Find Distance B/W Two Characters

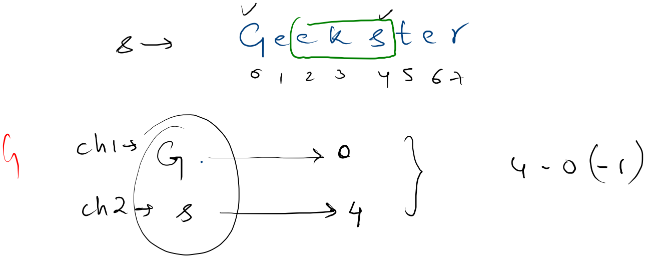
Given a string and two characters. Print the minimum distance between two given characters a and b in a string.

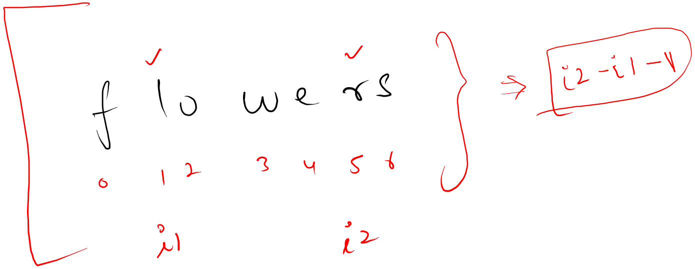
Sample Input 0

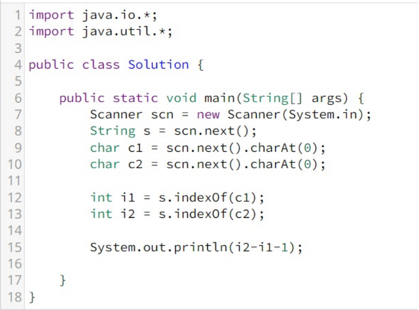
```
Geeks
G
s
```

Sample Output 0

```
3
```







```
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();
        char c1 = scn.next().charAt(0);
        char c2 = scn.next().charAt(0);

        int i1 = s.indexOf(c1);
        int i2 = s.indexOf(c2);

        System.out.println(i2-i1-1);

    }
}
```

m	o	h	i	t	k	u	m	a	r
0	1	2	3	4	5	6	7	8	9
		a					b		

Subarray

10 20 30 40

10

10 20

10 20 30

10 20 30 40

20

20 30

20 30 40

30

30 40

40

Substring

a b c d .

a

ab

abc

abcd

b

bc

bcd

c

cd

d

s → "a b c d"
0 1 2 3

0 a 0
0 ab 1
0 abc 2
0 abcd 3

1 b 1
1 bc 2
1 bcd 3

2 c 2
2 cd 3

3 d 3

0	0 1 2 3
1	1 2 3
2	2 3
3	3

s.substring(2) → single parameter

↳ start with idx 2 till end

→ cd

s → g e e k s t e r
0 1 2 3 4 5 6 7

s.substring(2) → ekster

s.substring(5) → ter

s → g e e k s t e r

0 1 2 3 4 5 6 7

s. substring (2, 8)

2 parameters → s.substring (1, 5)

↳ means start from 1 & end before 5

[1, 5) ⇒ [1, 4]

ans → eeks.

int

s → "1234" ⟶ Integer.parseInt(s)

↓

int → 1234

s ← "1234"

"1"

"12"

"123"

"1234"

"2"

"23"

234

3

34

4

s → "1 2 3"
0 1 2

i

j

ans = ~~0~~ ~~1~~ ~~13~~ = 1~~3~~6 ~~138~~ ~~161~~ 164

123

ss (0,2) → "12"

0,3 "123"

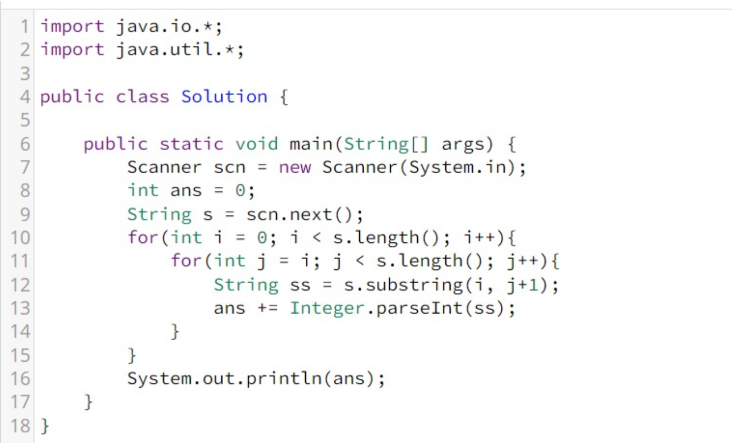
i=0

j= ~~0~~ ~~1~~ 2

$\boxed{ss(0,1)}$

ss → 1

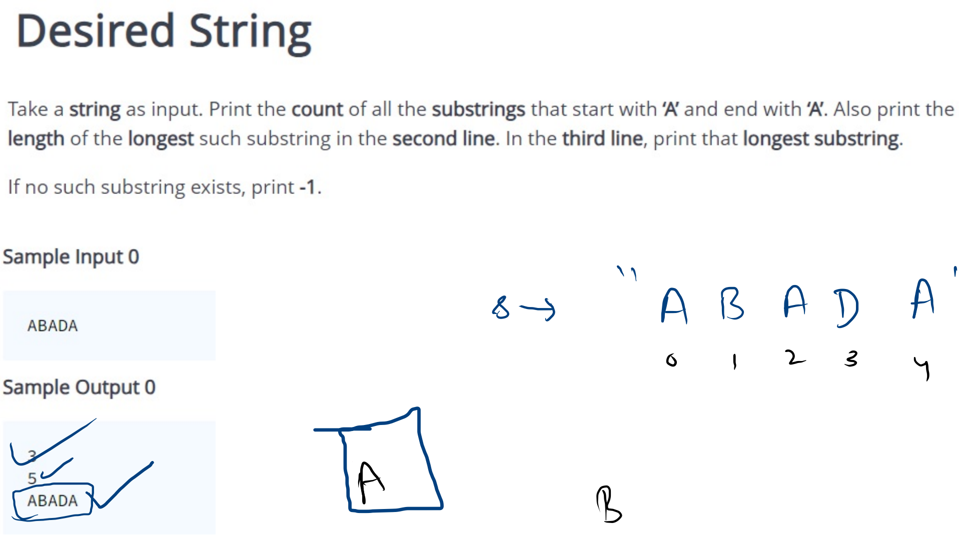
ans += 1



```
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) {
        Scanner scn = new Scanner(System.in);
        int ans = 0;
        String s = scn.next();
        for(int i = 0; i < s.length(); i++){
            for(int j = i; j < s.length(); j++){
                String ss = s.substring(i, j+1);
                ans += Integer.parseInt(ss);
            }
        }
        System.out.println(ans);
    }
}
```



Desired String

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
```

A . . . . . . A

8 → " A B A D A "

0 1 2 3 4

A

AB

ABA

ABAD

ABADA

B

BA

BAD

BADA

A

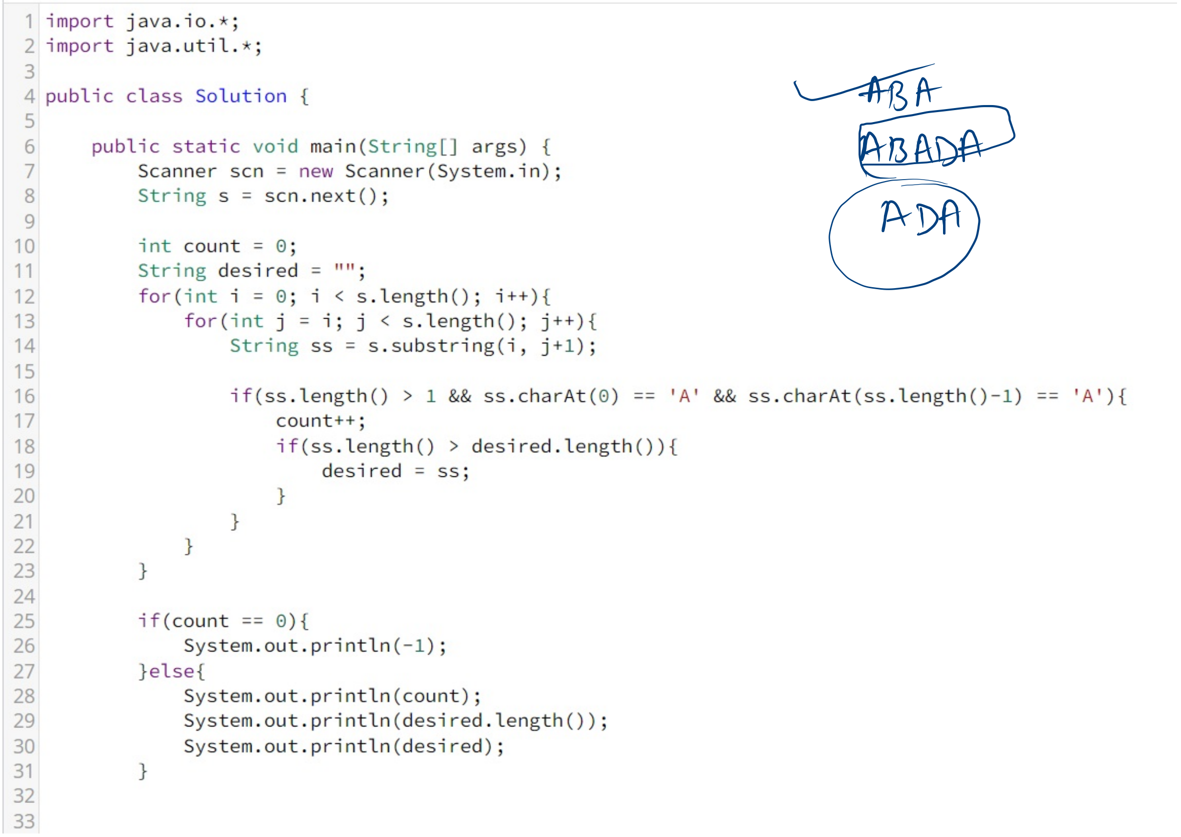
AD

ADA

D

DA

A



```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 desired = "";
        for(int i = 0; i < s.length(); i++){
            for(int j = i; j < s.length(); j++){
                String ss = s.substring(i, j+1);

                if(ss.length() > 1 && ss.charAt(0) == 'A' && ss.charAt(ss.length()-1) == 'A'){
                    count++;
                    if(ss.length() > desired.length()){
                        desired = ss;
                    }
                }
            }
        }

        if(count == 0){
            System.out.println(-1);
        }else{
            System.out.println(count);
            System.out.println(desired.length());
            System.out.println(desired);
        }

```

~~ABA~~

ABADA

ADA

"A B ADA"

3

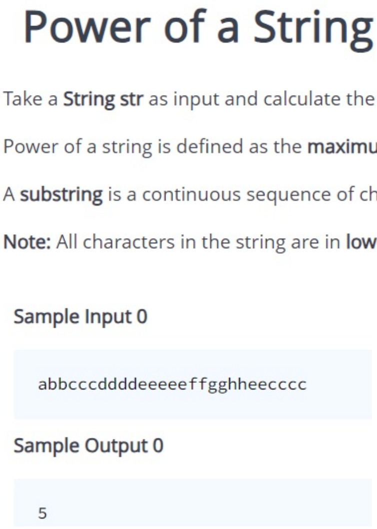
5

ABADA

ADA

3 > 0

5 > 3



Power of a String

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**.

Sample Input 0

```
abbcccddddeeeeeffgghheecccc
```

Sample Output 0

```
5
```

curr = ~~1~~ 2

max = ~~0~~ ~~2~~ 3

b b b a c c

2 3 4 5 6 7

i

if → s[i] == s[i-1]
    ↳ curr++

else → max (curr, max)
    curr = 1

aa