Count off 3

Input file:	`standard input`
Output file:	`standard output`
Time limit:	2 seconds
Memory limit:	512 megabytes

Xiao Z and Xiao R have started playing the counting game again. This time, facing the predicament of running out of new twists for their game and Xiao R going abroad for an exchange program, they find themselves unable to play together in person and can only communicate online.

However, the idea of playing online sparked a new inspiration. As is well-known, data is represented in binary form. Previously, they played the game in decimal, but this time they proposed playing in binary instead.

In the traditional counting game, players take turns counting numbers, and any number that is a multiple of 7 or contains the digit 7 must be skipped. Unfortunately, they quickly realized that this rule isn't very interesting in binary since the digit 7 doesn't exist in binary, and whether a number is a multiple of 7 is independent of its representation in any base.

But what if they looked at it from another angle? The same string of digits can represent vastly different numbers in different bases. For example, the string "1010" represents 10 in binary ($1010_2 = 0 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 1 \cdot 2^3 = 10$), but represents 30 in ternary ($1010_3 = 0 \cdot 3^0 + 1 \cdot 3^1 + 0 \cdot 3^2 + 1 \cdot 3^3 = 30$). Following this logic, how many strings of digits are there that, in any base, are not multiples of 7? They decided to make a rule that only such numbers could be counted.

Therefore, they defined the following formal rules:

A string s is defined as a *valid 01 string* if and only if s is non-empty, consists only of the characters '0' and '1', and does not start with '0'.

They also defined a *base function* $f(s, k)$, where s is a valid 01 string and k is an integer not less than 2. If s has a length of n, indexed from 0, then $f(s, k) = \sum_{i=0}^{n-1} s_{n-i-1} \cdot k^i$, noting that s_0 is the high-order bit and s_{n-1} is the low-order bit.

Given a valid 01 string s as the *upper limit* for counting, they wanted to know how many strings (modulo $10^9 + 7$), not exceeding s, are there that are not multiples of 7 in any base? In other words, how many valid 01 strings s' are there, satisfying that for any integer $k \geq 2$, 7 does not divide $f(s', k)$, where "not exceeding s" means in the binary sense, i.e., $f(s', 2) \leq f(s, 2)$?

Input

There are multiple test cases. The first line of the input contains a single integer T ($1 \leq T \leq 10$), indicating the number of the test cases. For each of the test case:

The first line of the input contains a single valid 01 string s ($|s| \leq 10^4$), indicating the *upper limit* for counting.

Output

For each test case, output a single line contains a single integer, indicating the answer. Since the answer can be huge, you need to output it modulo $10^9 + 7$.

Example

standard input	standard output
5 1 1010 110101 1000111000 101101001000	1 2 15 114 514