# Bank OCR

## Primera Historia de Usuario

Imagina que trabajas para un banco, que recientemente compró una máquina ingeniosa para ayudarlo a leer cartas y faxes enviados a las sucursales. La máquina escanea los documentos en papel y produce un archivo con una cantidad de entradas que se ven así:

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

Cada entrada se conforma de  4 líneas y cada línea tiene 27 caracteres. Las primeras 3 líneas de cada entrada contienen un número de cuenta escrito usando *pipes* y guiones bajos, y la cuarta línea está en blanco. Cada número de cuenta debe tener 9 dígitos, todos los cuales deben estar en el rango 0-9. Un archivo normal contiene alrededor de 500 entradas.

La primera tarea es escribir un programa que pueda tomar este archivo y convertirlo en números de cuenta reales.

## Segunda Historia de Usuario

Una vez hecho lo anterior,  rápidamente te das cuenta de que la ingeniosa máquina no es infalible. A veces comete errores en su escaneo. En consecuencia, el siguiente paso es validar que los números que lee son, de hecho, números de cuenta válidos. Se sabe que un número de cuenta válido tiene una suma de verificación válida y esto se puede calcular de la siguiente manera:

```
account number:  3  4  5  8  8  2  8  6  5
position name:  d9 d8 d7 d6 d5 d4 d3 d2 d1

checksum calculation:
(1*d1 + 2*d2 + 3*d3 + … + 9*d9) mod 11 = 0
```

Se necesita escribir un programa que calcule la suma de verificación para un número de cuenta determinado e identifique si se trata de un número de cuenta válido.

## Tercera Historia de Usuario

Su jefe está ansioso por ver sus resultados. Él le pide que escriba un archivo de sus hallazgos, con una línea para cada número de cuenta, en este formato:

```
457508000 OK
664371495 ERR
86110??36 ILL
```

Es decir, el archivo tiene un número de cuenta por fila. Si algunos caracteres son ilegibles, se reemplazan por un '?'. En el caso de que alguna suma de verificación sea incorrecta o existe algún número ilegible, este estado se indica en una segunda columna.

## Pistas

La recomendación es encontrar una forma de escribir celdas 3x3 en 3 líneas en su código, para que formen dígitos identificables. Incluso si el código en realidad no los representa de esa manera internamente. Es preferible leer:

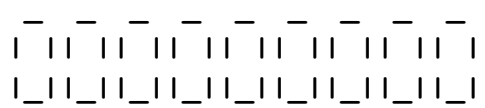
```
    "    " +
    "|_|" +
    "  |"
```

en lugar de:

```
    "    |_|   |"
```

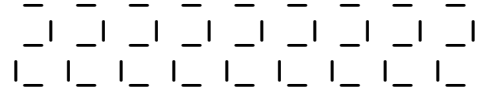
## Casos de prueba sugeridos

**1a historia de usuario**

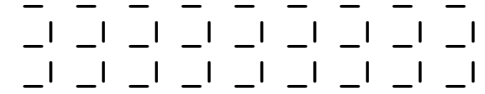


**=> 000000000 OK**

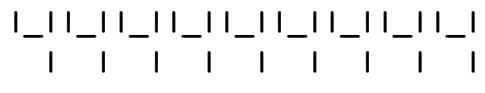
**=> 111111111 OK**



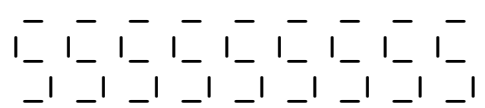
**=> 222222222 OK**



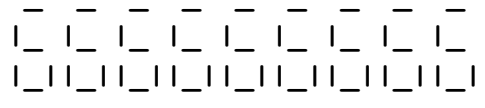
**=> 333333333 OK**



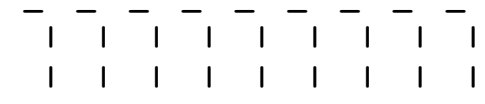
**=> 444444444 OK**



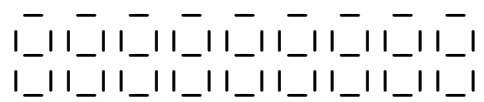
**=> 555555555 OK**



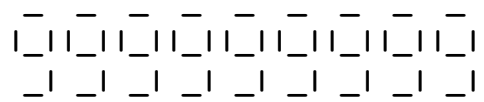
**=> 666666666 OK**



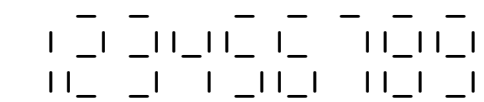
**=> 777777777 OK**



**=> 888888888 OK**

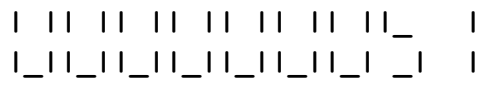


**=> 999999999 OK**

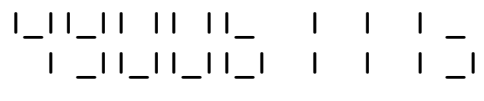


**=> 123456789 ERR**

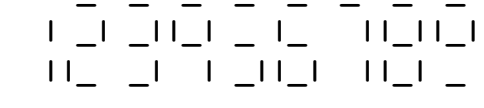
**3a historia de usuario**



**=> 000000051 ERR**



**=> 49006771? ILL**



**=> 1234?678? ILL**