



NAME – RAJDEEP JAISWAL
UID – 20BCS2761
BRANCH – CSE BTECH
SEC – WM 902 B

Worksheet Experiment – 2.2

Name: Anant Kumar Mathur
Branch: BE-IT
Semester: 5th

UID: 20BET1071
Section/Group: 20BET_WM-601-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 $\text{subset}[i][j]$ will be true if there is a subset of $\text{set}[0..j-1]$ with sum equal to i , otherwise false.
- iii. $\text{subset}[i][j] = \text{true}$ if there is a subset with:
- iv. the i -th element as the last element * sum equal to j
- v. $\text{subset}[i][0] = \text{true}$ as sum of $\{\} = 0$ vi. $\text{subset}[0][j] = \text{false}$ as with no elements we can get no sum
- vii. $\text{subset}[i][j] = \text{subset}[i-1][j-E1]$; where $E1 = \text{array}[i-1]$ viii. Finally, we return $\text{subset}[n][\text{sum}]$.

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() {
    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: $\Theta(n \cdot \text{sum})$
- Space complexity: $\Theta(\text{sum})$

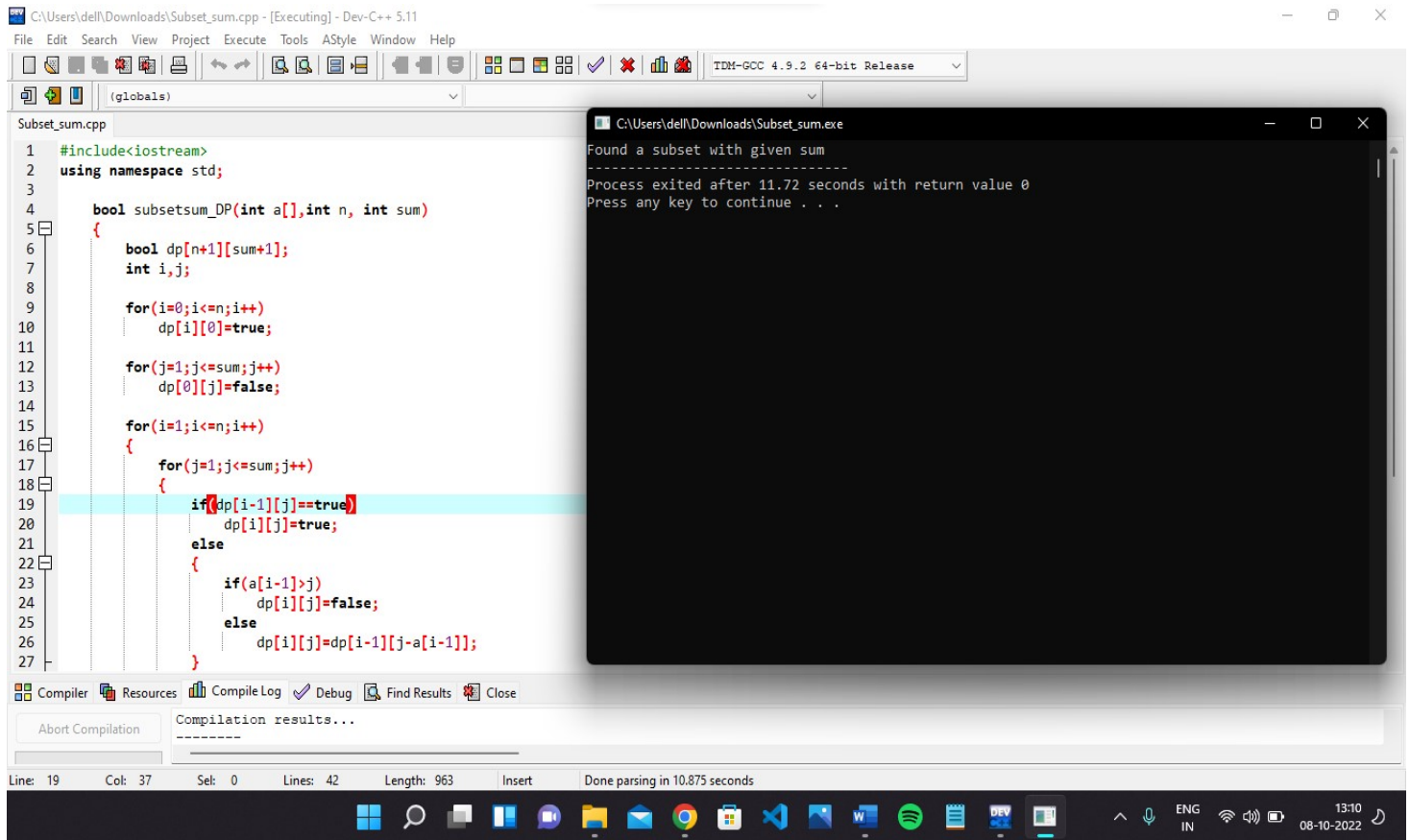
6. Result/Output/Writing Summary:



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
C:\Users\dell\Downloads\Subset_sum.exe
Found a subset with given sum
-----
Process exited after 11.72 seconds with return value 0
Press any key to continue . . .
```



```

1 #include<iostream>
2 using namespace std;
3
4 bool subsetsum_DP(int a[],int n, int sum)
5 {
6     bool dp[n+1][sum+1];
7     int i,j;
8
9     for(i=0;i<=n;i++)
10        dp[i][0]=true;
11
12    for(j=1;j<=sum;j++)
13        dp[0][j]=false;
14
15    for(i=1;i<=n;i++)
16    {
17        for(j=1;j<=sum;j++)
18        {
19            if(dp[i-1][j]==true)
20                dp[i][j]=true;
21            else
22            {
23                if(a[i-1]>j)
24                    dp[i][j]=false;
25                else
26                    dp[i][j]=dp[i-1][j-a[i-1]];
27            }
28        }
29    }
30 }

```

Terminal Output:

```

C:\Users\dell\Downloads\Subset_sum.exe
Found a subset with given sum
-----
Process exited after 11.72 seconds with return value 0
Press any key to continue . . .

```

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.