```java
public class Main
{
    public static void main(String[] args) {

        String s = "46";           //string to int
        int val = Integer.parseInt(s);

        

        System.out.println(val + 10);
    }
}

```

Sum of All Substrings

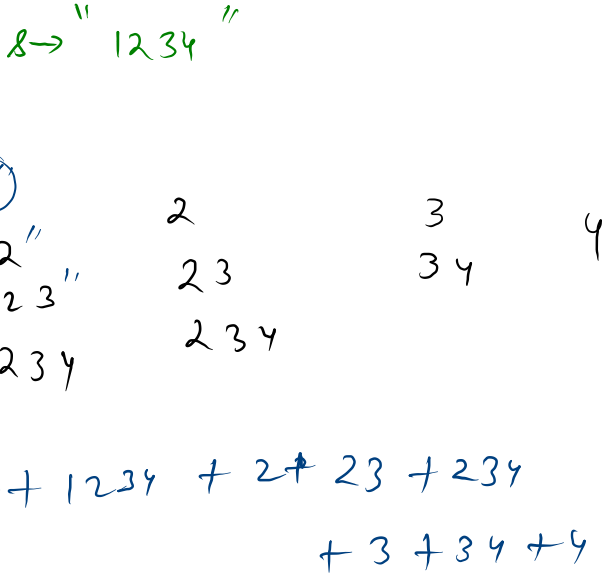
Problem | Submissions | Leaderboard | Discussions

Take a **String str** as input and print the **sum** of all **substrings** of a string representing a **number**.

Eg: str="**1234**"

Sum = 1 + 2 + 3 + 4 + 12 + 23 + 34 + 123 + 234 + 1234 = **1670**

Note: Number will be in string fromat.



```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 ans = 0;

        for(int start = 0; start < s.length(); start++){
            for(int end = start; end < s.length(); end++){
                ans += Integer.parseInt(s.substring(start, end+1));
            }
        }
        System.out.println(ans);
    }
}
```

Desired String

Problem | Submissions | Leaderboard | Discussions

Take a **string** as input. Print the **count** of all the **substrings** that start with **'A'** and end with **'A'**. Also print the **length** of the **longest** such substring in the **second line**. In the **third line**, print that **longest substring**.

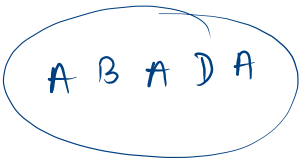
If no such substring exists, print **-1**.

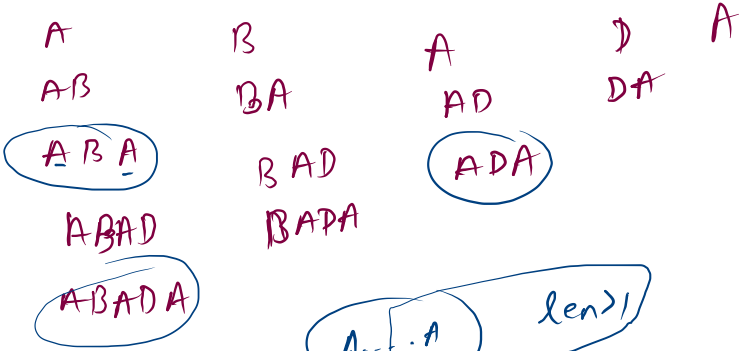
Sample Input 0

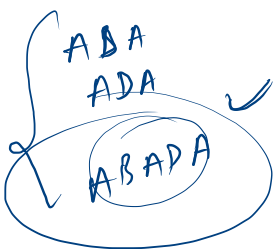
```
ABADA
```

Sample Output 0

```
3
5
ABADA
```







```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 count = 0;
        String maxSs = "";      //ABDBA

        for(int start = 0; start < s.length(); start++){
            for(int end = start; end < s.length(); end++){
                String ss = s.substring(start, end + 1);

                if(ss.length () > 1 && ss.charAt(0) == 'A' && ss.charAt(ss.length()-1) == 'A'){
                    //A....A
                    count++;
                    if(ss.length() > maxSs.length()){
                        maxSs = ss;
                    }
                }
            }
        }
        if(count == 0){
            System.out.println("-1");
        }
        else{
            System.out.println(count);
            System.out.println(maxSs.length());
            System.out.println(maxSs);
        }
    }
}
```

Power of a String

Problem | Submissions | Leaderboard | Discussions

Take a **String str** as input and calculate the **Power** of the string.

Power of a string is defined as the **maximum length** of **substring** that contains only one **unique** character.

A **substring** is a continuous sequence of characters within a string.

Note: All characters in the string are in **lowercase**.

Ans = 3 → ddd

str → abbddd

a
ab
abb
abbd
abbdd
abbddd

b
bb
bbd
bbdd
bbddd

b
bd
bdd
bddd

d
dd
ddd

d
dd

d

eg.

a	a	b	b	d	d	d	e
0	1	2	3	4	5	6	7

ans = ~~0~~ ~~2~~ 3

count = ~~1~~ ~~2~~ ~~3~~ 1

max (ans, count)

s[i] == s[i-1]

count ++

s[i] ≠ s[i-1]

ans

count = 1

bb a a ddd

b b a a d d d

↑

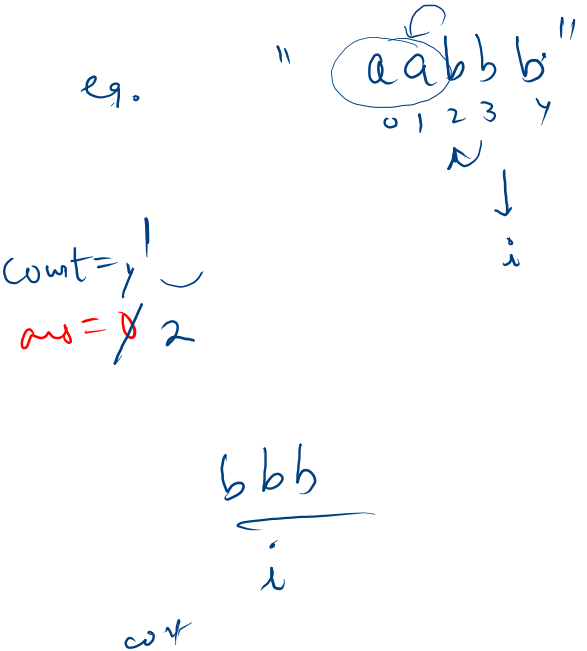
count = 1 ≠ 3

ans = 2

ans
reset

after loop :→ check.

```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 ans = 0;
        int count = 1;

        for(int i = 1; i < s.length(); i++){
            if(s.charAt(i) == s.charAt(i-1)){
                count++;
            }
            else{
                //ans -> reset
                ans = Math.max(ans, count);
                count = 1;
            }
        }
        ans = Math.max(ans, count);
        System.out.println(ans);

    }
}
```



Count Substring of 0 and 1

Problem | Submissions | Leaderboard | Discussions

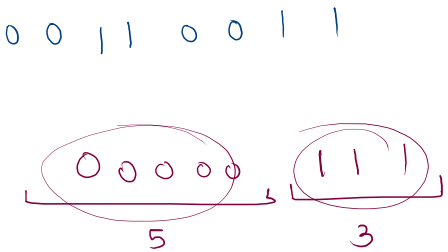
Given a binary string **s**, return the number of **non-empty** substrings that have the same number of **0's** and **1's**, and all the **0's** and all the **1's** in these substrings are grouped consecutively. Substrings that occur multiple times are counted the number of times they occur.

Sample Input 0

```
00110011
```

Sample Output 0

```
6
```

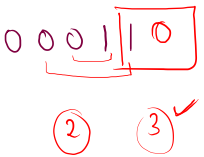


0 0 0 1 1 ⇝ 3

1 1 1 0 0 0 ⇝ 3

0 0 1 1 ⇝ 2

min(5,3)



prev = ~~0~~ 4 2

curr = ~~2~~ 3

ans = 0 + 2 + 2 = 4

ans = 0

p = 0

count = 1 ≠ 3

p, c

p = count 3

Count = 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 prev = 0;
        int curr = 1;
        int ans = 0;
        for(int i = 1; i < s.length(); i++){
            if(s.charAt(i) == s.charAt(i-1)){
                curr++;
            }
            else{
                ans += Math.min(curr, prev);
                prev = curr;
                curr = 1;
            }
        }
        ans += Math.min(curr, prev);
        System.out.println(ans);
    }
}
```