An extra row and column, including an extra corner piece is appended like so:

$$\begin{array}{cccc|c}
0 & 1 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 1 \\
0 & 1 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 & 1 \\
\hline
0 & 1 & 0 & 0 & 1
\end{array}$$

Each of the extra bits documents the pairty of its row.

4 The Hamming code

The Hamming code is a binary encoding scheme that uses parity to detect, and correct errors. The insertion of “parity bits” is a common practice in basic encoding. Parity refers to the “oddness” or “evenness” of some data. Commonly, this is determined by the sum of the data modulo 2. For example, “00101” results in a parity bit of 0, because the sum of all the bits is 2, which has a remainder of 0 when divided by 2 (is equal to 0 mod 2).

5 Implementing the Hamming code

The script implementing a simple binary Hamming code is as follows:

```python
#!/usr/bin/env python3

"""
Hamming encoding framework for binary objects, using even parity.
"""

from itertools import count, takewhile

def powers_to(n):
    return takewhile(lambda x: x < n, (1 << i for i in count()))

def hamming_encode(bin_stream):
    pwr = 1
    out = []

    for bit in bin_stream:
        while len(out) + 1 == pwr:
            pwr <<= 1
            out.append(0)
        out.append(bit)

    for i in powers_to(len(out)):
        out[i - 1] = 1 & sum(out[pbit] for pstart in range(i - 1, len(out), i << 1)
                                       for pbit in range(pstart, pstart + i))
    return out
```

Listing 1: Binary Hamming code in Python

This code is accompanied by the following testing scheme:

```python
"""
Unit tests for binary_hamming.py
"""

import unittest
```