Steve loves playing with palindromes. He has a string, s, consisting of n lowercase English alphabetic characters (i.e., a through z). He wants to calculate the number of ways to insert exactly 1 lowercase character into string s such that the length of the longest palindromic subsequence of s increases by *at least* k. Two ways are considered to be *different* if either of the following conditions are satisfied:

- The positions of insertion are different.
- The inserted characters are different.

This means there are *at most* $26 \times (n+1)$ different ways to insert exactly 1 character into a string of length n.

Given q queries consisting of n, k, and s, print the number of different ways of inserting exactly 1 new lowercase letter into string s such that the length of the longest palindromic subsequence of s increases by *at least* k.

Input Format

The first line contains a single integer, q, denoting the number of queries. The $2q$ subsequent lines describe each query over two lines:

1. The first line of a query contains two space-separated integers denoting the respective values of n and k.
2. The second line contains a single string denoting s.

Constraints

- $1 \leq q \leq 10$
- $1 \leq n \leq 3000$
- $0 \leq k \leq 50$
- It is guaranteed that s consists of lowercase English alphabetic letters (i.e., a to z) only.

Subtasks

- $1 \leq n \leq 100$ for 25% of the maximum score.
- $1 \leq n \leq 1000$ for 70% of the maximum score.

Output Format

On a new line for each query, print the number of ways to insert exactly 1 new lowercase letter into string s such that the length of the longest palindromic subsequence of s increases by *at least* k.

Sample Input

```
3
1 1
a
3 2
aab
3 0
aba
```

Sample Output

```
2
1
104
```

Explanation

We perform the following $q = 2$ queries:

1. The length of the longest palindromic subsequence of $s =$ a is 1. There are two ways to increase this string's length by *at least* $k = 1$:
    1. Insert an a at the start of string s, making it aa.
    2. Insert an a at the end of string s, making it aa.

Both methods result in `aa`, which has a longest palindromic subsequence of length 2 (which is longer than the original longest palindromic subsequence's length by $k = 1$). Because there are two such ways, we print 2 on a new line.

2. The length of the longest palindromic subsequence of $s =$ `aab` is 2. There is one way to increase the length by *at least* $k = 2$:

    1. Insert a `b` at the start of string s, making it `baab`.

    We only have one possible string, `baab`, and the length of its longest palindromic subsequence is 4 (which is longer than the original longest palindromic subsequence's length by $k = 2$). Because there is one such way, we print 1 on a new line.