```cpp
#include <vector>
#include <iostream>
#include <algorithm>
#include <cmath>
#include <climits>

using namespace std;

void subsetProblem(vector<int> &arr, int &n, int &range, vector<int> &subsetSumPossible)
{
    vector<vector<bool>> dp(n + 1, vector<bool>(range + 1, false));
    for (int i = 0; i < n + 1; i++)
    {
        for (int j = 0; j < range + 1; j++)
        {
            if (i == 0)
                dp[i][j] = false;
            if (j == 0)
                dp[i][j] = true;
        }
    }

    for (int i = 1; i < n + 1; i++)
    {
        for (int j = 1; j < range + 1; j++)
        {
            if (arr[i - 1] <= j)
                dp[i][j] = dp[i - 1][j - arr[i - 1]] || dp[i - 1][j];
            else
                dp[i][j] = dp[i - 1][j];
        }
    }
    int z;
    if (range % 2 == 0)
        z = (range / 2) + 1;
    else if (range % 2 != 0)
        z = (range + 1) / 2;

    for (int j = 0; j < z; j++)
    {
        if (dp[n][j] == true)
            subsetSumPossible.push_back(j);
    }
}

int main()
{
    int n = 4;
    vector<int> arr = {1, 6, 1, 5};

    int range = 0;
    for (int i = 0; i < n; i++)
        range += arr[i];

    cout << "Range is : " << range << endl;
    // s1+s2 == range
    // we will take s1 <= s2
    // after finding range, we have to calculate possible values of s1....
    // we will do it using subset sum problem...
    vector<int> subsetSumPossible;
    subsetProblem(arr, n, range, subsetSumPossible);

    for (int i = 0; i < subsetSumPossible.size(); i++)
        cout << subsetSumPossible[i] << " ";

    int siz = subsetSumPossible.size();
    int mn;
    // first half elements of subsetSumPossible are for s1 and other half for s2....
    for (int i = 0; i < siz; i++)
    {
        int x = 2 * subsetSumPossible[i];
        mn = min(mn, range - x);
```

```
        }
        cout << endl
             << "Min subset sum difference is : " << mn;
        return 0;
    }
```