Vivekanand Education Society's
Institute of Technology
Approved by AICTE & Affiliated to University of Mumbai

**Artificial Intelligence and Data Science Department.**
AOA / Even Sem 2021-22 / Experiment 9.

YASH SARANG.
47 / D6AD.
EXPERIMENT - 9.

---

**Aim:** Write a program for the sum of subsets problem.

---

## Theory:

Given a set of non-negative integers, and a value sum, determine if there is a subset of the given set with a sum equal to a given sum.

Example:
**Input:** set[] = {3, 34, 4, 12, 5, 2}, sum = 9
**Output:** True
There is a subset (4, 5) with sum 9.

**Input:** set[] = {3, 34, 4, 12, 5, 2}, sum = 30
**Output:** False
There is no subset that add up to 30.

To solve the problem in Pseudo-polynomial time use the Dynamic programming approach.

So we will create a 2D array of size (arr.size() + 1) * (target + 1) of type boolean. The state DP[i][j] will be true if there exists a subset of elements from A[0….i] with sum value = ‘j’.

The approach for the problem is:

```
if (A[i-1] > j)
DP[i][j] = DP[i-1][j]
else
DP[i][j] = DP[i-1][j] OR DP[i-1][j-A[i-1]]
```

1. This means that if the current element has a value greater than the ‘current sum value’ we will copy the answer for previous cases
2. And if the current sum value is greater than the ‘ith’ element we will see if any of the previous states have already experienced the sum=’j’ OR any previous states experienced a value ‘j – A[i]’ which will solve our purpose.

The below simulation will clarify the above approach:
set[]={3, 4, 5, 2}
target=6

| | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
|---|---|---|---|---|---|---|---|
| 0 | T | F | F | F | F | F | F |
| 3 | T | F | F | T | F | F | F |
| 4 | T | F | F | T | T | F | F |
| 5 | T | F | F | T | T | T | F |
| 2 | T | F | T | T | T | T | T |

**Time Complexity:** O(sum*n),
where the sum is the ‘target sum’ and ‘n’ is the size of the array.

**Auxiliary Space:** O(sum*n), as the size of the 2-D array, is sum*n.

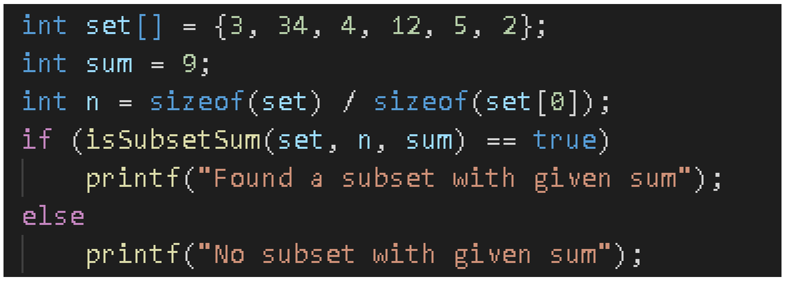
---

## CODE:

Code is in the Sum_of_Subsets.c file attached along with this doc.

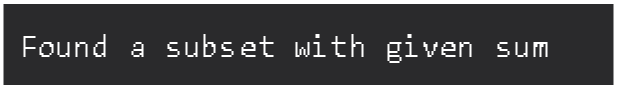
---

## INPUT:



```
int set[] = {3, 34, 4, 12, 5, 2};
int sum = 9;
int n = sizeof(set) / sizeof(set[0]);
if (isSubsetSum(set, n, sum) == true)
    printf("Found a subset with given sum");
else
    printf("No subset with given sum");
```

## OUTPUT:



```
Found a subset with given sum
```

---

## CONCLUSION:

By performing this experiment, I can conclude that the worst-case time complexity of the Sum of sub-sets problem is significantly lower while implementing using the Dynamic Programming Approach.

It has the time complexity O(sum*n), which  is very small as compared to the exponential complexity in the Naive implementation.

---