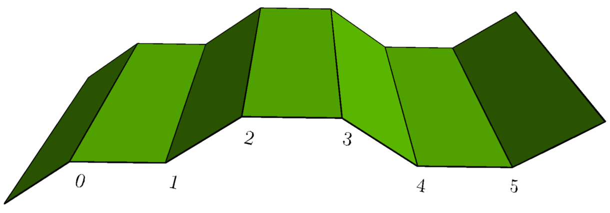
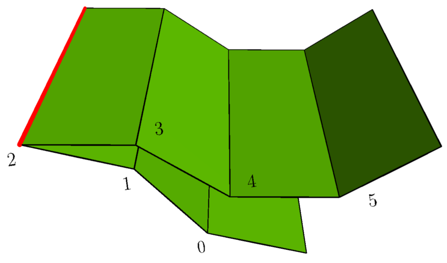
Foglietto illustrativo (`foglietto`)

While cleaning his house, Massimo found some expired drugs. Not knowing how to dispose of them, he responsibly read the indications on the leaflet. Now he has an even bigger challenge: folding back the leaflet into the package!

The leaflet is exactly $N + 1$ cm long and is composed of N horizontal folds spaced 1 cm apart. Each of them is represented by a 0 if downward, by a 1 if upward. Massimo can fold the leaflet at any point, provided that the folds on one half match the ones of the other: that is, if whenever one half there is a 0, on the other half there must be a 1, and vice versa. The two halves don't necessarily have to be the same length.



Before the fold: `010011`



After the fold: `011`

After this operation, the folds on the leaflet can be represented by the string obtained from the previous one by removing the prefix or suffix corresponding to the shortest side (including the fold itself).

The box is very small, and to put the leaflet back in, it must be folded back to a length of 1 cm. Help Massimo calculate the **minimum number of folds** that are required to put back the leaflet into the package!

Implementation

You need to submit a single file, with the extension `.cpp`.

> ☞ Among the attachments in this task you will find a template `foglietto.cpp` with a sample implementation.

You will have to implement the following function:

```
C++    | int piega(int N, string S);
```

- The integer N represents the number of folds in the leaflet.
- The string S, indexed from 0 to $N-1$, contains the folds of the leaflet, where the i-th character is 0 if the fold is downward, 1 if upward.
- The function must return the minimum number of folds to put back the leaflet in its box.

The grader will call function `piega` and print the return value on the output file.

Sample grader

Among this task's attachments you will find a simplified version of the grader used during evaluation, which you can use to test your solutions locally. The sample grader reads data from `stdin`, calls the function that you should implement and writes back on `stdout` using the following format.

The input file is made up of two lines, containing:

- Line 1: the integer N.
- Line 2: the string S.

The output file is composed of a single line, containing the return value of function `piega`.

Constraints

- $1 \leq N \leq 70\,000$.
- The string S is exactly N characters long and it's composed only by the characters '`0`' and '`1`'.

Scoring

Your program will be tested on a number of testcases grouped in subtasks. In order to obtain the score associated to a subtask, you need to correctly solve all the testcases it contains.

- **Subtask 1** [**0 points**]: Sample cases.
- **Subtask 2** [**4 points**]: The string S is a series of 0's followed by a series of 1's.
- **Subtask 3** [**7 points**]: $N \leq 10$.
- **Subtask 4** [**14 points**]: $N \leq 20$.
- **Subtask 5** [**17 points**]: $N \leq 600$
- **Subtask 6** [**27 points**]: $N \leq 3000$
- **Subtask 7** [**23 points**]: The string S is generated randomly[1].
- **Subtask 8** [**8 points**]: No additional constraints.

[1]The string S is generated in the following way: initially S it empty; then a fair coin is flipped N times, each time if it results in head 0 is added at the end of the string S, otherwise 1 is added at the end of the string.

Examples

stdin	stdout
6 010011	3
11 01100100111	4
10 0111010011	5
10 0101010101	10

Explanation

In the **first sample case** it's possible to fold back the leaflet with 3 moves:

- initially the leaflet is 7 cm long;
- fold on the third crease, reducing its length to 4 cm;
- fold the leaflet in half reducing its length to 2 cm;
- fold once again the leaflet in half: it's now completely folded and thus can be put back into the packaging.

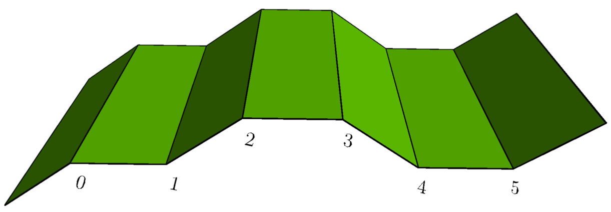


Figure 1: Initial leaflet

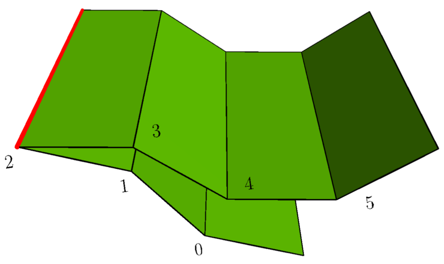


Figure 2: Step 1

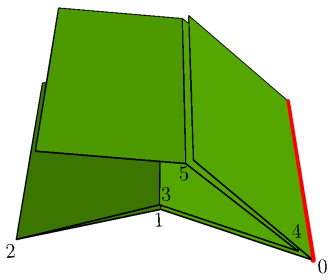


Figure 3: Step 2

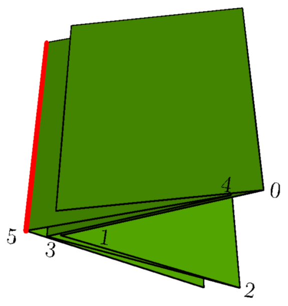


Figure 4: Step 3

In the **second sample case** it's possible to fold back the leaflet with 4 moves:

- initially the leaflet is 12 cm long;
- fold on the fifth crease reducing its length to 7 cm;

- fold in the point where the first and ninth crease match, reducing the leaflet length to 4 cm;
- fold the leaflet in half reducing its length to 2 cm;
- fold another time the leaflet in half: it's now completely folded and thus can be put back into the packaging.

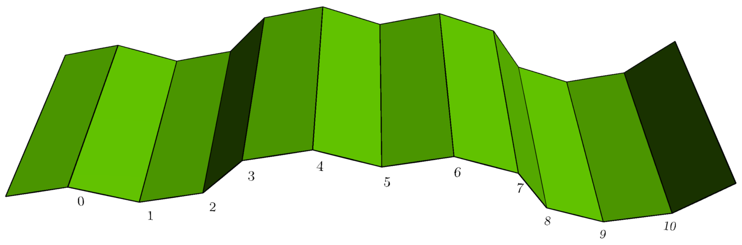


Figure 5: Initial leaflet

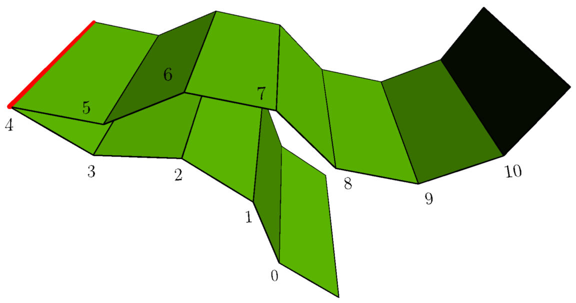


Figure 6: Step 1

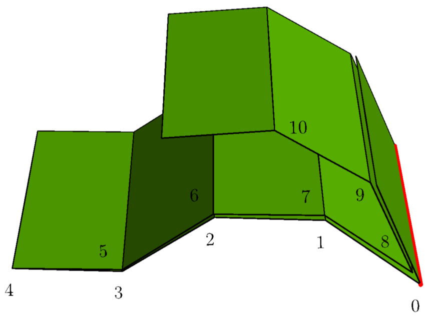


Figure 7: Step 2

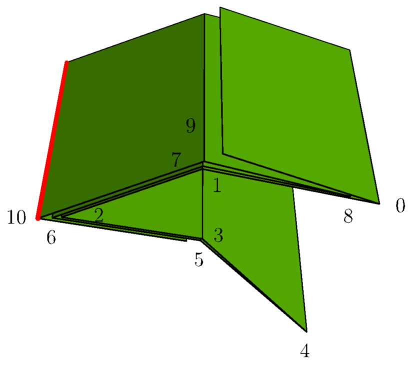


Figure 8: Step 3

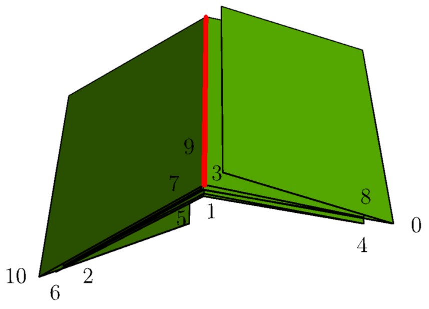


Figure 9: Step 4

In the **third sample case**, it's possible to fold back the leaflet in 5 moves:

- initially the leaflet is 11 cm long;
- fold on the second crease, reducing its length to 9 cm;
- fold on the forth crease, reducing its length to 7 cm;
- fold on the seventh crease, reducing its length to 4 cm;
- fold the leaflet in half reducing its length to 2 cm;
- fold once again the leaflet in half: it's now completely folded and thus can be put back into the packaging.

In the **fourth sample case**, the only way to fold back completely the leaflet is starting from an end and folding back and forth on each point individually. The minimum number of folds is thus 10.