Source Code 1. Generator, der einen (klassischen) Gray–Code-Iterator liefert.

```
def classical_gray_code(maximum=8):
    """Generator-Funktion: Liefert Iterator, der einen (klassischen) Gray-Code
    erzeugt"""
    subset = set()
    # Als Binärzahl ist das also 0000...
    while maximum > 0:
        maximum-= 1
        # Der Befehl "yield" wirkt wie "return", macht aber aus der Funktion einen
        # "Generator", die einen "Iterator" liefert:
        yield subset
        element_to_flip = (min(subset) + 1) if (len(subset) % 2) else 1
        # Die Symmetrische Differenz zweier Mengen A und B ist
        # (A vereinigt B) ohne (A geschnitten mit B):
        subset = subset.symmetric_difference(set([element_to_flip]))
```

Die Binärzahlen eines Gray-Codes interpretieren wir hier als charakteristische Funktionen (siehe Definition ??) von Teilmengen. Der Datentyp `set` in Python entspricht dem mathematischen Begriff einer (natürlich endlichen!) Menge.