HKUST Local Contest Fall 2010
October 17, 2010

Contest Time: 1:00pm - 5:30pm

Letter	Page	Time limit	Memory limit	Name
A	2	2 sec	64MB	Concatenate Numbers
B	3	10 sec	64MB	Let's Avoid Fours
C	4	5 sec	64MB	Nice or Ugly
D	5	2 sec	64MB	Coin Problem Revisited
E	6	60 sec	64MB	Reverse Difference
F	7	10 sec	64MB	Odd-Digitable Numbers

Contest Organizer:
Prof. Ke Yi
Mr. Derek Hao Hu

Problemsetter:
Mr. Derek Hao Hu

Judges:
Mr. Derek Hao Hu
Mr. Tong Zhu

Contest Rules and Regulations:

1. **This contest is an individual contest. Discussions between contestants are strictly prohibited.** Sanctions will be imposed on contestants if they are found to have violated the regulations governing integrity and honesty.
2. In this contest, **the contestants are given six programming problems**. The goal is to solve as many problems as possible. For those who solve the same number of problems, the one with lower score wins. (The scoring system will be explained below.)
3. **The programming languages to be used in this contest are C/C++ and JAVA.** The contestants use PC^2 to submit their source codes to the judge and the source codes are compiled by Visual C++ or JAVA.
4. **The contestant should read the input and write the output via standard I/O.** The contestants can assume that all test cases are of the format as stated in the problem statements. i.e. No exception handling is needed.
5. The correctness of each submission is judged by inputting test cases into the submitted program. The submission is regarded as correct if its outputs match completely with the model outputs. The submission is judged as correct or wrong. **No partial credit is given.**
6. The contestants can re-submit another source code after previous wrong submissions.
7. **All programs should not run for more than the time limit specified in the problem** (in most cases a "correct" implementation will run far less than the time limit we provide).
8. **The contestants are ranked firstly by the number of problems solved, and secondly the total time spent on solving the problems.** Time spent on solving one problem is the time between the start of contest and the submission of the correct implementation of that problem. For each problem you solved, a penalty of 20 minutes will be added to your score for each wrong submission of that problem.
9. **The contestants are allowed to bring any hard copies of books, notes, references, dictionaries and sketch papers to the contest site.** Electronic devices are forbidden.

Problem A. Concatenate Numbers

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	2 seconds
Memory limit:	64 megabytes

The first problem is always the most straightforward problem in its description. But is it the easiest problem?

Given a positive integer N, concatenate one or more copies of number to create an integer that is divisible by k. Do not add any leading zeroes. Output the least number of copies needed, or -1 if it is impossible.

Input

The first line of the input is an integer T, which indicates the number of test cases.

T test cases follow. Each test case contains two numbers N and K, where $1 \leq N \leq 1000000000$ and $1 \leq K \leq 100000$.

Output

You should output T lines exactly. Each line contains the least number of copies needed to create an integer that is divisible by k, or -1 if impossible.

Example

Standard Input	Standard Output

5
2 9
121 11
1 2
35 98765
1000000000 3
	9
	1
	-1
	9876
	3

Problem B. Let’s Avoid Fours

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	10 seconds
Memory limit:	64 megabytes

It is a known fact that of all numbers, 4 is the one that brings the worst luck. It is for this reason that when generating number sequences, we need to avoid patterns related to the number 4 as much as possible.

You are given an integer n. Count the number of positive integers that satisfy all of the following conditions:

- The number contains at most n digits.
- The number does not contain four consecutive ’4’ digits. For example, 43444124 is allowed, but 45444474 is not.
- The number of digits in the number is not a multiple of any of the integers greater than 10 that contain only ’4’ in their decimal representations (44, 444, 4444, 44444, ...).

Output the total count of these numbers modulo 1000000007.

Input

The first line of the input is an integer T, which indicates the number of test cases.

T test cases follow. Each test case contains only one integer N where N will be between 1 and 4^{10}, inclusive. (Reminder, use “long long” to store this number.)

Output

You should output T lines exactly. Each line contains the total count of the numbers modulo 1000000007 that satisfy the abovementioned conditions with at most n digits.

Example

Standard Input	Standard Output
5	9998
4	99980
5	576334228
87	576334228
88	547731225
4128	

Problem C. Nice or Ugly

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	5 seconds
Memory limit:	64 megabytes

A string is called ugly if it has 3 vowels in a row, or 5 consonants in a row, or both. A string is called nice if it is not ugly. You are given a string s, consisting of uppercase letters (’A’-’Z’) and question marks (’?’). Return “UGLY” if the string is definitely ugly (that means you cannot substitute letters for question marks so that the string becomes nice), “NICE” if the string is definitely nice, and “UNKNOWN” if it can be either ugly or nice. (The letters AEIOU are vowels and all others are consonants.)

Input

The first line of the input is an integer T, which indicates the number of test cases.

T lines follow, each line contains a string s, which contains between 1 and 50 characters, inclusive. Each character in s will be either ‘?’ or an uppercase letter.

Output

You should output T lines exactly. Each line contains either “UGLY”, “NICE”, or “UNKNOWN”.

Example

Standard Input	Standard Output
5	NICE
HELLOWORLD	UGLY
ABCDEFGHIJKLMNOPQRSTUVWXYZ	UNKNOWN
HELLOW?RLD	NICE
H??LOWOR??	UGLY
EE?FFFF	

Problem D. Coin Problem Revisited

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	2 seconds
Memory limit:	64 megabytes

We use coins everyday and the problem with coins are almost everywhere. Let's now consider a monetary system in the Kingdom of Imagination. The coins come in different values. The values used are:

1, 10, 25, 100, 1000, 2500, 10000, 100000, 250000, 1000000, . . .

Formally, for each $K \geq 0$ there are coins worth 10^K, and coins worth $25 * 100^K$.

You want to buy a new car with price C. I want to know the smallest number of coins necessary to pay exactly the cost of the car (assuming you have a sufficient supply of coins of each of the types you will need). So can you help me solve this problem?

Input

The first line of the input contains an integer T, indicating the number of test cases. T lines follow, each line contains only one integer $C(1 \leq C \leq 10^{15})$.

Reminder: use "long long" to store this number.

Output

You should output T lines where each line indicates the smallest number of coins necessary to pay the cost of the car.

Example

Standard Input	Standard Output
4 47 9 250111 76540123	5 9 4 16

Problem E. Reverse Difference

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	60 seconds
Memory limit:	64 megabytes

To obtain the reverse of a number, write it backwards from right to left. For example, the reverse of 1234 is 4321, and the reverse of 100 is 1 (leading zeroes are always ignored). Output the smallest non-negative number **x**, such that the difference **x** - reverse(**x**) is equal to the given integer **D**. If no such number exists, output "NONE".

Input

The first line of the input contains an integer T, indicating the number of test cases.

T lines follow. Each line has only one integer D, where $1 \leq D \leq 1000000$.

Output

You should output the smallest non-negative number such that satisfies the problem requirement. If it does not exist, output "NONE".

Example

Standard Input	Standard Output
6 18 15 4275 900 1989 857232 45	20 NONE 5080 101001 100990 860300 50

Problem F. Odd-Digitable Numbers

Input file:	`Standard Input`
Output file:	`Standard Output`
Time limit:	10 seconds
Memory limit:	64 megabytes

An odd-digitable number is a positive integer which consists of only odd digits. For example, 1, 7, 15, 91 and 73353 are odd-digitable numbers, but 2, 70, 94 and 72653 are not odd-digitable.

You will be given integers N and M. Tell me the smallest odd-digitable number that equals M modulo N. If there are no such odd-digitable numbers, output -1 instead.

Reminder: It is possible that this number may exceed the representation power of a signed integer $2^{31} - 1$.

Input

The first line of the input contains an integer T, indicating the number of test cases.

T test cases follow, each test case contain two numbers, N and M. ($2 \leq N \leq 100000, 0 \leq M \leq N - 1$).

Output

For each test case, either output the smallest odd-digitable number that equals M modulo N or output -1 if such number does not exist for the particular pair of M and N.

Example

`Standard Input`	`Standard Output`

`4`
`10 7`
`22 12`
`29 0`
`5934 2735`
	`7`
	`-1`
	`319`
	`791957`