A - Fewest Flops

Source file name: `flops.py`
Time limit: 1 second

A common way to uniquely encode a string is by replacing its consecutive repeating characters (or "chunks") by the number of times the character occurs followed by the character itself. For example, the string `aabbbaabaaaa` may be encoded as `2a3b2a1b4a`. (Note for this problem even a single character `b` is replaced by `1b`).

Suppose we have a string S and a number k such that k divides the length of S. Let S_1 be the substring of S from 1 to k, S_2 be the substring of S from $k+1$ to $2k$, and so on. We wish to rearrange the characters of each block S_i independently so that the concatenation of those permutations S' has as few chunks of the same character as possible. Output the fewest number of chunks.

For example, let S be `uuvuwwuv` and k be 4. Then S_1 is `uuvu` and has three chunks, but may be rearranged to `uuuv` which has two chunks. Similarly, S_2 may be rearranged to `vuww`. Then S', or $S_1 S_2$, is `uuuvvuww` which is 4 chunks, indeed the minimum number of chunks.

Input

The input begins with a line containing t ($t \geq 0$), the number of test cases. The following t lines contain an integer k and a string S made of no more than 1000 lowercase English alphabet letters. It is guaranteed that k will divide the length of S.

The input must be read from standard input.

Output

For each test case, output a single line containing the minimum number of chunks after we rearrange S as described above.

The output must be written to standard output.

Sample Input	**Sample Output**

`2`
`5 helloworld`
`7 thefewestflops`
	`8`
	`10`