Target sum Subset in C++

```
#include <iostream>
#include <vector>
using namespace std;

bool targetSumSubsets(vector<int>& arr, int target) {
    int n = arr.size();
    vector<vector<bool>> dp(n + 1, vector<bool>(target + 1, false));

    for (int i = 0; i <= n; i++) {
        for (int j = 0; j <= target; j++) {
            if (i == 0 && j == 0) {
                dp[i][j] = true;
            } else if (i == 0) {
                dp[i][j] = false;
            } else if (j == 0) {
                dp[i][j] = true;
            } else {
                if (dp[i - 1][j]) {
                    dp[i][j] = true;
                } else {
                    int val = arr[i - 1];
                    if (j >= val && dp[i - 1][j - val]) {
                        dp[i][j] = true;
                    }
                }
            }
        }
    }
    return dp[n][target];
}

int main() {
    vector<int> arr = {4, 2, 7, 1, 3};
    int target = 10;

    if (targetSumSubsets(arr, target)) {
        cout << "True" << endl;
    } else {
        cout << "False" << endl;
    }

    return 0;
}
```

Dry Run

Input:

- Array: arr = {4, 2, 7, 1, 3}
- Target: target = 10

Steps:

1. **Initialize DP Table**:
   - dp has dimensions (n+1) × (target+1), i.e., 6 × 11 (since n = 5 and target = 10).
2. **Fill the DP Table**:
   - Start filling the table row by row, column by column.

DP Table Construction

Initial DP Table:

dp[i][j] = false for all i, j

Base Cases:

- dp[i][0] = true for all i.
- dp[0][j] = false for j > 0.

DP Transitions:

- **Row 1 (i = 1, element = 4)**:
  - For j = 1, 2, 3: dp[1][j] = false (4 cannot form these sums).
  - For j = 4: dp[1][4] = true (4 forms sum 4).
  - For j = 5 to 10: dp[1][j] = false.
- **Row 2 (i = 2, element = 2)**:
  - For j = 1: dp[2][1] = false.
  - For j = 2: dp[2][2] = true (2 forms sum 2).
  - For j = 4: dp[2][4] = true (Subset {4}).
  - For j = 6: dp[2][6] = true (Subset {4, 2}).
  - For j = 7 to 10: dp[2][j] = false.
- **Row 3 (i = 3, element = 7)**:
  - For j = 7: dp[3][7] = true (7 forms sum 7).
  - For j = 9: dp[3][9] = true (Subset {2, 7}).
  - For j = 10: dp[3][10] = true (Subset {4, 7}).
- **Row 4 (i = 4, element = 1)**:
  - For j = 1: dp[4][1] = true (1 forms sum 1).
  - For j = 10: dp[4][10] = true (Subset

{4, 7}).

- **Row 5 (i = 5, element = 3)**:
  - For j = 10: dp[5][10] = true (Subset {4, 3, 3}).

Final DP Table:

```
dp = {
   {T, F, F, F, F, F, F, F, F, F, F},
   {T, F, F, F, T, F, F, F, F, F, F},
   {T, F, T, F, T, F, T, F, F, F, F},
   {T, F, T, F, T, F, T, T, F, T, T},
   {T, T, T, F, T, T, T, T, T, T, T},
}
```

Output:-
True
dp[n][target] is dp[5][10] = true