// CPP program for Dynamic Programming

// approach of SumOverSubsets DP

#include <bits/stdc++.h>

using namespace std;

const int N = 1000;

// function to print the sum over subsets value

void SumOverSubsets(int a[], int n) {

    // array to store the SumOverSubsets

    int sos[1 << n] = {0};

    int dp[N][N];

    // iterate for all possible x

    for (int x = 0; x < (1 << n); x++)

    {

        // iterate till n

        for (int i = 0; i < n; i++)

        {

            // if i-th bit is set

            if (x & (1 << i))

            {

                if (i == 0)

                    dp[x][i] = a[x] + a[x ^ (1 << i)];

                else // dp recurrence

                     dp[x][i] = dp[x][i - 1] +

                                dp[x ^ (1 << i)][i - 1];

            }

            else // i-th bit is not set

            {

                if (i == 0)

                    dp[x][i] = a[x]; // base condition

                else

                    dp[x][i] = dp[x][i - 1]; // dp recurrence

            }

        }

        // stores the final sum of subset of mask x

        sos[x] = dp[x][n - 1];

    }

    // print all the sum of subsets

    for (int i = 0; i < (1 << n); i++)

        cout << sos[i] << " ";

}

// Driver Code

int main()

{

    int a[] = {7, 12, 14, 16};

    int n = 2;

    SumOverSubsets(a, n);

    return 0;

}