Q → **Search for a Range**

Given a sorted integer array find the left most & right most index of the given integer B. If not found return [-1, -1]

	0	1	2	3	4	5	6	7	8	9
A = [	1	1	5	5	5	5	8	10	10	10]

B = 5

Ans = [2 5]

Binary Search

```
ans[0] = ans[1] = -1
l = 0      r = N-1
while (l <= r) {
    mid = (l + r) / 2
    // First occurence
    if (A[mid] == B && (mid == 0 || A[mid-1] < B))
            ans[0] = mid
            break
    }
    if (A[mid] < B)   l = mid + 1
    else      r = mid - 1
}

l = 0      r = N-1
while (l <= r) {
    mid = (l + r) / 2
    // last occurence
    if (A[mid] == B && (mid == N-1 || A[mid+1] > B))
            ans[1] = mid
            break
```

```
    }
    if (A[mid] <= B)   l = mid + 1
    else               r = mid - 1
}
return ans
```

Q → <u>King & Palindromes</u>

Find the length of longest palindrome that could be made by letters of the given string.

Eg → "banana" ⟶ "anana"

Ans = 5

'a' → 2
'b' → 4

a b b b b a → 6

'a' → 5 − 4 = 1
'b' → 1

a a a a a → 5 (b)

```
// Frequency array of length = 26
for i → 0 to (N-1) {
    ch = A[i]
    F[ch - 'a']++
}
cnt = 0
add = true
```

```
for i → 0 to 25 {
    if (F[i] % 2 == 0)
        cnt += F[i]
    else {
        cnt += F[i] - 1
        if (odd) { cnt++
                   odd = false }
    }
}
return cnt
```

'a' → 4 aa aa

'b' → 5

'c' → 1

TC = O(N) SC = O(26) → O(1)

Q → Decreasing order words

Arrange the words in descending order of length, if two words are of equal length, arrange them in original order.

Stable Sort

A = ["hi", "he", "hello"]

↳ ["hello" "hi" "he"]

```
Arrays.sort(A, (String u, String v) → {
        ul = u.length()
        vl = v.length()
        return (vl - ul)
});
```