HW_Length of Longest Palindrome

length = 0 + 2 + 4 + 2

```java
public static int longest(String str){

    HashMap<Character, Integer> mp = new HashMap<>();

    for(char c : str.toCharArray()){
        mp.put(c, mp.getOrDefault(c, 0) + 1); // freq
    }

    int length = 0;
    boolean odd = false;

    for(int count : mp.values()){
        if(count % 2 == 0){
            length += count;
        }
        else{
            length += count - 1;
            odd = true;
        }
    }
    if(odd){
        length += 1;
    }
    return length;
}
```

O(n)

a = 2

b = 2 | 3

c = 2

d = 4

[abccccdd]

a = 1, b = 1, c = 4, d = 2

length = 0 + 4 + 2 = 6 + 1 = 7

odd = ft

decacCd ~ 7

dccbccd ~ 7

HW_Find the missing number

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

Set<Integer> numbers = new HashSet<>();

for(int i = 0; i < n; i++){
    numbers.add(s.nextInt());
}

boolean missing = false;
StringBuilder sb = new StringBuilder();

for(int i = l; i <= r; i++){
    if(!numbers.contains(i)){
        missing = true;
        sb.append(i).append(" ");
    }
}
if(missing){
    System.out.print(sb.toString().trim());
}
else{
    System.out.println("-1");
}
}
```

O(n)

HW_Permutation in String 2

2-3-4-

0 1 2 3 4 5 6

ab - str1 (2)
0 1

eidbaooo - str2 (8)
0 1 2 3 4 5 6 7

a, b == e, i X

a, b == e i d = 2-2=0

a, b == i d b = 3-2=1

a, b == d b a = 4-2=2

true.

```java
public static boolean check(String s1, String s2){

    if(s1.length() > s2.length()){
        return false;
    }

    Map<Character, Integer> s1map = new HashMap<>();
    Map<Character, Integer> s2map = new HashMap<>();

    for(char c : s1.toCharArray()){
        s1map.put(c, s1map.getOrDefault(c, 0) + 1);
    }

    for(int i = 0; i < s1.length(); i++){
        char c = s2.charAt(i);
        s2map.put(c, s2map.getOrDefault(c, 0) + 1);
    }
    if(s1map.equals(s2map)){
        return true;
    }
```

S1 = ab , S2 = eidbaooo
(2) (4)

s1map = {a:1, b:1}

s2map = {e:1, i:1, d:1, b:1, a:1, o:3}
0 1 2 3

a:1, b:1 == e:1, i:1 X

newChar = d:1

oldChar = 2-2 = 0

a:1, b:1 == i:1, d:1 X

a, b == d, b X

a, b == b, a ✓ True

```java
    for(int i = s1.length(); i < s2.length(); i++){
        char newChar = s2.charAt(i);
        char oldChar = s2.charAt(i - s1.length());

        s2map.put(newChar, s2map.getOrDefault(newChar, 0) + 1);

        if(s2map.get(oldChar) == 1){  // remove
            s2map.remove(oldChar);
        }
        else{
            s2map.put(oldChar, s2map.get(oldChar) - 1);
        }

        if(s1map.equals(s2map)){
        return true;
        }
    }
    return false;
}
```

O(n)