These days, it has become commonplace to make purchases over the internet using a credit card. However, because credit card numbers are relatively long, it is easy to make a mistake while typing them in. In order to quickly identify errors like typos, most e-commerce websites use a checksum algorithm to verify credit card numbers.

One popular checksum algorithm is the Luhn algorithm, which can detect any single-digit error as well as many common multiple-digit errors:

1. Starting with the second-last digit and moving backwards, double every other digit to obtain a list of numbers.
2. Add up the digits of these numbers, then add the undoubled digits from the original number. Sum the two results.
3. If the total ends in a 0, the credit card number is valid, and it is invalid otherwise.

For example, using the number `5181 2710 9900 0012`:

1. Double the appropriate digits (**5**1**8**1 **2**7**1**0 **9**9**0**0 **0**0**1**2) to obtain the values: 10, 16, 4, 2, 18, 0, 0, 2.
2. Add up the digits of these values to get (1+0) + (1+6) + 4 + 2 + (1+8) + 0 + 0 + 2 = 25. The sum of the undoubled digits is 1+1+7+0+9+0+0+2 = 20, so the total is 20+25=45.
3. 45 does not end in a 0, so this credit card number is invalid.

For this problem, you must write a program that checks the validity of credit card numbers according to the Luhn algorithm.

Input

The input begins with a number N on a single line, followed by N lines each containing a single credit card number. Each credit card number consists of 16 decimal digits in groups of four separated by single spaces.

Output

The output consists of one line for each input credit card number. If the credit card number is valid, this line consists of the string '`Valid`', otherwise it reads '`Invalid`'.

Sample Input

```
2
5181 2710 9900 0012
5181 2710 9900 0017
```

Sample Output

```
Invalid
Valid
```