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;
}
```

Initial Setup:

- n = 5
- dp is a vector of size n + 1 = 6, initially set to {1, 1, 0, 0, 0, 0}.

Loop Breakdown:

Iteration 1: i = 2

1. inside = 2 - 1 = 1
2. outside = 0

   For inside = 1 and outside = 0:

   - dp[2] += dp[1] * dp[0] → dp[2] += 1 * 1 → dp[2] = 1.

   Now, decrease inside to 0 and increase outside to 1.

   For inside = 0 and outside = 1:

   - dp[2] += dp[0] * dp[1] → dp[2] += 1 * 1 → dp[2] = 2.

   So, after this iteration, dp[2] = 2.

Iteration 2: i = 3

1. inside = 3 - 1 = 2
2. outside = 0

   For inside = 2 and outside = 0:

   - dp[3] += dp[2] * dp[0] → dp[3] += 2 * 1 → dp[3] = 2.

   Now, decrease inside to 1 and increase outside to 1.

   For inside = 1 and outside = 1:

   - dp[3] += dp[1] * dp[1] → dp[3] += 1 * 1 → dp[3] = 3.

   Now, decrease inside to 0 and increase outside to 2.

   For inside = 0 and outside = 2:

   - dp[3] += dp[0] * dp[2] → dp[3] += 1 * 2 → dp[3] = 5.

   So, after this iteration, dp[3] = 5.

Iteration 3: i = 4

1. inside = 4 - 1 = 3
2. outside = 0

   For inside = 3 and outside = 0:

   - dp[4] += dp[3] * dp[0] → dp[4] += 5 * 1 → dp[4] = 5.

   Now, decrease inside to 2 and increase outside to 1.

   For inside = 2 and outside = 1:

   - dp[4] += dp[2] * dp[1] → dp[4] += 2 * 1 → dp[4] = 7.

   Now, decrease inside to 1 and increase outside to 2.

   For inside = 1 and outside = 2:

   - dp[4] += dp[1] * dp[2] → dp[4] += 1 * 2 → dp[4] = 9.

   Now, decrease inside to 0 and increase outside to 3.

   For inside = 0 and outside = 3:

   - dp[4] += dp[0] * dp[3] → dp[4] += 1 * 5 → dp[4] = 14.

   So, after this iteration, dp[4] = 14.

Iteration 4: i = 5

1. inside = 5 - 1 = 4
2. outside = 0

   For inside = 4 and outside = 0:

   - dp[5] += dp[4] * dp[0] → dp[5] += 14 * 1 → dp[5] = 14.

   Now, decrease inside to 3 and increase outside to 1.

   For inside = 3 and outside = 1:

   - dp[5] += dp[3] * dp[1] → dp[5] += 5 * 1 → dp[5] = 19.

   Now, decrease inside to 2 and increase outside to 2.

For inside = 2 and outside = 2:

- dp[5] += dp[2] * dp[2] → dp[5] += 2 * 2 → dp[5] = 23.

Now, decrease inside to 1 and increase outside to 3.

For inside = 1 and outside = 3:

- dp[5] += dp[1] * dp[3] → dp[5] += 1 * 5 → dp[5] = 28.

Now, decrease inside to 0 and increase outside to 4.

For inside = 0 and outside = 4:

- dp[5] += dp[0] * dp[4] → dp[5] += 1 * 14 → dp[5] = 42.

So, after this iteration, dp[5] = 42.

Final Output:

The dp array is:

Copy code

```
1 1 2 5 14 42
```

Output:-

```
1 1 2 5 14 42
```