Print Prefix Sum between L and R

$n = 8$, $l = 2$, $r = 6$

arr

1	3	2	-1	4	0	7	-2
0	1	2	3	4	5	6	7

pre

1	4	6	5	9	9	16	14
0	1	2	3	4	5	6	7

psudo code

- define prefix array
- 0^{th} ele of prefix array is 0^{th} ele. of arr
- iterate from 1 to n-1
  - pre[i] = arr[i] + pre[i-1]
- iterate from left to right on prefix array
  - print ele.

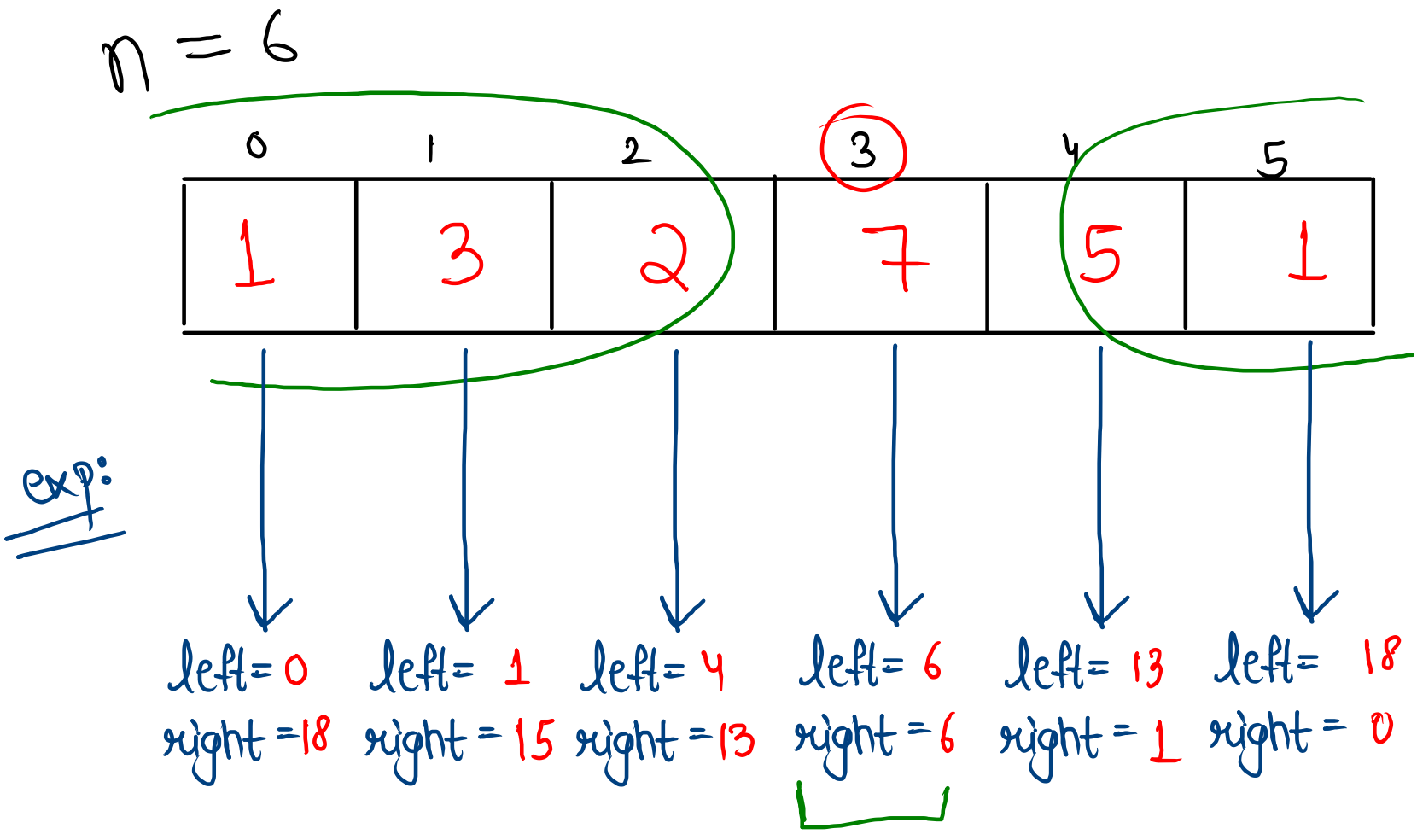
code

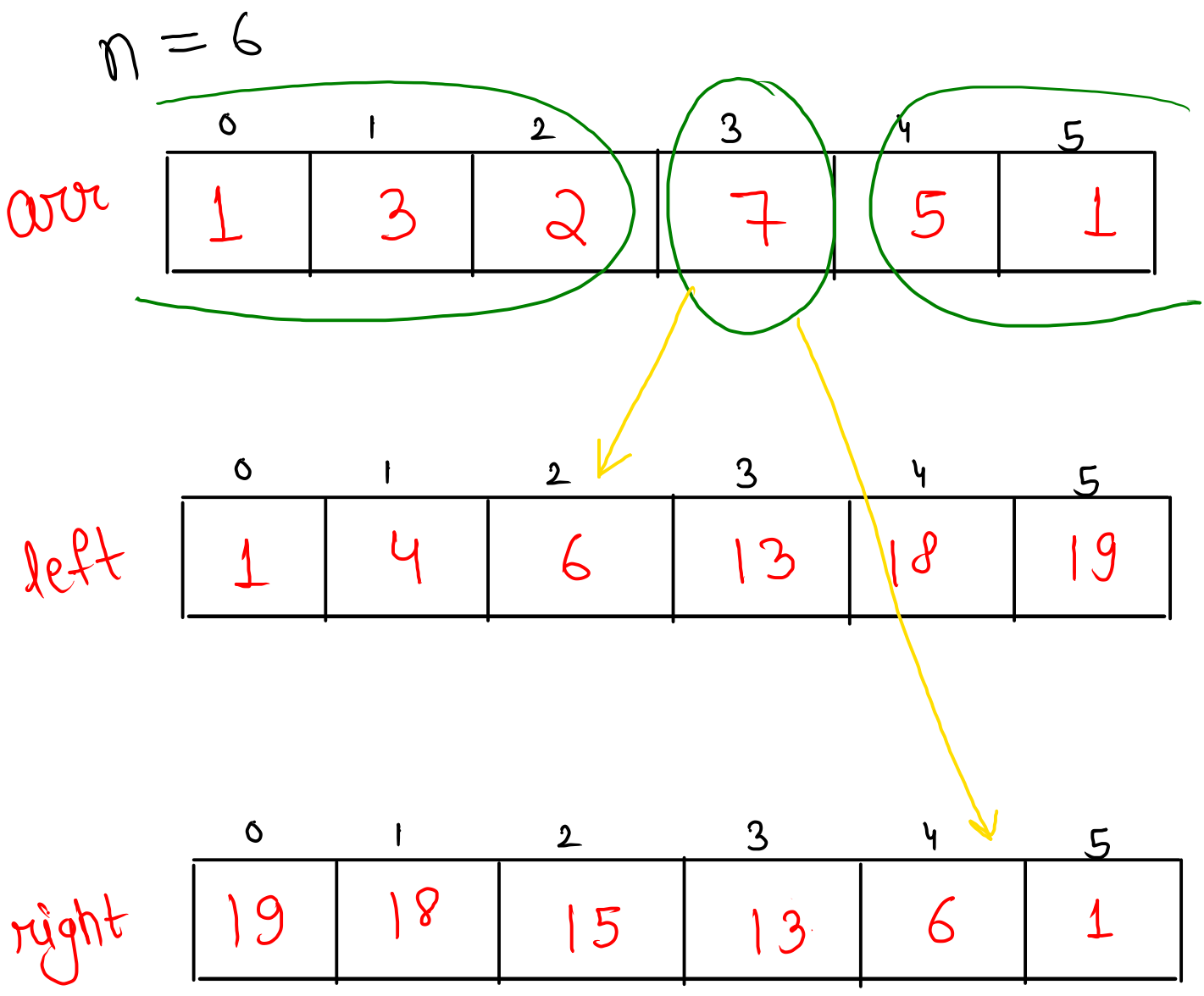
```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    int left = scn.nextInt();
    int right = scn.nextInt();

    preSumFromLeftToRight(arr, n, left, right);
}

public static void preSumFromLeftToRight(int[] arr, int n, int left, int right) {
    int[] pre = new int[n];
    pre[0] = arr[0];
    for (int i = 1; i < n; i++) {
        pre[i] = arr[i] + pre[i - 1];
    }

    for (int i = left; i <= right; i++) {
        System.out.println(pre[i]);
    }
}
```

T.C = O(N)

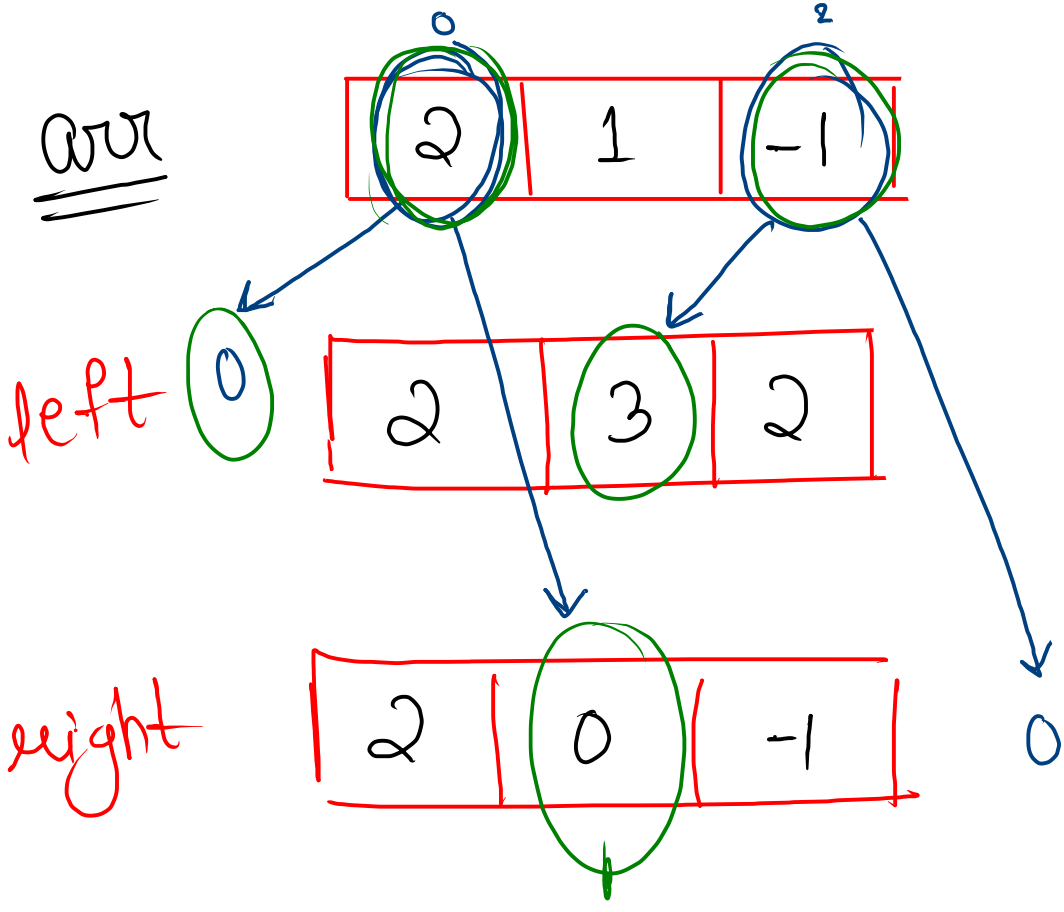
S.C = O(N)

Find Pivot Index 1



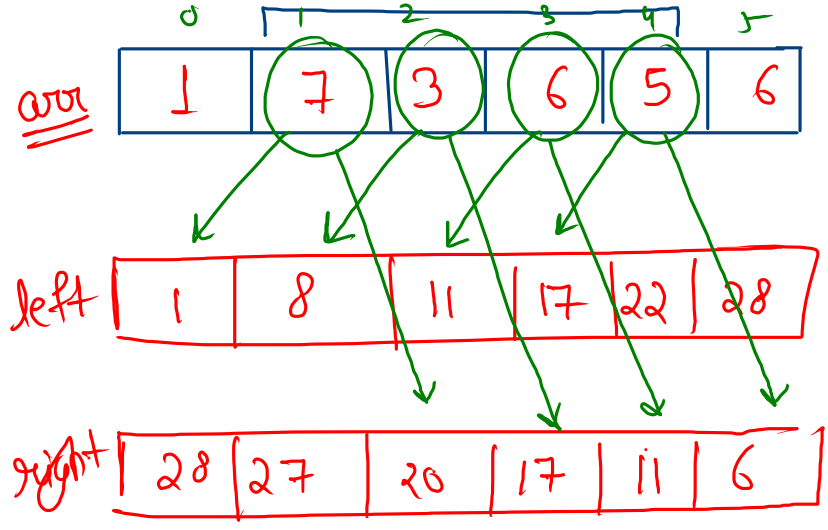

n = 6
arr
0 1 2 3 4 5
1 3 2 7 5 1
left
0 1 2 3 4 5
1 4 6 13 18 19
right
0 1 2 3 4 5
19 18 15 13 6 1

ans[i] ⇒ if left[i-1] == right[i+1]



Code

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    int n = scn.nextInt();
    int[] arr = new int[n];
    for (int i = 0; i < n; i++) {
        arr[i] = scn.nextInt();
    }
    System.out.println(findPivot(arr, n));
}

public static int findPivot(int[] arr, int n) {
    int[] left = new int[n];    // prefix sum array
    left[0] = arr[0];
    for (int i = 1; i < n; i++) {
        left[i] = arr[i] + left[i - 1];
    }

    int[] right = new int[n];   // suffix sum array
    right[n - 1] = arr[n - 1];
    for (int i = n - 2; i >= 0; i--) {
        right[i] = arr[i] + right[i + 1];
    }

    if ( n > 1 && right[1] == 0 ) return 0;

    if ( n > 1 && left[n - 2] == 0 ) return n - 1;

    for (int i = 1; i <= n - 2; i++) {
        if ( left[i - 1] == right[i + 1] ) {
            return i;
        }
    }

    return -1;
}
```



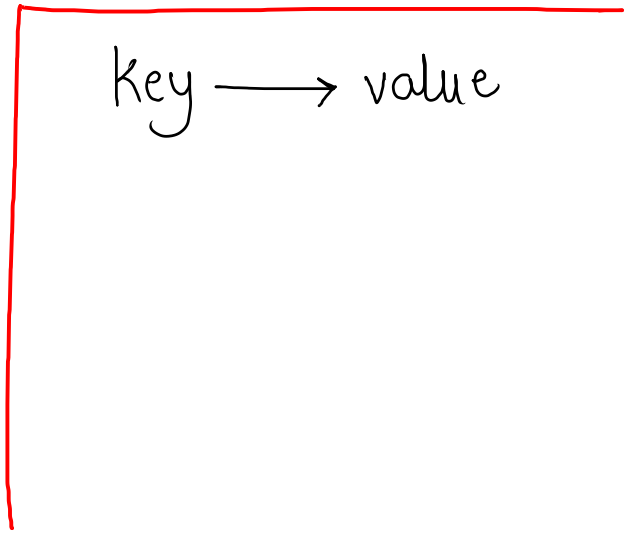
⇒ Arrays as Hashmap

↳ It is a data structure which stores data in form of key and value

hashmap

String → Integer

key → value



hashmap

10^6 MLE → memory limit exceed

String → Integer

"India" → 459

"SL" → 55

"Aus" → 200

"SA" → 350

"Pak" → 0

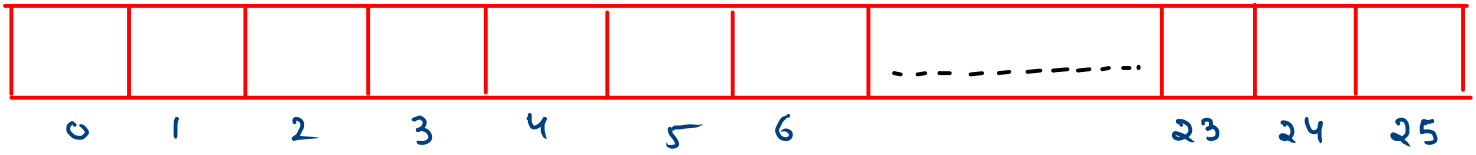
Note:-

↳ key should be unique and value can repeat

↳ Unordered (Indexing is not present in HM)

Print Freq of Alphabet in String

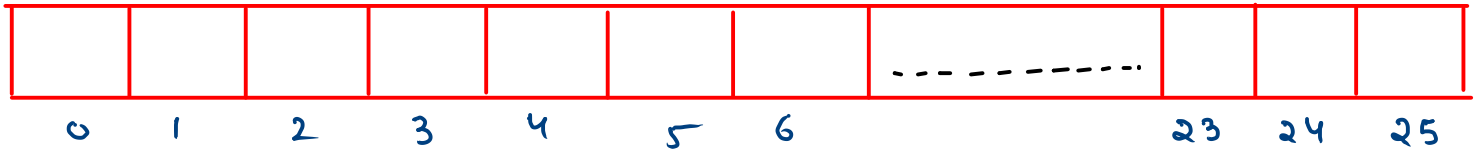
Str = " a b c a c b d e a b d "



a → 3
b → 3
c → 2
d → 2
e → 1

Note:- if range of input is smaller, then we can use array as hashmap

Str = " a b c a c b d e a b d "



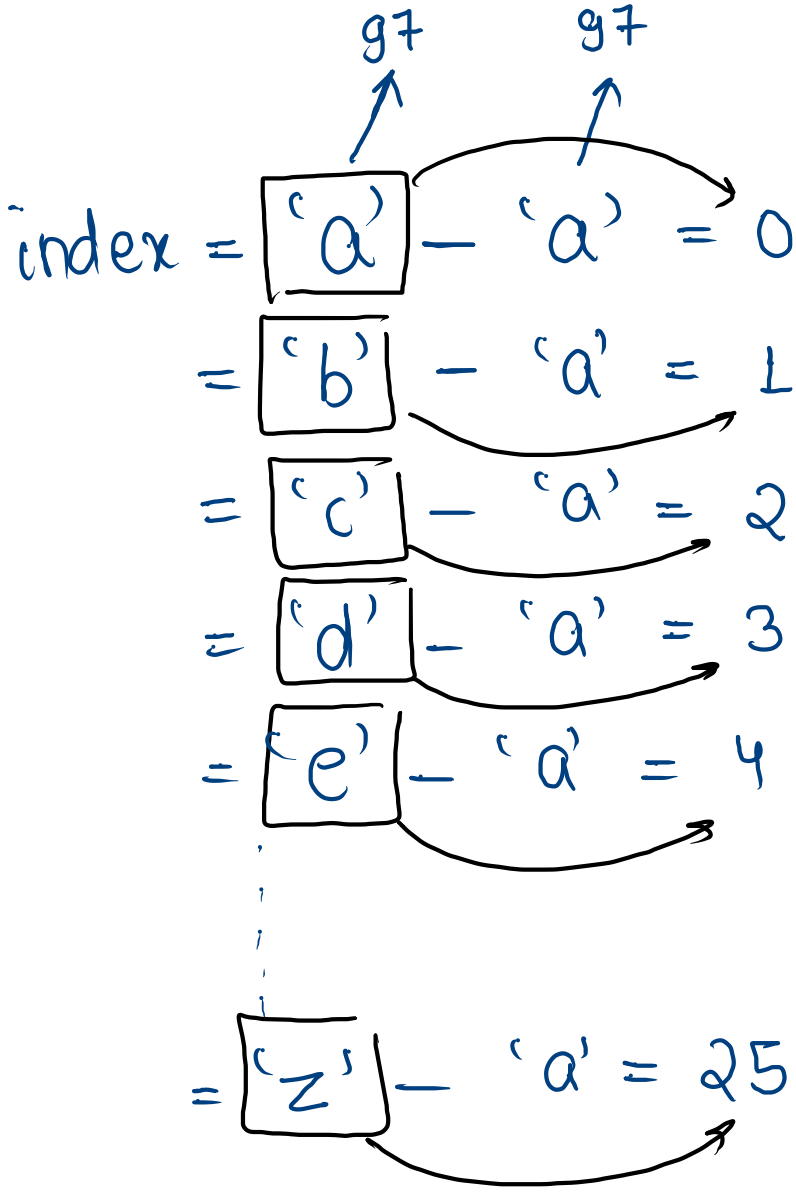
map indices with char

Goal

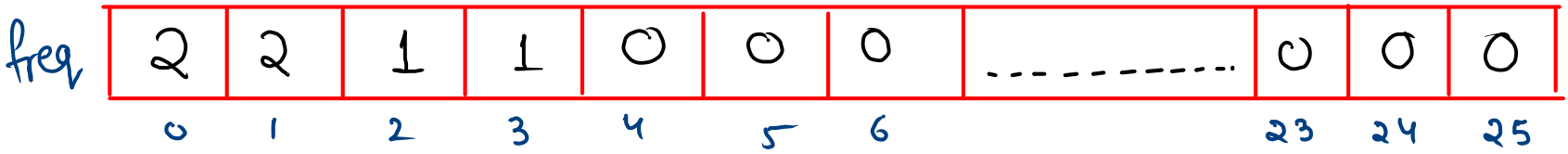
0 → 'a'
1 → 'b'
2 → 'c'
3 → 'd'
.
.
.
25 → 'z'

ASCII

'a' → 97
'b' → 98
'c' → 99
'd' → 100
.
.
.
'z' → 122


g7
g7
index = 'a' - 'a' = 0
= 'b' - 'a' = 1
= 'c' - 'a' = 2
= 'd' - 'a' = 3
= 'e' - 'a' = 4
= 'z' - 'a' = 25

str = "abcdab"



1) ch = 'a'
idx = 'a' - 'a' = 0

2) ch = 'b'
idx = 'b' - 'a' = 1

3) ch = 'c'
idx = 'c' - 'a' = 2

4) ch = 'd'
idx = 'd' - 'a' = 3

5) ch = 'a'
idx = 'a' - 'a' = 0

6) ch = 'b'
idx = 'b' - 'a' = 1

```java
public static void main(String[] args) {
    Scanner scn = new Scanner(System.in);
    String str = scn.next();
    printFreq(str);
}

public static void printFreq(String str) {
    int n = str.length();
    int[] freq = new int[26];
    for (int i = 0; i < n; i++) {
        char ch = str.charAt(i);
        int idx = ch - 'a';
        freq[idx]++;
    }

    for (int i = 0; i < 26; i++) {
        System.out.print(freq[i] + " ");
    }
}
```