Problem Statement : 0/1 Knapsack by Dynamic Programming

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

using namespace std;

int main() {
    int n, W;
    cin >> n >> W;

    vector<int> profits(n);
    vector<int> weights(n);

    for (int i = 0; i < n; ++i) {
        cin >> profits[i] >> weights[i];
    }

    vector<vector<int>> dp(n + 1, vector<int>(W + 1, 0));

   for (int i = 1; i <= n; ++i) {
        for (int w = 1; w <= W; ++w) {
            int current_weight = weights[i - 1];
            int current_profit = profits[i - 1];

            if (current_weight > w) {
                dp[i][w] = dp[i - 1][w];
            } else {
                dp[i][w] = max(dp[i - 1][w], current_profit + dp[i - 1][w -
current_weight]);
            }
        }
    }

    cout << dp[n][W] << endl;
 return 0;
 }
```