|  |  |
| --- | --- |
| **Balanced Parenthesis in C++** | |
| #include <iostream>  #include <vector>  using namespace std;  int main() {      int n = 5;      vector<int> dp(n + 1, 0);      dp[0] = 1;      dp[1] = 1;      for (int i = 2; i <= n; i++) {          int inside = i - 1;          int outside = 0;          while (inside >= 0) {              dp[i] += dp[inside] \* dp[outside];              inside--;              outside++;          }      }      for (int i = 0; i < dp.size(); i++) {          cout << dp[i] << " ";      }      // char c = 'b';      // cout << (c - '0') << endl;      return 0;  } | ****Dry Run with Table**** Let's analyze **step-by-step calculations for n = 5**. ****Initialization****  | **i** | **inside** | **outside** | **Computation** | **dp[i]** | | --- | --- | --- | --- | --- | | 0 | - | - | dp[0] = 1 | 1 | | 1 | - | - | dp[1] = 1 | 1 |  ****Filling**** dp ****Array****  | **i** | **inside** | **outside** | **Computation (dp[i] += dp[inside] \* dp[outside])** | **dp[i]** | | --- | --- | --- | --- | --- | | 2 | 1 | 0 | dp[2] += dp[1] \* dp[0] = 1 \* 1 | 1 | |  | 0 | 1 | dp[2] += dp[0] \* dp[1] = 1 \* 1 | 2 | | 3 | 2 | 0 | dp[3] += dp[2] \* dp[0] = 2 \* 1 | 2 | |  | 1 | 1 | dp[3] += dp[1] \* dp[1] = 1 \* 1 | 3 | |  | 0 | 2 | dp[3] += dp[0] \* dp[2] = 1 \* 2 | 5 | | 4 | 3 | 0 | dp[4] += dp[3] \* dp[0] = 5 \* 1 | 5 | |  | 2 | 1 | dp[4] += dp[2] \* dp[1] = 2 \* 1 | 7 | |  | 1 | 2 | dp[4] += dp[1] \* dp[2] = 1 \* 2 | 9 | |  | 0 | 3 | dp[4] += dp[0] \* dp[3] = 1 \* 5 | 14 | | 5 | 4 | 0 | dp[5] += dp[4] \* dp[0] = 14 \* 1 | 14 | |  | 3 | 1 | dp[5] += dp[3] \* dp[1] = 5 \* 1 | 19 | |  | 2 | 2 | dp[5] += dp[2] \* dp[2] = 2 \* 2 | 23 | |  | 1 | 3 | dp[5] += dp[1] \* dp[3] = 1 \* 5 | 28 | |  | 0 | 4 | dp[5] += dp[0] \* dp[4] = 1 \* 14 | 42 |  ****Final**** dp ****Array Output**** 1 1 2 5 14 42 ****Final Output (****dp[5]****)**** 42  This means **42 unique BSTs** can be formed using 5 nodes. |
| Output:- 1 1 2 5 14 42 | |