**Number of distinct subsequences: -**

**Hard** Accuracy: **30.43%** Submissions: **30K+** Points: **8**

Given a string consisting of lower case English alphabets, the task is to find the number of distinct subsequences of the string  
**Note:** Answer can be very large, so, ouput will be answer modulo **109+7**.

**Example 1:**

**Input:**   
s = "gfg"

**Output:**   
7

**Explanation:**   
The seven distinct subsequences are "", "g", "f", "gf", "fg", "gg" and "gfg" .

**Example 2:**

**Input:**   
s = "ggg"

**Output:**   
4

**Explanation:**   
The four distinct subsequences are "", "g", "gg", "ggg".

**Your task:**  
You do not need to read any input or print anything. The task is to complete the function **distinctSubsequences()**, which takes a string as input and returns an integer.

**Expected Time Complexity:** O(|str|)  
**Expected Auxiliary Space:** O(|str|)

**Constraints:**  
1 ≤ |s| ≤ 105  
s contains lower case English alphabets

**Code: -**

//{ Driver Code Starts

#include <bits/stdc++.h>

using namespace std;

// } Driver Code Ends

//User function template for C++

class Solution{

public:

int mod = 1000000007;

int distinctSubsequences(string s){

int n = s.size();

vector<int> dp(n+1);

dp[0] = 1;

unordered\_map<char,int> mp;

for(int i=1; i<=n; ++i){

char ch = s[i-1];

dp[i] = (2 \* dp[i-1]) % mod;

if(mp.find(ch) != mp.end()){

int foundatindex = mp[ch];

dp[i] = (dp[i] - dp[foundatindex - 1] + mod) % mod;

}

mp[ch] = i;

}

return dp[n];

}

};

//{ Driver Code Starts.

int main()

{

ios\_base::sync\_with\_stdio(0);

cin.tie(NULL);

cout.tie(NULL);

int t;

cin >> t;

while(t--)

{

string s;

cin >> s;

Solution ob;

cout << ob.distinctSubsequences(s) << "\n";

}

return 0;

}

// } Driver Code Ends

**T.C: - O(|string|)**

**S.C: - O(|string|)**