|  |  |
| --- | --- |
| **Coin Change Permutation in C++** | |
| #include <iostream>  #include <vector>  using namespace std;  int main() {      vector<int> coins = {2, 3, 5};      int tar = 7;      vector<int> dp(tar + 1, 0);      dp[0] = 1; // Base case: 1 way to make amount 0 (using no coins)      for (int amt = 1; amt <= tar; amt++) {          for (int coin : coins) {              if (coin <= amt) {                  int ramt = amt - coin;                  dp[amt] += dp[ramt];              }          }      }      cout << dp[tar] << endl; // Output the number of permutations to make the target amount      return 0;  } | **Initial dp Array**  Before processing:  dp = [1, 0, 0, 0, 0, 0, 0, 0] // (Indexes represent amounts from 0 to 7)  **Dry Run with Iteration Table**  **Iterating over amt from 1 to 7**   | **amt** | **Coin Used** | **dp[amt] = dp[amt] + dp[amt - coin]** | **Updated dp** | | --- | --- | --- | --- | | 1 | 2 (skipped) | - | [1, 0, 0, 0, 0, 0, 0, 0] | |  | 3 (skipped) | - |  | |  | 5 (skipped) | - |  | | 2 | 2 | dp[2] += dp[0] = 1 | [1, 0, 1, 0, 0, 0, 0, 0] | |  | 3, 5 (skipped) | - |  | | 3 | 2 | dp[3] += dp[1] = 0 | [1, 0, 1, 0, 0, 0, 0, 0] | |  | 3 | dp[3] += dp[0] = 1 | [1, 0, 1, 1, 0, 0, 0, 0] | |  | 5 (skipped) | - |  | | 4 | 2 | dp[4] += dp[2] = 1 | [1, 0, 1, 1, 1, 0, 0, 0] | |  | 3 | dp[4] += dp[1] = 0 | [1, 0, 1, 1, 1, 0, 0, 0] | |  | 5 (skipped) | - |  | | 5 | 2 | dp[5] += dp[3] = 1 | [1, 0, 1, 1, 1, 1, 0, 0] | |  | 3 | dp[5] += dp[2] = 1 | [1, 0, 1, 1, 1, 2, 0, 0] | |  | 5 | dp[5] += dp[0] = 1 | [1, 0, 1, 1, 1, 3, 0, 0] | | 6 | 2 | dp[6] += dp[4] = 1 | [1, 0, 1, 1, 1, 3, 1, 0] | |  | 3 | dp[6] += dp[3] = 1 | [1, 0, 1, 1, 1, 3, 2, 0] | |  | 5 | dp[6] += dp[1] = 0 | [1, 0, 1, 1, 1, 3, 2, 0] | | 7 | 2 | dp[7] += dp[5] = 3 | [1, 0, 1, 1, 1, 3, 2, 3] | |  | 3 | dp[7] += dp[4] = 1 | [1, 0, 1, 1, 1, 3, 2, 4] | |  | 5 | dp[7] += dp[2] = 1 | [1, 0, 1, 1, 1, 3, 2, 5] |   **Final dp Array**  After processing all amounts:  dp = [1, 0, 1, 1, 1, 3, 2, 5]  **Final Output**  5  This means **there are 5 different permutations to form amount 7 using {2, 3, 5}**:   1. **2 + 2 + 3** 2. **2 + 3 + 2** 3. **3 + 2 + 2** 4. **2 + 5** 5. **5 + 2** |
| Output:- 5 | |