**Program for Sum of Subset**

**#include <stdio.h>**

**#include <stdlib.h>**

**static int total\_nodes;**

**// prints subset found**

**void printSubset(int A[], int size)**

**{**

**for (int i = 0; i < size; i++)**

**{**

**printf("%\*d", 5, A[i]);**

**}**

**printf("\n");**

**}**

**// qsort compare function**

**int comparator(const void \*pLhs, const void \*pRhs)**

**{**

**int \*lhs = (int \*)pLhs;**

**int \*rhs = (int \*)pRhs;**

**return \*lhs > \*rhs;**

**}**

**// inputs**

**// s - set vector**

**// t - tuplet vector**

**// s\_size - set size**

**// t\_size - tuplet size so far**

**// sum - sum so far**

**// ite - nodes count**

**// target\_sum - sum to be found**

**void subset\_sum(int s[], int t[],**

**int s\_size, int t\_size,**

**int sum, int ite,**

**int const target\_sum)**

**{**

**total\_nodes++;**

**if (target\_sum == sum)**

**{**

**// We found sum**

**printSubset(t, t\_size);**

**// constraint check**

**if (ite + 1 < s\_size && sum - s[ite] + s[ite + 1] <= target\_sum)**

**{**

**// Exclude previous added item and consider next candidate**

**subset\_sum(s, t, s\_size, t\_size - 1, sum - s[ite], ite + 1, target\_sum);**

**}**

**return;**

**}**

**else**

**{**

**// constraint check**

**if (ite < s\_size && sum + s[ite] <= target\_sum)**

**{**

**// generate nodes along the breadth**

**for (int i = ite; i < s\_size; i++)**

**{**

**t[t\_size] = s[i];**

**if (sum + s[i] <= target\_sum)**

**{**

**// consider next level node (along depth)**

**subset\_sum(s, t, s\_size, t\_size + 1, sum + s[i], i + 1,**

**target\_sum);**

**}**

**}**

**}**

**}**

**}**

**// Wrapper that prints subsets that sum to target\_sum**

**void generateSubsets(int s[], int size, int target\_sum)**

**{**

**int \*tuplet\_vector = (int \*)malloc(size \* sizeof(int));**

**int total = 0;**

**// sort the set**

**qsort(s, size, sizeof(int), &comparator);**

**for (int i = 0; i < size; i++)**

**{**

**total += s[i];**

**}**

**if (s[0] <= target\_sum && total >= target\_sum)**

**{**

**subset\_sum(s, tuplet\_vector, size, 0, 0, 0, target\_sum);**

**}**

**free(tuplet\_vector);**

**}**

**int main()**

**{**

**int i, size, target;**

**printf("Enter number of weights: \n");**

**scanf("%d",&size);**

**int weights[size];**

**printf("Enter weights: \n");**

**for(i = 0;i < size; i++){**

**scanf("%d",&weights[i]);**

**}**

**printf("Enter target: \n");**

**scanf("%d",&target);**

**generateSubsets(weights, size, target);**

**printf("\n Nodes generated: %d\n", total\_nodes);**

**return 0;**

**}**

**Output:**

