Explanation:

1. **Input Reading:**
    - The program reads the number of students (and topics) nnn.
    - It then reads an n×nn \times nn×n preference matrix where preference[i][j] is 1 if student iii likes topic jjj, and 0 otherwise.
2. **Dynamic Programming (DP) Setup:**
    - We use a bitmask to represent subsets of topics.
    - dp[mask] will store the number of valid ways to assign topics corresponding to the set bits in mask.
3. **Main DP Logic:**
    - For each mask (representing a subset of topics), we calculate the number of valid assignments.
    - k is the number of students considered in the current subset (determined by the number of set bits in mask).
    - We iterate through all topics, checking if they are in the current subset (mask & (1 << j)) and if the current student likes this topic (preference[k][j]).
    - If both conditions are met, we add the number of valid assignments from the previous state (dp[mask ^ (1 << j)]).
4. **Result Output:**
    - The result for all students and topics is stored in dp[(1 << n) - 1] which represents all topics being assigned.

This approach ensures that we efficiently count all valid assignments using dynamic programming and bitmasking.