|  |  |
| --- | --- |
| **Coin Change Combination in C++** | |
| #include <iostream>  #include <vector>  using namespace std;  int main() {      vector<int> arr = {2, 3, 5};      int amt = 7;      vector<int> dp(amt + 1, 0);      dp[0] = 1; // Base case: 1 way to make amount 0 (using no coins)      for (int i = 0; i < arr.size(); i++) {          for (int j = arr[i]; j <= amt; j++) {              dp[j] += dp[j - arr[i]];          }      }      cout << dp[amt] << endl; // Output the number of combinations for amount `amt`      return 0;  } | ****Initial**** dp ****Array**** Before processing:  arr=[2, 3, 5]  dp = [1, 0, 0, 0, 0, 0, 0, 0]  (Index represents amount: 0 to 7) ****Dry Run with Iteration Table********Processing coin**** 2  | **j (amt)** | **dp[j] = dp[j] + dp[j - 2]** | **Updated dp** | | --- | --- | --- | | 2 | dp[2] += dp[0] = 1 | [1, 0, 1, 0, 0, 0, 0, 0] | | 3 | dp[3] += dp[1] = 0 | [1, 0, 1, 0, 0, 0, 0, 0] | | 4 | dp[4] += dp[2] = 1 | [1, 0, 1, 0, 1, 0, 0, 0] | | 5 | dp[5] += dp[3] = 0 | [1, 0, 1, 0, 1, 0, 0, 0] | | 6 | dp[6] += dp[4] = 1 | [1, 0, 1, 0, 1, 0, 1, 0] | | 7 | dp[7] += dp[5] = 0 | [1, 0, 1, 0, 1, 0, 1, 0] |  ****Processing coin**** 3  | **j (amt)** | **dp[j] = dp[j] + dp[j - 3]** | **Updated dp** | | --- | --- | --- | | 3 | dp[3] += dp[0] = 1 | [1, 0, 1, 1, 1, 0, 1, 0] | | 4 | dp[4] += dp[1] = 0 | [1, 0, 1, 1, 1, 0, 1, 0] | | 5 | dp[5] += dp[2] = 1 | [1, 0, 1, 1, 1, 1, 1, 0] | | 6 | dp[6] += dp[3] = 1 | [1, 0, 1, 1, 1, 1, 2, 0] | | 7 | dp[7] += dp[4] = 1 | [1, 0, 1, 1, 1, 1, 2, 1] |  ****Processing coin**** 5  | **j (amt)** | **dp[j] = dp[j] + dp[j - 5]** | **Updated dp** | | --- | --- | --- | | 5 | dp[5] += dp[0] = 1 | [1, 0, 1, 1, 1, 2, 2, 1] | | 6 | dp[6] += dp[1] = 0 | [1, 0, 1, 1, 1, 2, 2, 1] | | 7 | dp[7] += dp[2] = 1 | [1, 0, 1, 1, 1, 2, 2, 2] |  ****Final**** dp ****Array**** After processing all coins:  dp = [1, 0, 1, 1, 1, 2, 2, 2] ****Final Output**** 2  This means **there are 2 ways to form amount 7 using {2, 3, 5}**:   1. **2 + 2 + 3** 2. **2 + 5** |
| **Output:-**  2 | |