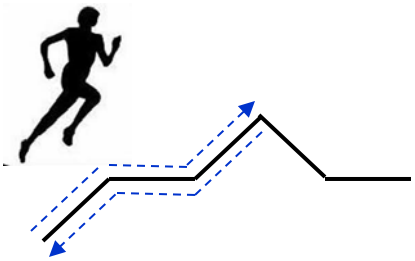
Practice Exercise #36: Jogging in NUS

http://www.comp.nus.edu.sg/~cs1020/4_misc/practice.html



Objective:

- Using recursion

Task statement:

John likes jogging inside the NUS campus. Every time John starts jogging from PGP and he must be back to PGP within **M** seconds ($1 \le \mathbf{M} \le 1{,}000{,}000$). However, the road in NUS is not always flat, sometimes uphill or downhill. The road can be divided into **T** units ($1 \le \mathbf{T} \le 10{,}000$) in length and consists of equal-length portions that are uphill, flat, or downhill.

John takes **U** seconds ($1 \le \mathbf{U} \le 1000$) to run one unit of uphill road, **F** seconds ($1 \le \mathbf{F} \le 1000$) for a unit of flat road, and **D** seconds ($1 \le \mathbf{D} \le 1000$) for a unit of downhill road. Note that when returning to PGP, uphill units become downhill units and downhill units become uphill units.

Given the road description and time limit (**M** seconds), help John to figure out the farthest distance (number of units) he can run from PGP and still can make it back to PGP within **M** seconds.

(In your program, you should use more descriptive variable names instead of **M**, **T**, **U**, **F** and **D** and follow Java naming convention.)

Input

Line 1: **M**, **T**, **U**, **F**, and **D** separated by space.

Line 2: A **T**-character string describing the road. Each character is '**u**', '**f**', or '**d**' indicating uphill, flat, or downhill respectively.

Output

A single integer that is the farthest distance (number of units) that John can run from PGP and make it back in time.

Sample Input

```
13 5 3 2 1
ufudf
```

Sample Output

```
3
```