

## Dynamic Programming: Implement 0/1 Knapsack problem

```
#include <iostream>

#include <vector>

#include <algorithm>

using namespace std;

// Function to solve 0/1 Knapsack using Dynamic Programming
int knapsack(int W, const vector<int> &wt, const vector<int> &val, int n) {

    vector<vector<int>> > dp(n + 1, vector<int>(W + 1, 0)); // note the space between >>

    // Build table dp[][] in bottom-up manner
    for (int i = 1; i <= n; i++) {
        for (int w = 1; w <= W; w++) {
            if (wt[i - 1] <= w) {
                dp[i][w] = max(val[i - 1] + dp[i - 1][w - wt[i - 1]], dp[i - 1][w]);
            } else {
                dp[i][w] = dp[i - 1][w];
            }
        }
    }

    return dp[n][W];
}

int main() {

    // old-style vector initialization
    int val_arr[] = {60, 100, 120};
    int wt_arr[] = {10, 20, 30};
```

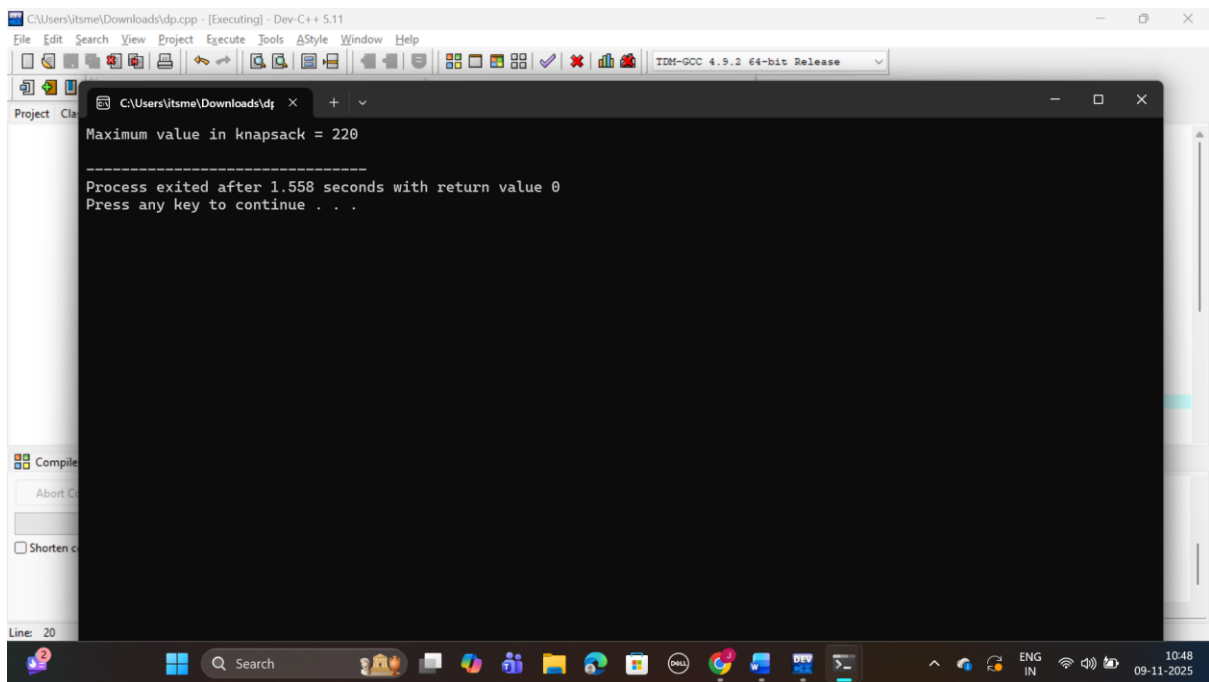
```
int n = sizeof(val_arr) / sizeof(val_arr[0]);

int W = 50;

vector<int> val(val_arr, val_arr + n);
vector<int> wt(wt_arr, wt_arr + n);

cout << "Maximum value in knapsack = " << knapsack(W, wt, val, n) << endl;

return 0;
}
```



```
C:\Users\itsme\Downloads\ydp.cpp - [Executing] - Dev-C++ 5.11
File Edit Search View Project Execute Tools AStyle Window Help
Maximum value in knapsack = 220
Process exited after 1.558 seconds with return value 0
Press any key to continue . . .
Line: 20
```