Ques Given a string. Find the minimum and maximum distance b/w two vowels.

distance b/w indices

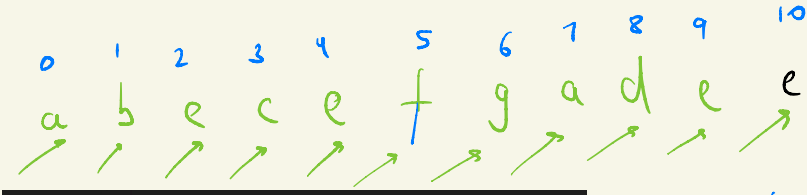
Input string ⇒ a b e c e f g a d e f

0 1 2 3 4 5 6 7 8 9 10

⇒

min distance = 2 (b/w index (7 and 9) or (2 and 4))

max distance = 9 (b/w index 0 and 9)




```
int n=str.length();
int first_occurrence=-1;
int last_occurrence=-1;

int last=-1;
int min_distance=n+1;

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

    if(isVowel(ch)==true){
        if(first_occurrence==-1){ // i have not yet dicovered any vowel
            first_occurrence=i;
        }

        last_occurrence=i;

        // finding min distance
        if(last==-1){
            last=i;
        } else {
            int distance = i - last;
            min_distance=Math.min(min_distance,distance);

            last=i;
        }
    }
}
```

first_occurrence = 0

last_occurrence = 0 2 4 7 9

10 - 9 ⇒ 1

i = 9 10 → 9

last = -1 0 2 4 7 9

min_distance = 12 2 4 1

Ques Given a string of numeric characters. Convert it to integer data type from string data type.

Input ⇒ "134"

Output ⇒ 134

Function

```
int val = Integer.parseInt(str);
```

↓ consisting of numbers

eg. my str = "345";
int val = 345;

Ques Given a numeric string. Check if it is balanced.
Sum of first half is equal to second half

string str = "2433" → true
0 1 2 3
6 6

str = "1234" → false
3 7

str ⇒ "2457335"
0 1 2 3 4 5 6
n = 7

fh = 0 ~~+6~~ 11
sh = ~~3+5~~ 11

```java
public static boolean checkBalanced(String str){
    int n=str.length();
    int first_half=0;
    int second_half=0;

    if(n%2==0){
        // calculating first half
        for(int i=0; i<n/2; i++){
            char ch=str.charAt(i);

            int val=ch-'0';
            first_half=first_half+val;
        }

        // calculating second half
        for(int i=n/2; i<n; i++){
            char ch=str.charAt(i);

            int val=ch-'0';
            second_half=second_half+val;
        }
    } else {

    }

    if(first_half==second_half){
        return true;
    } else {
        return false;
    }
}
```

$i = \left(\frac{n}{2}+1\right)$

$\frac{7}{2}+1$

$3+1 \neq 4$

'a' = 0
'a' = 1
'0' = 48
'0' - '0' = 0
'1' - '0' = 1

'0' → 0
'1' → 1

Ques Given two strings. Check if sum of ascii values of both strings are equal or not.

⇒ str1 ⇒ "abcef"

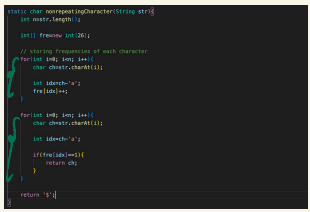
97+98+99+101+102

str2 ⇒ "bcdghi"

98+99+100+103+104+105

zxvczbtxyzvy

ch='c'



Ques Remove all the duplicate from sorted string.

Input

```
public static String removeDuplicateSorted(String str){
    String ans="";

    ans=ans+str.charAt(index: 0);

    for(int i=1; i<str.length(); i++){
        char ch=str.charAt(i);
        char ch_prev=str.charAt(i-1);

        if(ch!=ch_prev){
            ans=ans+ch;
        }
    }

    return ans;
}
```

String ans = ""

ans = ab

ch = ~~a~~ b

ch_prev = ~~a~~ a

Input → b a c c a b f g d f

ch = 'b'

```
4   public static String removeDuplicateUnsorted(String str){
5       String ans="";
6
7       int[] fre=new int[26];
8
9       for(int i=0; i<str.length(); i++){
10          char ch=str.charAt(i);
11
12          int idx=ch-'a';
13
14          if(fre[idx]==0){
15              ans=ans+ch;
16          }
17
18          fre[idx]++;
19      }
20
21      return ans;
22  }
```

ans = b a c f g d

a → ~~0~~ 1 2

b → ~~0~~ 1 2

c → ~~0~~ 1 2

d → ~~0~~ 1

f → ~~0~~ 1

g → ~~0~~ 1

ch = 'c'

idx = 2

String str = a b c d e

0 1 2 3 4

sp = 0

ep = ~~0~~ 1 2 3

b c d e

bc cd de

abc bcd cde

abcd bcde

abcde

ep = sp ; ep < n

sp < n; sp++){

```
15  public static void generateAllSubstrings(String str){
16      int n=str.length();
17
18      for(int sp=0; sp<n; sp++){
19          for(int ep=sp; ep<n; ep++){
20              String sub=makeString(str, sp, ep);
21              System.out.print(sub+" ");
22          }
23      }
24  }
```

a b c d e

0 1 2 3 4

sp = 0 + 2

ep = 2 3 4

sub = cd

```java
public static void generateAllSubstrings_better(String str){
    int n=str.length();

    for(int sp=0; sp<n; sp++){
        String sub="";
        for(int ep=sp; ep<n; ep++){
            sub=sub+str.charAt(ep);
            System.out.print(sub+" ");
        }
    }
}
```

a b c d e

a ab abc abcd abcde

b bc bcd bcde

c cd cde

d de

e

Ques Given a string, find the number of substrings with no vowels.

0 1 2 3 4 5

b a f c d e

count = 0 + 1 + 2 + 3 + 4 + 5 + 6

⇒

```java
public static int countSubstringsWithoutVowel(String str){
    int n=str.length();

    int count=0;

    for(int sp=0; sp<n; sp++){
        for(int ep=sp; ep<n; ep++){
            char ch=str.charAt(ep);

            if(isVowel(ch)){
                break;
            }
            count++;
        }
    }

    return count;
}
```

$O(n^2)$

$O(n)$

isVowel( ) → true / false

Ques Given a number string, find the sum of all the substrings.

=>

"1 2 3 4"
0 1 2 3

```java
public static int sumOfSubstrings_better(String str){
    int n=str.length();

    int ans=0;

    for(int sp=0; sp<n; sp++){
        int num=0;
        for(int ep=sp; ep<n; ep++){
            char ch=str.charAt(ep);

            int digit=ch-'0';
            num=num*10+digit;

            ans=ans+num;
        }
    }

    return ans;
}
```

sp=0
ep=sp+1
num=0+1=1
ch='1'
digit=1
num=1+2=12
ch='2'
digit=2
num=12+3=123
ch='3'
digit=3

int num=0+1*2+3

num=num*10+digit

ans=1+13+135+1370+1375+1395

ans=1+13+136

Ques Given a string, find number of substrings starting and ending with same character.

=> abab => 6

sp=0 'a'
ep=2 'a'

a
ab
aba
abab

b
ba
bab

a b

```java
public static String getSubString(String s, int idx, int len){
    String ans="";

    int si=idx;
    int ei=idx+len-1;

    for(int i=si; i<=ei; i++){
        ans=ans+s.charAt(i);
    }

    return ans;
}

public static boolean check(String str, String tar){
    String s=str+str;
    int n=str.length();

    for(int i=0; i<n; i++){
        String sub=getSubstring(s,i,n);

        if(sub.equals(tar)){
            return true;
        }
    }

    return false;
}
```

Ques

str=abcde
n=5
tar=cdeab

s=abcdeabcde
0 1 2 3 4 5 6 7 8 9

string, si, ei

→ all the characters from 'si' to 'ei-1'

```
String sub = str.substring(si, ei); // won't include ei
```

↳ O(n)

```
0 1 2 3 4
str = "abcde"
```

(0,5)

```
sub = str.substring(2, 4)
sub = "cd"

sub = str.substring(0, 3);
sub = "abc"
```

```
String sub = str.substring(si); // whole string starting from si
```

eg.

```
sub = str.substring(3);
sub = "de"
```

. contains

```
String s1 = "abcdef"
String s2 = "cde"

boolean ans = s1.contains(s2); // checks if s2 is a substring of s1
ans = true

s2 = "cdf"
ans = false
```

StringBuilder

⇒ Initial

```
StringBuilder sb = new StringBuilder();
String str = "abc";
StringBuilder sb = new StringBuilder(str);
```

(new StringBuilder() → empty)

add

```
char ch = 'a';
sb.append(ch) → O(1)
```

⇒ sb = abca

get

```
ch = sb.charAt(2);
ch = 'c';
```

length

```
len = sb.length();
```

set char At

```
sb.setCharAt(index, ch);
```

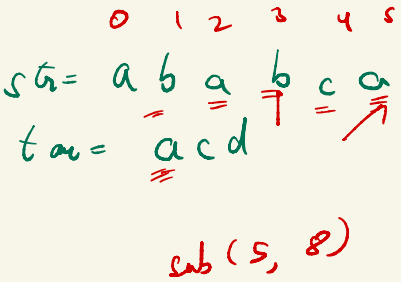
→ O(1)

delete

→ O(n)

to String : → converted to string

```
String ans = sb.toString();
```




```java
// https://www.hackerrank.com/contests/may-practice-java-dsa/challenge
public static int startingIndex(String str, String tar){
    int n=str.length();
    int tar_len=tar.length();

    for(int i=0; i<n; i++){
        if(str.charAt(i)==tar.charAt(index: 0)){
            String sub=str.substring(i, i+tar_len);

            if(sub.equals(tar)){
                return i;
            }
        }
    }

    return -1;
}
```

Ques Power of a String


```java
public static int powerOfString(String str){
    int n=str.length();

    int curr_ch=str.charAt(index: 0);
    int curr_count=1;

    int ans=curr_count;

    for(int i=1; i<n; i++){
        char ch=str.charAt(i);

        if(curr_ch==ch){
            curr_count++;
        } else {
            // ans update
            // curr_count=1
            // curr_ch update

            ans=Math.max(ans,curr_count);
            curr_ch=ch;
            curr_count=1;
        }
    }

    ans=Math.max(ans,curr_count);
    return ans;
}
```

"aaa"

ch='a'

curr-count = 3

ans = 3

Que

25 → 3^2+4^2

17 → 4^2+1^2

$i^2+j^2=n$

$0 <= n <= 10^6$

```
for ( int i = 0; i<=n; i++) {
    for ( int j=0; j<=n; j++) {
        int sum = i*i + j*j;
        if ( sum == n) return true;
    }
}
```

$i = 0+1+2+3+4+5+6$

$a^2+b^2 = n$

$b^2 = n$

$a = \sqrt{n}$

$b = \sqrt{n}$

$\sqrt{n}$ → $a=0$

$b=0$

$b=\sqrt{n}$

$a <= \sqrt{n}$

$a^2 <= n$

$a*a <= n$

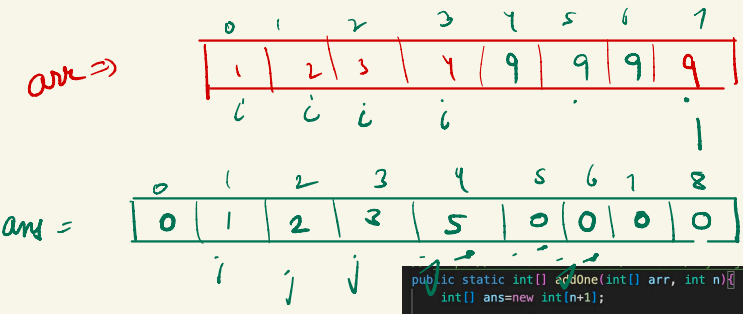
Ques

String target = "abcedef"

String str = "bef"

```
public static boolean checksubsequence(String str, St
    int n=tar.length();
    int m=str.length();

    int i=0; // to traverse in tar string
    int j=0; // to traverse in str string

    while(i<n && j<m){
        int char_i=tar.charAt(i);
        int char_j=str.charAt(j);

        if(char_i==char_j){
            i++;
            j++;
        } else {
            i++;
        }
    }

    if(j==m){
        return true;
    } else {
        return false;
    }
}
```

Ques




```
public static int[] addOne(int[] arr, int n){
    int[] ans=new int[n+1];

    int i=n-1;
    int j=n;
    int carry=0;

    while(i>=0 && j>=0){
        int sum=0;
        if(i==n-1){
            sum=arr[i]+1;
        } else {
            sum=arr[i]+carry;
        }

        if(sum==10){
            carry=1;
            sum=0;
        } else {
            carry=0;
        }

        ans[j]=sum;
        i--;
        j--;
    }

    ans[j]=carry;

    return ans;
}
```