An avid hiker keeps meticulous records of their hikes. During the last hike that took exactly $steps$ steps, for every step it was noted if it was an uphill, U, or a downhill, D step. Hikes always start and end at sea level, and each step up or down represents a 1 unit change in altitude. We define the following terms:

- A mountain is a 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 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 the sequence of up and down steps during a hike, find and print the number of valleys walked through.

Example

$steps = 8$ $path = [DDUUUUDD]$

The hiker first enters a valley 2 units deep. Then they climb out and up onto a mountain 2 units high. Finally, the hiker returns to sea level and ends the hike.

Function Description

Complete the countingValleys function in the editor below.

countingValleys has the following parameter(s):

- int steps: the number of steps on the hike
- string path: a string describing the path

Returns

- int: the number of valleys traversed

Input Format

The first line contains an integer $steps$, the number of steps in the hike.
The second line contains a single string $path$, of $steps$ characters that describe the path.

Constraints

- $2 \leq steps \leq 10^6$
- $path[i] \in \{UD\}$

Sample Input

```
8
UDDDUDUU
```

Sample Output

```
1
```

Explanation

If we represent _ as sea level, a step up as /, and a step down as \, the hike can be drawn as:

```
_/\    _
   \  /
    \/\/
```

The hiker enters and leaves one valley.

```python
#
# The function is expected to return an INTEGER.
# The function accepts following parameters:
#  1. INTEGER steps
#  2. STRING path
#

def countingValleys(steps, path):
    # Write your code here
    valleys = 0
    tracker = 1 if path[0] == 'U' else -1
    for p in path[1:]:
        down = False

        if tracker < 0: down = True

        if p == 'D':
            tracker -= 1
        else:
            tracker += 1

        if tracker == 0 and down is True:
            valleys += 1

    return valleys

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    steps = int(input().strip())

    path = input()

    result = countingValleys(steps, path)

    fptr.write(str(result) + '\n')

    fptr.close()
```

Line: 27 Col: 9

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

Congratulations

You solved this challenge. Would you like to challenge your friends?

Next Challenge

- Test case 0
- Test case 1
- Test case 2
- Test case 3
- Test case 4
- Test case 5
- Test case 6

Compiler Message

```
Success
```

Input (stdin) Download

```
8
UDDDUDUU
```

Expected Output Download

```
1
```