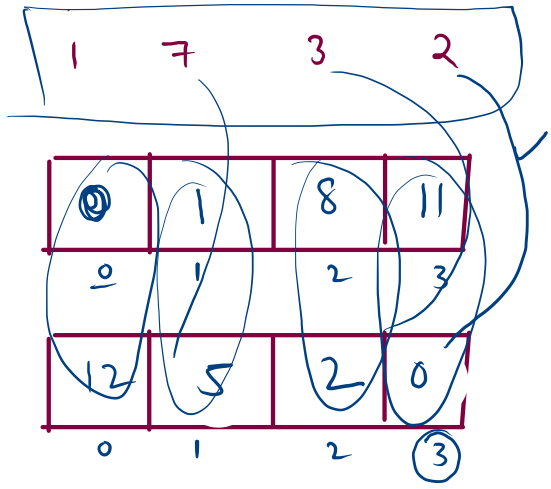


l2r ?

```java
public static int [] findL2R(int [] A){
    int [] l2r = new int[A.length];
    l2r[0]=0;
    for(int i = 1 ; i < A.length; i++){
        l2r[i] = A[i-1] + l2r[i-1];
    }
    return l2r;
}

public static int [] findR2L(int [] A){
    int [] r2l = new int[A.length];
    r2l[A.length-1] = 0;
    for(int i = A.length-2 ; i>=0 ; i--){
        r2l[i] = A[i+1] + r2l[i+1];
    }
    return r2l;
}
```

l2r

r2l



−1

A

```java
import java.util.*;

public class Main {
    public static int [] findL2R(int [] A){
        int [] l2r = new int[A.length];
        l2r[0]=0;
        for(int i = 1 ; i < A.length; i++){
            l2r[i] = A[i-1] + l2r[i-1];
        }
        return l2r;
    }

    public static int [] findR2L(int [] A){
        int [] r2l = new int[A.length];
        r2l[A.length-1] = 0;
        for(int i = A.length-2 ; i>=0 ; i--){
            r2l[i] = A[i+1] + r2l[i+1];
        }
        return r2l;
    }

    public static int find(int [] l2r, int [] r2l ){
        for(int i = 0; i < l2r.length; i++){
            if(l2r[i] == r2l[i]){
                return i;
            }
        }
        return -1;
    }

    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 [] l2r = findL2R(A);
        int [] r2l = findR2L(A);

        int pivot = find(l2r, r2l );
        System.out.println(pivot);
    }
}
```

Print Freq of Alphabet in String

Problem | Submissions | Leaderboard | Discussions

John is a software engineer who is passionate about programming. One day, he stumbled upon a challenging problem in an online coding platform. The problem required him to find the **frequency** of each alphabet in a given **string** and print the frequency of each alphabet present in the string.

help John and write a program that return the frequency of each element of string using **array as hashmap**.

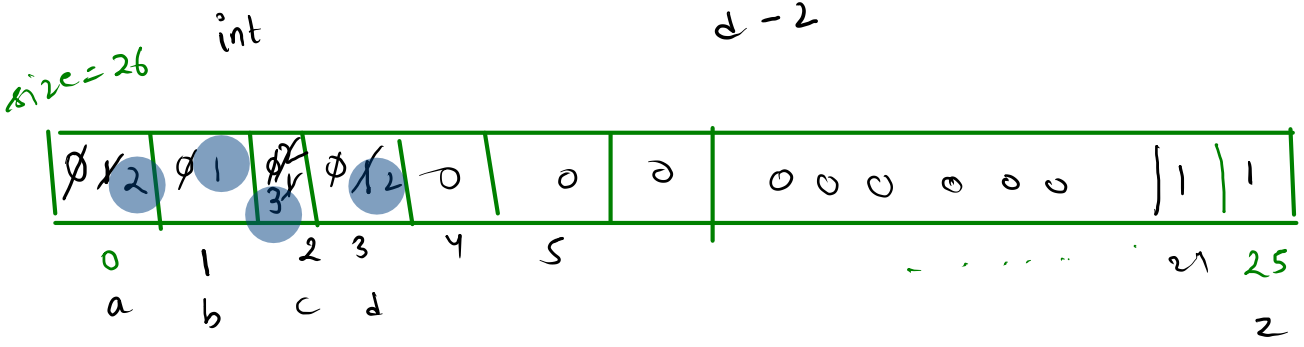
Sample Input 0

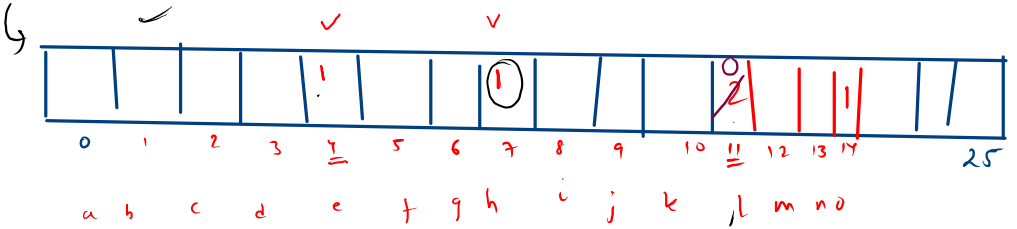
```
abcdaccd
```

Sample Output 0

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

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





h - 1

e - 1

l - 2

l - 2

`l' - `a'

?

ch - 'a' = idx

X

e - 1

h - 1

l - 2

o - 1

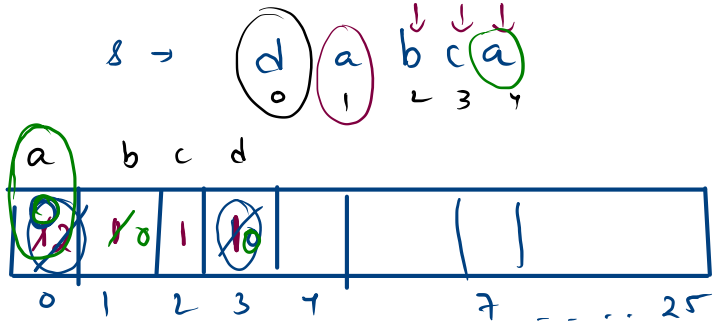
h - 1

e - 1

l - 2

o - 1

```java
import java.io.*;
import java.util.*;

public class Solution {

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


        for(int i = 0; i < s.length(); i++){
            char ch = s.charAt(i);
            freq[ch - 'a']++;
        }

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

    }
}
```



ch = ~~a~~ ~~b~~ c

ch - 'a'

99 - 97

97 - 97

freq[3]

d-1

a-2

b-1

c-1