



EXPERIMENT 2.2

STUDENT NAME: Yash Kumar

UID: 20BCS9256

BRANCH: CSE

SECTION: 616 'B'

SUBJECT NAME: DAA LAB

SUBJECT CODE: 21CSP-312

AIM/OVERVIEW OF THE PRACTICAL:

To implement subset-sum problem using Dynamic Programming.

TASK TO BE DONE/WHICH LOGISTICS USED:

To implement subset sum problem using dynamic programming and problem-solving skills with the designing of the algorithm.

ALGORITHM/ FLOWCHART:

A naive solution would be to cycle through all subsets of n numbers and, for every one of them, check if the subset sums to the right number. The running time is of order $O(2^n \cdot n)$ since there are 2^n subsets, and to check each subset, we need to sum at most n elements. A better exponential-time algorithm uses recursion. Subset sum can also be thought of as a special case of the 0–1 Knapsack problem. For each item, there are two possibilities:

1. Include the current item in the subset and recur for the remaining items with the remaining total.
2. Exclude the current item from the subset and recur for the remaining items.

Finally, return true if we get a subset by including or excluding the current item; otherwise, return false. The recursion's base case would be when no items are left, or the sum becomes negative. Return true when the sum becomes 0, i.e., the subset is found.



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

STEPS FOR EXPERIMENT/PRACTICAL/CODE:

```
#include<bits/stdc++.h>
using namespace std;

bool sumSet(int* arr,int sum,int n)
{
    if (sum==0){
        return true;
    }
    if (n==0 && sum!=0){
        return false;
    }
    //if arr[i]>j//EXACT NEECHE COPY HOGA
    if (arr[n-1]>sum)
    {
        return sumSet(arr,sum,n-1);
    }
    //Reverse Checking
    //ACCORDING TO THE TOP ELEMENT IF TRUE THEN AS IT IS, IF FALSE THEN
    //REVEAL TILL THE Nth ELEMENT.
    return sumSet(arr,sum,n-1)||sumSet(arr,sum-arr[n-1],n-1);
}

int main ()
{
    int size;
    cout<<"Enter The Size Of The Array = ";
    cin>>size;
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

```
int arr[size];
cout<<"Enter The Elements In The Array = ";
for (int i=0;i<size;i++)
{
    cin>>arr[i];
}
int sum;
cout<<"Enter Targeted Sum = ";
cin>>sum;
if(sumSet(arr,sum,size)==true){
    cout<<"Sum Found."<<endl;
}
else
{
    cout<<"Sum Not Found."<<endl;
}
return 0;
getchar();
}
```

OUTPUT:

```
Enter The Size Of The Array = 3
Enter The Elements In The Array = 1
2
< 3
Enter Targeted Sum = 6
Sum Found.
```

```
Enter The Size Of The Array = 5
Enter The Elements In The Array = 1
2
< 3
54
5
Enter Targeted Sum = 23
Sum Not Found.

...Program finished with exit code 0
Press ENTER to exit console.
```



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

OBSERVATIONS/DISCUSSIONS/ COMPLEXITY ANALYSIS:

- $O(N \cdot \text{sum})$ where, N is the size of the array.
- Space Complexity: $O(N \cdot \text{sum})$.

LEARNING OUTCOMES:

- a) Learnt about dynamic programming.
- b) Learnt about design and algorithm analysis.
- c) Learnt about calculating time complexity of the subset sum problem using dynamic programming.
- d) Learnt about recursion.
- e) Learnt about initiating the base case of the condition.