Programming Problem Set

1. Count Palindromes

You are given the following:

- String s of length n consisting of lowercase alphabets

Task

Determine the number of subsequences of string s such that the subsequence becomes palindromic after rearrangement of the subsequence (if needed).

> **Notes:**
>
> - A string a is said to be a subsequence of string b if a can be obtained from b by deleting some characters without changing the ordering of the remaining characters.
> - An empty string is palindromic.

Example

Assumptions:

- $n = 3$
- $s =$ `"abb"`

Approach:

The subsequences {"", "a", "b", "b", "bb"} of string s are palindromic with no rearrangement.

The subsequence {"abb"} of string s after rearrangement gives {"bab"} which is palindromic.

Therefore, the number of subsequences of string s which becomes palindromic after rearrangement (if needed) is 6. Hence, the answer is 6.

Function description

Complete the `CountPalindromes` function provided in the editor. This function takes the following 2 parameters and returns the number of subsequences of string s such that subsequence becomes palindromic after rearrangement:

- n: Represents the length of string s
- s: Represents the string s of length n

Input format

Note: This is the input format that you must use to provide custom input (available above the Compile and Test button).

- The first line contains an integer T denoting the number of test cases. T also denotes the number of times you have to run the `CountPalindromes` function on a different set of inputs.
- For each test case:
  - The first line contains a single integer n.
  - The second line contains a string s of length n consisting of lowercase English alphabets.

Output format

For each test case, print an integer in a new line representing the number of subsequences of string s such that subsequence becomes palindromic after the rearrangement of the subsequence (if needed).

Constraints

- $1 \leq T \leq 10$
- $1 \leq n \leq 120$
- s consists of lowercase alphabets

2. Ways

You are given an array A of distinct positive integers of length n.

Task

Determine the number of ways to arrange the array such that the maximum length of subarray containing only even elements is no more than two. Since the answer can be very large, print it modulo ($10^9 + 7$).

> **Notes:**
>
> - 0 based indexing is followed.
> - **Subarray:** A subarray is the sequence of consecutive elements of the array. For example, in array `[1, 2, 3]` the subarrays are as follows:
>   - [] - empty
>   - [1]
>   - [2]
>   - [3]
>   - [1, 2]
>   - [2, 3]
>   - [1, 2, 3]

Function description

Complete the `solve` function provided in the editor. This function takes the following 2 parameters and returns an integer:

- n: Represents the length of the array.
- A: Represents an array.

Input format

Note: This is the input format that you must use to provide custom input (available above the Compile and Test button).

- The first line contains an integer T denoting the number of test cases. T also denotes the number of times you have to run the `solve` function on a different set of inputs.
- For each test case:
  - The first line contains an integer denoting n.
  - The second line contains n space-separated integers.

Output format

For each test case, print the answer representing the number of ways to arrange the array modulo ($10^9 + 7$) in a new line.