Warunki

1. W implementacjach można korzystać tylko z elementarnych konstrukcji Python'a (funkcje, instrukcje warunkowe, pętle, `range`, klasy użyte do definiowania struktur danych). **Nie wolno korzystać ze słowników i zbiorów, itp. Wolno korzystać z:**
   (a) wbudowanego sortowania,
   (b) wbudowanej kolejki (`deque` z biblioteki `collections`),
   (c) wbudowanej kolejki priorytetowej (`PriorityQueue` z biblioteki `queue`).
2. Rozwiązania muszą być efektywne obliczeniowo (także w zadaniach, w których nie podajemy wprost ograniczenia na złożoność obliczeniową). Zadania o zbyt wysokiej złożoności będą otrzymywały obniżone oceny (lub 0).
3. Rozwiązania zadań proszę umieszczać w załączonych plikach (`zadX.py`).

Zadanie 1 (cykl Eulera)

Dany jest graf nieskierowany reprezentowany przez macierz sąsiedztwa G, w której pole $G[i][j]$ ma wartość `True` wtedy gdy istnieje krawędź między wierzchołkami i oraz j (wartość `False` oznacza brak krawędzi). Wierzchołkami są liczby naturalne $0, 1, \ldots, n-1$, gdzie $n = \texttt{len}(G)$. Proszę zaimplementować funkcję:

```
euler(G)
```

która sprawdza czy G posiada cykl Eulera i jeśli tak, to zwraca listę z kolejnymi numerami wierzchołków na takim cyklu, a jeśli nie to zwraca `None`. Funkcja nie powinna niszczyć macierzy G.

Przykład

Dla grafu reprezentowanego przez macierz:

```
G = [[0,1,1,0,0,0],
     [1,0,1,1,0,1],
     [1,1,0,0,1,1],
     [0,1,0,0,0,1],
     [0,0,1,0,0,1],
     [0,1,1,1,1,0]]
```

wynikiem może być, między innymi:

```
[0,1,5,4,2,1,3,5,2,0]
```