Ejercicio 7.7.3

El programa siguiente genera la función

f:{1, 2, … , 12} → f:{1, 2, … , 12}

definida por

$f(x) = 2^x \bmod 13$:

```
def encode():
    encoded_data = []
    for i in range(16):
        encoded_data = encoded_data + [(2**i)%13]
    return encoded_data
```

Es una permutación ya que $2^{12} = 315 \cdot 13 + 1 \equiv 1 \bmod 13$. Por tanto el conjunto $\{2^0, 2^1, 2^2, \dots, 2^{11}\}$ es el grupo multiplicativo de los naturales módulo 13.

La sentencia siguiente permite introducir el mensaje encriptado (M) por el teclado:

```
M = int(input("mensaje codificado(entre 1 y 12:"))
```

A continuación, se genera la lista `truth_table` que define la función $F(x)$ del comentario 7.4:

```
def table(M):
    t = encode()
    for i in range(16):
        if i == 0 or i > 12:
            t[i] = 0
        elif t[i] == M:
            t[i] = 1
        else:
            t[i] = 0
    return t

truth_table = table(M)
```

Queda por añadir el programa del ejemplo 7.10 con la lista truth_table que acaba de ser generada.

Ejecútese el programa con varios valores de M:

```
mensaje codificado(entre 1 y 12):8
Counter({3: 97, 0: 1, 10: 1, 12: 1})
```

```
mensaje codificado(entre 1 y 12):12
Counter({6: 99, 8: 1})

mensaje codificado(entre 1 y 12):3
Counter({4: 93, 3: 2, 9: 1, 10: 1, 8: 1, 6: 1, 2: 1})


mensaje codificado(entre 1 y 12):7
Counter({11: 96, 14: 2, 4: 1, 1: 1})
```

Queda por comprobar que

$f(8) = 3, f(12) = 6, f(3) = 4, f(7) = 11.$