LCS

```
def LCS(s, t):
    n = len(s) + 1
    m = len(t) + 1
    memo = [[0] * m for _ in range(n)]

    for i in range(1, n):
        for j in range(1, m):
            if s[i - 1] == t[j - 1]:
                memo[i][j] = memo[i - 1][j - 1] + 1
            else:
                memo[i][j] = max(memo[i - 1][j], memo[i][j - 1])

    return memo[-1][-1]

def twin_towers():
    twin_tower_number = 1

    while True:
        N1, N2 = map(int, input().split())

        if N1 == 0 and N2 == 0:
            break

        tower1 = list(map(int, input().split()))
        tower2 = list(map(int, input().split()))

        num_tiles = LCS(tower1, tower2)

        print(f"Twin Towers #{twin_tower_number}")
        print(f"Number of Tiles : {num_tiles}\n")

        twin_tower_number += 1

twin_towers()
```

#55226341 | Nack34's solution for [UVA-10066]

Status	Time	Length	Lang	Submitted	Open	Share text	RemoteRunId
Accepted	50ms	803	PYTH3 3.5.1	2024-10-19 15:30:58	☑	☐	29895487

Copy Python

```python
def LCS(s, t):
    n = len(s) + 1
    m = len(t) + 1
    memo = [[0] * m for _ in range(n)]

    for i in range(1, n):
        for j in range(1, m):
            if s[i - 1] == t[j - 1]:
                memo[i][j] = memo[i - 1][j - 1] + 1
            else:
                memo[i][j] = max(memo[i - 1][j], memo[i][j - 1])

    return memo[-1][-1]

def twin_towers():
    twin_tower_number = 1

    while True:
        N1, N2 = map(int, input().split())

        if N1 == 0 and N2 == 0:
            break

        tower1 = list(map(int, input().split()))
        tower2 = list(map(int, input().split()))

        num_tiles = LCS(tower1, tower2)

        print(f"Twin Towers #{twin_tower_number}")
        print(f"Number of Tiles : {num_tiles}\n")

        twin_tower_number += 1

twin_towers()
```