|  |  |
| --- | --- |
| **Rod cutting In C++** | |
| #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 (Tabular)********Given Prices**** Length: 1 2 3 4 5 6 7 8  Prices: 1 5 8 9 10 17 17 20 ****DP Computation Table****  | **Rod Length (i)** | **Price (np[i])** | **Possible Cuts (li, ri)** | **Best Revenue (dp[i])** | | --- | --- | --- | --- | | 1 | 1 | (1) | 1 | | 2 | 5 | (1,1) → 1+1=2 | 5 | | 3 | 8 | (1,2) → 1+5=6, (2,1) → 5+1=6 | 8 | | 4 | 9 | (1,3) → 1+8=9, (2,2) → 5+5=10 | 10 | | 5 | 10 | (1,4) → 1+10=11, (2,3) → 5+8=13 | 13 | | 6 | 17 | (1,5) → 1+13=14, (2,4) → 5+10=15, (3,3) → 8+8=16 | 17 | | 7 | 17 | (1,6) → 1+17=18, (2,5) → 5+13=18, (3,4) → 8+10=18 | 18 | | 8 | 20 | (1,7) → 1+18=19, (2,6) → 5+17=22, (3,5) → 8+13=21, (4,4) → 10+10=20 | 22 |  Final answer The maximum revenue we can get for **length = 8** is **22**. |
| Output:-  22 | |