

Program 2:-

Gary is an avid hiker. He tracks his hikes meticulously, paying close attention to small details like topography. During his last hike, he took exactly n steps. For every step he took, he noted if it was an uphill or a downhill step. Gary's hikes start and end at sea level. We define the following terms:

A mountain is a non-empty sequence of consecutive steps above sea level, starting with a step up from sea level and ending with a step down to sea level. A valley is a non-empty sequence of consecutive steps below sea level, starting with a step down from sea level and ending with a step up to sea level.

Given Gary's sequence of up and down steps during his last hike, find and print the number of valleys he walked through.

Input Format

The first line contains an integer, n , denoting the number of steps in Gary's hike.

The second line contains a single string of characters. Each character belongs to $\{U, D\}$ (where U indicates a step up and D indicates a step down), and the i (th) character in the string describes Gary's i (th) step during the hike.

Constraints

$2 \leq N \leq 10^6$

Output Format

Print a single integer denoting the number of valleys Gary walked through during his hike.

Sample Input

8
UDDDUDUU

Implementation_:-

```
import java.util.Scanner;

class AvidHiker {
    int countingValleys(int n, String str) {
        int i = 0;
        int countValley = 0;

        int countMount = 0;
        while (i < n) {
            if (str.charAt(i) == 'U') {
                countValley--;
                countMount++;
            } else {
                countValley++;
                countMount--;
            }
            i++;
        }
        if (countValley < 0) {
```

```

        return 0;
    }

    return countValley;
}

}

class Hiker {
    public static void main(String[] args) {
        AvidHiker gray = new AvidHiker();
        Scanner scan = new Scanner(System.in);
        int n = scan.nextInt();

        String str = scan.next();
        if(str.contains("U") || str.contains("D")){

            System.out.println(gray.countingValleys(n, str.toUpperCase()));
        }
        else{
            System.out.println("Unacceptable character");
        }

    }

}

```

Testing:-

The screenshot displays the Visual Studio Code interface with the 'Hiker.java' file open. The left sidebar shows the 'CPH JUDGE: RESULTS' panel, which lists two test cases, both of which passed. The first test case, 'Testcase 1', passed in 191ms with an input of '8' and 'UDDUDDUU', an expected output of '0', and a received output of '0'. The second test case, 'Testcase 2', passed in 157ms with an input of '7' and 'DDDDDDDD', an expected output of '7', and a received output of '7'. The main editor area shows the code for the 'Hiker' class, which includes a 'main' method and a 'countingValleys' method. The code is written in Java and uses the 'Scanner' class for input and 'System.out.println' for output.

Local: Hiker

CPH JUDGE: RESULTS

Testcase 1 Passed 191ms

Input: 8
UDDUDDUU
Expected Output: 0
Received Output: 0

Testcase 2 Passed 157ms

Input: 7
DDDDDDDD
Expected Output: 7
Received Output: 7

+ New Testcase

Set ONLINE_JUDGE

Run All + New Stop Help

Run | Debug

Class Hiker {

public static void

AvidHiker gray

Scanner scan =

int n = scan.n

String str = s

if(str.contain

System.out

else{

System.out

}

}

Ln 32, Col 43 Spaces: 4 UTF-8 LF () Java Go Live