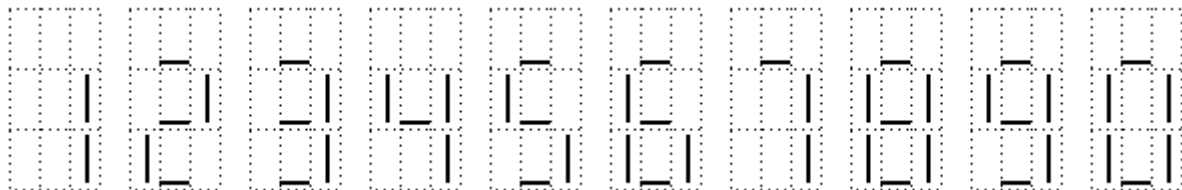


Banks, always trying to increase their profit, asked their computer experts to come up with a system that can read bank cheques; this would make the processing of cheques cheaper. One of their ideas was to use optical character recognition (ocr) to recognize bank accounts printed using 7 line-segments.

Once a cheque has been scanned, some 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 equation holds:  $(d_1 + 2 \times d_2 + 3 \times d_3 + \dots + 9 \times d_9) \bmod 11 = 0$ . Digits are numbered from right to left like this:  $d_9d_8d_7d_6d_5d_4d_3d_2d_1$ .

Unfortunately, the scanner sometimes makes mistakes: some line-segments may be missing. Your task is to write a program that deduces the original number, assuming that:

- when the input represents a valid account number, it is the original number;
- at most one digit is garbled;
- the scanned image contains no extra segments.

For example, the following input

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

used to be "123456789".

### 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.

### Output

For each test case, the output contains 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

```
4
| _ | _ || || _ _ | _ || || |
| _ _ | | _ || | || | | |
| _ || || || || _ | | | || _
| _ || || || || | | | | | | | | | | | |
| || || || || || || || || || ||
| || || || || || || || || || ||
| _ | || || || || || || || || ||
| _ | || || || || || || || || ||
```

### Sample Output

```
123456789
ambiguous
failure
878888888
```