CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

UNIVERSITY INSTITUTE OF ENGINEERING

Department of Computer Science & Engineering

Subject Name: DAA Lab

Subject Code: 20ITP-312

Submitted to:

Er. Hemant kumar saini

Name: Ruchika Raj

UID: 20BCS9285

Section: 20BCS-WM-615

Group: B

Worksheet Experiment – 2.2

Name: Ruchika Raj **UID: 20BCS9285**

Branch: BE-CSE **Section/Group: 20BCS615 B**

Subject: DAA Lab

1. Aim/Overview of the practical:

To implement subset-sum problem using Dynamic Programming .

2. Task to be done/ Which logistics used:

find whether or not there exists any subset of the given set .

3. Algorithm/Flowchart:

i. We create a boolean subset[][] and fill it in bottom up manner.

ii. The value of subset[i][j] will be true if there is a subset of set[0..j-1] with sum equal to i., otherwise false.

iii. subset[i][j] = true if there is a subset with:

iv. the i-th element as the last element * sum equal to j

v. subset[i][0] = true as sum of {} = 0 vi. subset[0][j] = false as with no elements we can get no sum vii. subset[i][j] = subset[i-1][j-E1]; where E1 = array[i-1] viii. Finally, we return subset[n][sum].

4. Steps for experiment/practical/Code:

```
#include <iostream>
using namespace std;

bool subsetsum_DP(int a[], int n, int sum)
{
    bool dp[n + 1][sum + 1];
    int i, j;
```

```
    for (i = 0; i <= n; i++)
        dp[i][0] = true;

    for (j = 1; j <= sum; j++)
        dp[0][j] = false;

    for (i = 1; i <= n; i++)
    {
        for (j = 1; j <= sum; j++)
        {
            if (dp[i - 1][j] == true)
                dp[i][j] = true;
            else
            {
                if (a[i - 1] > j)
                    dp[i][j] = false;
                else
                    dp[i][j] = dp[i - 1][j - a[i - 1]];
            }
        }
    }
    return dp[n][sum];
}
int main()
{
    cout << endl
         << "This worksheet belongs to Ruchika Raj (20BCS9285)\n";

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

5. Observations/Discussions/ Complexity Analysis:

- Worst case time complexity: Θ(n*sum)
- Space complexity: Θ(sum)

6. Result/Output/Writing Summary:

```
3 4 5 6 7
The minimum number of multiplication operations required to multiply the matrix  chain is: 276
PS D:\CU\3rd Year\Sem 5\DAA\Worksheet\Unit 2> cd "d:\CU\3rd Year\Sem 5\DAA\Worksheet\Unit 2\" ; if

This worksheet belongs to Ruchika Raj (20BCS9285)
Found a subset with given sum
PS D:\CU\3rd Year\Sem 5\DAA\Worksheet\Unit 2>
```

Learning Outcomes:-

1. Create a program keeping in mind the time complexity
2. Create a program keeping in mind the space complexity
3. Steps to make optimal algorithm
4. Learnt about how to implement subset sum problem using dynamic programming.