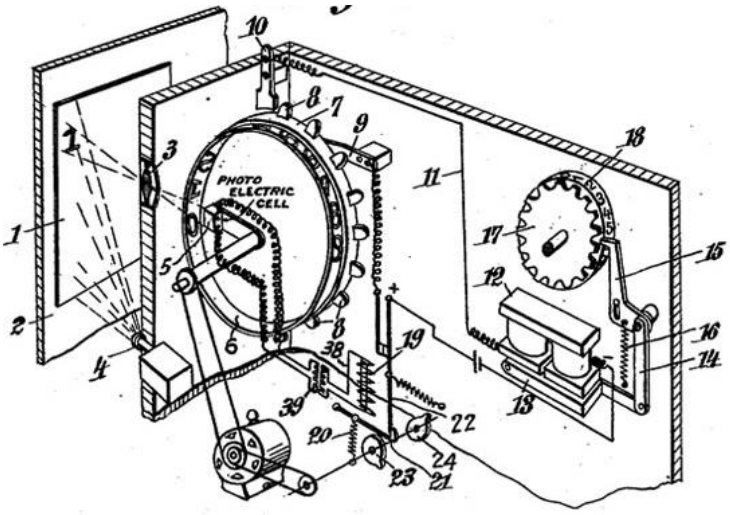
Organization Number OCR Kata

You work for a Norwegian company which has recently purchased an ingenious machine to assist in reading letters and faxes sent in by various other Norwegian companies. The machine scans the paper documents, extracts the *organization number* from each document and produces a file with a number of entries which each look like this:




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

Each entry is 4 lines long and each line has 27 characters. The first 3 lines of each entry contain an organization number written using pipes and underscores, and the fourth line is blank. Each organization number should have 9 digits, all of which should be in the range 0-9.

Task 1

Write a program that can take this file and parse it into actual organization numbers.

Having done that, you realize that the ingenious machine sometimes makes mistakes. The next step therefore is to validate that the numbers you read are in fact valid organization numbers. A valid organization number has a valid *checksum*.

The checksum (right-most digit) of a Norwegian organization number is calculated using a *weighted mod11 algorithm*:

Organization number without checksum:	9	4	3	5	7	4	5	3
Position names:	d_8	d_7	d_6	d_5	d_4	d_3	d_2	d_1

$$checksum = 11 - (2d_1 + 3d_2 + 4d_3 + 5d_4 + 6d_5 + 7d_6 + 2d_7 + 3d_8) \bmod 11$$
$$checksum = 11 - (6 + 15 + 16 + 35 + 30 + 21 + 8 + 27) \bmod 11$$
$$= 11 - 158 \bmod 11$$
$$= 11 - 4$$
$$= 7$$

Note that if the formula produces a checksum of $11 - 0 = 11$, the checksum digit is set to 0. If the formula produces a checksum of $11 - 1 = 10$, the checksum digit is *undefined*, meaning that the organization number is invalid.

Task 2

Enhance your program by adding checksum validation.

Your boss is keen to see your results. She asks you to write out a file of your findings, one for each input file, in this format:

```
943574537
974808843 ERR
985775??1 ILL
```

I.e. the file has one organization number per row. If some characters are illegible, they are replaced by a `?` and the suffix `ILL` is appended. In the case of a wrong checksum, the suffix `ERR` is appended.

Task 3

Make your program produce the files requested by your boss.

It turns out that often when a number comes back as `ERR` or `ILL`, it is because the scanner has failed to pick up on one pipe or underscore for one of the digits. For example

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

The 9 could be an 8 if the scanner had missed one `|`. Or the 0 could be an 8. Or the 1 could be a 7. The 5 could be a 9 or 6. Your boss wants your program to further process the `ERR` and `ILL` numbers and try to guess what they should be, by adding or removing *just one pipe or underscore*. If there is only one possible number with a valid checksum, then use that. If there are several options, append the suffix `AMB` followed by a comma-separated list of the possible numbers.

Task 4

Enhance you program with the guessing capabilities specified by your boss.

To relieve you of the tedious task of typing pipes and underscores for initial test cases, a set of OCR'ed organization numbers are available here:

http://tinyurl.com/org-nos