|  |  |
| --- | --- |
| **Count Distinct Subsequence C++** | |
| #include <iostream>  using namespace std;  int countValleysAndMountains(int n) {      int dp[n + 1] = {0}; // Initialize the array with zeros      dp[0] = 1; // Base case: empty sequence      dp[1] = 1; // Sequence of length 1: either V or M      for (int i = 2; i <= n; i++) {          int valleys = 0;          int mountains = i - 1;          while (mountains >= 0) {              dp[i] += dp[valleys] \* dp[mountains];              valleys++;              mountains--;          }      }      return dp[n];  }  int main() {      int n = 5;      cout << countValleysAndMountains(n) << endl;      return 0;  } | ****Step-by-Step Calculation****  | **i** | **dp[i] Computation** | **dp[i] Value** | | --- | --- | --- | | 0 | dp[0] = 1 | **1** | | 1 | dp[1] = dp[0] \* dp[0] | **1** | | 2 | dp[2] = dp[0] \* dp[1] + dp[1] \* dp[0] | **2** | | 3 | dp[3] = dp[0] \* dp[2] + dp[1] \* dp[1] + dp[2] \* dp[0] | **5** | | 4 | dp[4] = dp[0] \* dp[3] + dp[1] \* dp[2] + dp[2] \* dp[1] + dp[3] \* dp[0] | **14** | | 5 | dp[5] = dp[0] \* dp[4] + dp[1] \* dp[3] + dp[2] \* dp[2] + dp[3] \* dp[1] + dp[4] \* dp[0] | **42** |  **Final Output** 42 |
| Output:- 42 | |