Print all path with minimum Cost In C++

```cpp
#include <iostream>
#include <vector>
#include <algorithm>

using namespace std;
int solution(vector<int>& prices) {
    vector<int> np(prices.size() + 1);
    for (int i = 0; i < prices.size(); i++) {
        np[i + 1] = prices[i];
    }

    vector<int> dp(np.size());
    dp[0] = 0;
    dp[1] = np[1];

    for (int i = 2; i < dp.size(); i++) {
        dp[i] = np[i];

        int li = 1;
        int ri = i - 1;
        while (li <= ri) {
            if (dp[li] + dp[ri] > dp[i]) {
                dp[i] = dp[li] + dp[ri];
            }

            li++;
            ri--;
        }
    }

    return dp[dp.size() - 1];
}

int main() {
    vector<int> prices = {1, 5, 8, 9, 10, 17,
17, 20};

    cout << solution(prices) << endl;

    return 0;
}
```

Dry Run of the Code

Given prices = {1, 5, 8, 9, 10, 17, 17, 20} (rod lengths from 1 to 8):

- **Step 1**: Initialize np and dp:
  - np = {0, 1, 5, 8, 9, 10, 17, 17, 20}
  - dp = {0, 1, 0, 0, 0, 0, 0, 0, 0}
- **Step 2**: Start filling dp:
  - For i = 2 (rod length 2):
    - dp[2] = np[2] = 5
    - Check splits: 1 + 4 = 5 (no better than dp[2] = 5)
  - For i = 3 (rod length 3):
    - dp[3] = np[3] = 8
    - Check splits: 1 + 7 = 8, 5 + 3 = 8 (no better than dp[3] = 8)
  - For i = 4 (rod length 4):
    - dp[4] = np[4] = 9
    - Check splits: 1 + 8 = 9, 5 + 4 = 9 (no better than dp[4] = 9)
  - For i = 5 (rod length 5):
    - dp[5] = np[5] = 10
    - Check splits: 1 + 9 = 10, 5 + 5 = 10, 8 + 2 = 10 (no better than dp[5] = 10)
  - For i = 6 (rod length 6):
    - dp[6] = np[6] = 17
    - Check splits: 1 + 16 = 17, 5 + 12 = 17, 8 + 9 = 17, 9 + 8 = 17, 10 + 7 = 17 (no better than dp[6] = 17)
  - For i = 7 (rod length 7):
    - dp[7] = np[7] = 17
    - Check splits: 1 + 16 = 17, 5 + 12 = 17, 8 + 9 = 17, 9 + 8 = 17, 10 + 7 = 17, 17 + 0 = 17
  - For i = 8 (rod length 8):
    - dp[8] = np[8] = 20
    - Check splits: 1 + 19 = 20, 5 + 15 = 20, 8 + 12 = 20, 9 + 11 = 20, 10 + 10 = 20, 17 + 3 = 20, 17 + 3 = 20
- **Step 3**: After filling all values, the maximum revenue is found at dp[8] = 22.

Output:-

```
22
```