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;
}
```

We have **array**:

arr = {4, 2, 7, 1, 3}, target = 10

We create a **dp table of size (n+1) x (target+1)**:
dp[i][j] → i is the first i elements, j is the sum.

Initial Table (Before Processing)

i\j	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	T	F	F	F	F	F	F	F	F	F	F
1											
2											
3											
4											
5											

- **dp[0][0] = true** → A sum of 0 can be achieved with an empty subset.
- **dp[0][j] = false** for j > 0 → No subset can sum up to a positive number with zero elements.

Step 2: Fill the Table

We iterate through i = 1 to n, updating dp[i][j].

Processing arr[0] = 4

We consider only element **4**.

- dp[1][4] = true (We can form sum 4 using {4})

i\j	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
0	T	F	F	F	F	F	F	F	F	F	F
1	T	F	F	F	T	F	F	F	F	F	F

Processing arr[1] = 2

Now considering {4,2}:

- dp[2][2] = true (Subset {2})
- dp[2][4] = true (Subset {4})
- dp[2][6] = true (Subset {4,2})

i\j	0	1	2	3	4	5	6	7	8	9	10
0	T	F	F	F	F	F	F	F	F	F	F
1	T	F	F	F	T	F	F	F	F	F	F
2	T	F	T	F	T	F	T	F	F	F	F

Processing arr[2] = 7

Now considering {4,2,7}:

- dp[3][7] = true (Subset {7})
- dp[3][9] = true (Subset {2,7})
- dp[3][10] = true (Subset {4,2,7})

i\j	0	1	2	3	4	5	6	7	8	9	10
0	T	F	F	F	F	F	F	F	F	F	F
1	T	F	F	F	T	F	F	F	F	F	F
2	T	F	T	F	T	F	T	F	F	F	F
3	T	F	T	F	T	F	T	T	F	T	T

Processing arr[3] = 1

Now considering {4,2,7,1}:

- dp[4][1] = true (Subset {1})
- dp[4][3] = true (Subset {2,1})
- dp[4][5] = true (Subset {4,1})
- dp[4][8] = true (Subset {7,1})

i\j	0	1	2	3	4	5	6	7	8	9	10
0	T	F	F	F	F	F	F	F	F	F	F
1	T	F	F	F	T	F	F	F	F	F	F
2	T	F	T	F	T	F	T	F	F	F	F
3	T	F	T	F	T	F	T	T	F	T	T
4	T	T	T	T	T	T	T	T	T	T	T

Processing arr[4] = 3

Including **3** confirms all sums, but **dp[5][10]** remains **true**.

Final Answer

	Since dp[5][10] = true, we **return true**, meaning a subset exists with the sum **10**. Output: True
Output:- True dp[n][target] is dp[5][10] = true	