Beautiful Binary String

by Shafaet

Alice has a binary string, B, of length n. She thinks a binary string is beautiful if and only if it doesn't contain the substring **"010"**.

In one step, Alice can change a 0 to a 1 (or vice-versa). Count and print the minimum number of steps needed to make Alice see the string as beautiful.

Input Format

The first line contains an integer, n (the length of binary string B).
The second line contains a single binary string, B, of length n.

Constraints

- $1 \leq n \leq 100$
- Each character in $B \in \{0, 1\}$.

Output Format

Print the minimum number of steps needed to make the string beautiful.

Sample Input 0

```
7
0101010
```

Sample Output 0

```
2
```

Sample Input 1

```
5
01100
```

Sample Output 1

```
0
```

Sample Input 2

```
10
0100101010
```

Sample Output 2

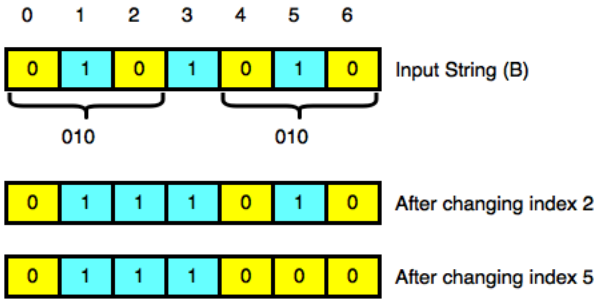
```
3
```

Explanation

Sample Case 0:

In this sample, $B = \text{"0101010"}$

The figure below shows a way to get rid of each instance of "010":



Because we were able to make the string beautiful by changing 2 characters (B_2 and B_5), we print 2.

Sample Case 1:

In this sample $B = \text{"01100"}$

The substring "010" does not occur in B, so the string is already beautiful and we print 0.

Submissions:25959

Max Score:20

Difficulty: Easy

Rate This Challenge:

More

Current Buffer (saved locally, editable) Java 7

```
import java.io.*;
import java.util.*;
import java.text.*;
import java.math.*;
import java.util.regex.*;

public class Solution {

    static int minSteps(int n, String B){
        // Complete this function
    }

    public static void main(String[] args) {
        Scanner in = new Scanner(System.in);
        int n = in.nextInt();
        String B = in.next();
        int result = minSteps(n, B);
        System.out.println(result);
    }
}

```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code