Parity index	00001	00010	00100	01000	10000
Coverage	00011	00011	00101	01001	10001
	00101	00110	00110	01010	10010
	00111	00111	00111	01011	10011
	01001	01010	01100	01100	10100
	01011	01011	01101	01101	10101
	01101	01110	01110	01110	10110
	01111	01111	01111	01111	10111
	10001	10010	10100	11000	11000
	10011	10011	10101	11001	11001
	10101	10110	10110	11010	11010
	10111	10111	10111	11011	11011
	11001	11010	11100	11100	11100
	11011	11011	11101	11101	11101
	11101	11110	11110	11110	11110
	11111	11111	11111	11111	11111

Table 2: Indices covered by each parity bit shown in binary

The second is shown in figure 2. It represents each covered bit as a filled in square, and each non-covered bit as an empty square, so the whole codeword is shown in every row.

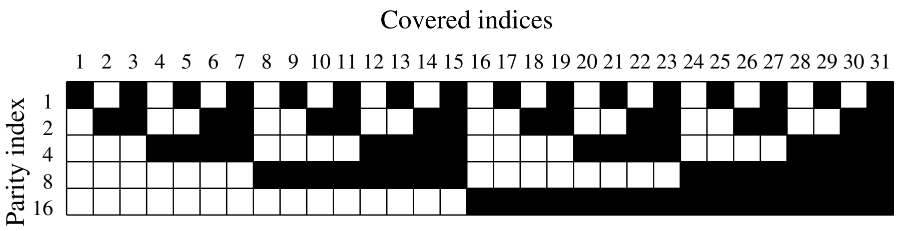

Figure 2: Index coverage of Hamming parity bits

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

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


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

# imports the "count" and "takewhile" functions
from itertools import count, takewhile

# function to get all of the powers of 2 up to a given upper limit.
# Uses count() to produce the set of natural numbers (0, 1, 2, 3..)
# and takewhile() to keep taking powers of 2 until they exceed the limit.
def powers_to(n):
    return takewhile(lambda x: x < n, (1 << i for i in count()))

# function that generates the particular indices covered by a parity bit
```