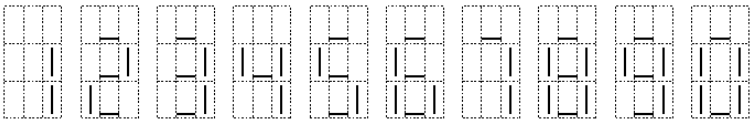
CodeWars 2007
Problem #13 -- Not Quite OCR
15 Points

JAVA programmers: your program name must be: Prob13.class
C programmers: your program name must be: prob13.exe

Task Description

Banks, always trying to increase their profit, asked their computer experts to come up with a system that can read bank checks; this would save money in check processing. One idea was to use optical character recognition (OCR) to recognize account numbers printed using 7 line segments.

Once a check has been scanned, image processing software would convert the horizontal and vertical bars to ASCII bars '|' and underscores '_'. The ASCII 7-segment versions of the ten digits look like this:



A bank account has a 9-digit account number with a checksum. For a valid account number, the following checksum equation holds: (d1 + 2 × d2 + 3 × d3 + . . . + 9 × d9) mod 11 = 0. Digits are numbered from right to left like this: d9d8d7d6d5d4d3d2d1.
Unfortunately, the scanner sometimes makes mistakes: some line segments may be missing. Your task is to write a program that deduces the original account number, assuming that:

> **Definition : mod**
> x mod y = 0 when x is divisible by y with no remainder.

- when the input represents a valid account number, it is the original number;
- at most one digit is garbled;
- garbling will NOT convert one valid digit to another (a garbled digit will never be a valid digit). Essentially, this means the check was misprinted.
- the scanned image contains no extra segments.

For example, the following input:

```
    _  _     _  _  _  _  _
  | _| _||_||_ |_   ||_||_|
  ||_  _|  | _||_|  ||_| _|
```

represents the account number "123456789".

Program Input

The input file starts with a line with one integer specifying the number of account numbers that have to be processed. Each account number occupies 3 lines of 27 characters (there are no extra spaces separating the digit representations).

Program Output

For each account number you process, your output should contain one line with 9 digits if the correct account number can be determined, the string "`failure`" if no solutions were found and "`ambiguous`" if more than one solution was found.

Sample Input

```
5
    _  _     _  _  _  _  _
  | _| _||_||_ |_   ||_||_|
  ||_  _|  | _||_|  ||_| _|

    _  _     _  _  _  _
  | _| _||_||_ |_   ||_|  |
  ||_  _|  | _||_|  ||_|  |

    _  _  _  _  _  _     _
|_||_|| || ||_   |  |  || |
  | _||_||_||_|  |  |  | _|

 _  _  _  _  _  _  _  _  _
|_||_||_||_||_||_||_||_||_|
|_||_||_||_||_||_||_||_||_|

 _  _  _  _  _  _  _  _  _
|_|  |_||_||_||_||_||_||_|
|_|  ||_||_||_||_||_||_||_|
```

Sample Output

```
123456789
failure
490067719
failure
878888888
```