

## EXPERIMENT 12

**Experiment No:** 12

**Date:** 05/05/2021

**Aim:** Implementation of Sum of Subset problem  
(Using Backtracking)

**Theory:**

### Subset Sum Problem

- The Subset-Sum Problem is to find a subset 's' of the given set  $S = (S_1 S_2 S_3 \dots S_n)$  where the elements of the set  $S$  are  $n$  positive integers in such a manner that  $s' \in S$  and sum of the elements of subset 's' is equal to some positive integer 'X.'
- The Subset-Sum Problem can be solved by using the backtracking approach.
- In this implicit tree is a binary tree, the root of the tree is selected in such a way that represents that no decision is yet taken on any input.
- We assume that the elements of the given set are arranged in increasing order:

$$S_1 \leq S_2 \leq S_3 \dots \leq S_n$$

- The left child of the root node indicated that we have to include 'S<sub>1</sub>' from the set 'S' and the right child of the root indicates that we have to execute 'S<sub>1</sub>'.
- Each node stores the total of the partial solution elements.
- If at any stage the sum equals to 'X' then the search is successful and terminates.

## EXPERIMENT 12

- The dead end in the tree appears only when either of the two inequalities exists:

- The sum of  $s'$  is too large i.e.

$$s' + S_{i+1} > X$$

- The sum of  $s'$  is too small i.e.

$$s' + \sum_{j=i+1}^n S_j < X$$

### **Backtracking**

- In Backtracking algorithm as we go down along depth of tree we add elements so far, and if the added sum is satisfying explicit constraints, we will continue to generate child nodes further.
- Whenever the constraints are not met, we stop further generation of sub-trees of that node, and backtrack to previous node to explore the nodes not yet explored.
- We need to explore the nodes along the breadth and depth of the tree.
- Generating nodes along breadth is controlled by loop and nodes along the depth are generated using recursion (post order traversal).

## EXPERIMENT 12

### Algorithm Writing

- Start with an empty set
- Add the next element from the list to the set
- If the subset is having sum M, then stop with that subset as solution.
- If the subset is not feasible or if we have reached the end of the set, then backtrack through the subset until we find the most suitable value.
- If the subset is feasible (sum of subset < M) then go to step 2.
- If we have visited all the elements without finding a suitable subset and if no backtracking is possible then stop without solution.

### Algorithm

Algorithm SumOfSub(s,k,r)

//w[1...n] is the weight array of size of n

//k is the chosen index

//r the is sum from k to n

//s the is sum from j to k-1  $w[j]*x[j]$

//assumptions 1 : w is in ascending order

2 :  $w[1] \leq m$

3 : Summation of 1 to n of w array  $\geq m$

{

$x[k] = 1$

if( $s+w[k] == \text{sum}$ )

## EXPERIMENT 12

```
then{  
  
    write(x[1...n])  
  
}  
  
else if(s+w[k]+w[k+1]<=m)  
  
{ //generate left child  
  
    SumOfSub(s+w[k],k+1,r-w[k])  
  
}  
  
//generate right child  
  
if(s+r-w[k]>=m and s+w[k+1]<=m)  
  
{ x[k] = 0  
  
    SumOfSub(s,k+1,r-w[k])  
  
}  
  
}
```

## EXPERIMENT 12

### Tracing with Example

Sum of Subset

$w[1:6] = \{5, 10, 12, 13, 15, 18\}$

$n = 6 \quad m = 30$

$B =$  Bounding function is nothing but a condition which has not been satisfied

$$\sum_{i=1}^K w_i x_i + w_{K+1} \leq m$$

$$\sum_{i=1}^K w_i x_i + \sum_{i=K+1}^n w_i > m$$

$x = [1, 1, 0, 0, 1, 0]$

$\therefore$  we can get multiple solution if we visit each and every node  
one solution is  $(5, 10, 15) = 30$

## EXPERIMENT 12

### Complexity

- Worst case time complexity:  $\Theta(2^n)$
- Space complexity:  $\Theta(1)$

### Program

```
#include<bits/stdc++.h>

using namespace std;

int c=0;

void SumofSub(int s,vector<int>,int k,int m );

void print(vector<int> ,int);

vector<int> v;

int a,sum;

int main()

{

    int m=0,s=0,k=1;

    cout<<"Enter The Value of N : "<<endl;

    cin>>a;

    v.resize(a+1,0);

    vector<int> x;

    x.resize(a+1,0);

    cout<<"\nEnter The elements : "<<endl;
```

## EXPERIMENT 12

```
for(int i=1;i<=a;i++)  
  
{  
  
    cin>>v[i];  
  
    m=m+v[i];  
  
}  
  
cout<<"\nEnter The sum : "<<endl;  
  
cin>>sum;  
  
for(int i=1;i<a;i++)  
  
{  
  
    c++;  
  
    for(int j=i;j<=a;j++)  
  
    {  
  
        c++;  
  
        c++;  
  
        if(v[i]>v[j])  
  
        {  
  
            int temp = v[i];  
  
            v[i] = v[j];  
  
            v[j] = temp;  
  
            c=c+3;
```

## EXPERIMENT 12

```
        }

    }

    c++;

}

c++;

SumofSub(s,x,k,m);

cout<<"*****"<<endl;

cout<<"Step Counts: "<<c<<endl;

cout<<"*****"<<endl;

}

void SumofSub(int s,vector<int> x,int k,int m )

{

    c++;

    if(k>a) return;

    c++;

    x[k] = 1;

    c++;
```



## EXPERIMENT 12

```
if(s+v[k]==sum)

{

    print(x,a);

}

else if(s+v[k]+v[k+1]<=sum)

{

    c++;

    SumofSub(s+v[k],x,k+1,m-v[k]);

}

c++;

if((s+m-v[k]>=sum)&&(s+v[k+1]<=sum))

{

    c++;

    x[k] = 0;

    SumofSub(s,x,k+1,m-v[k]);

}

}
```

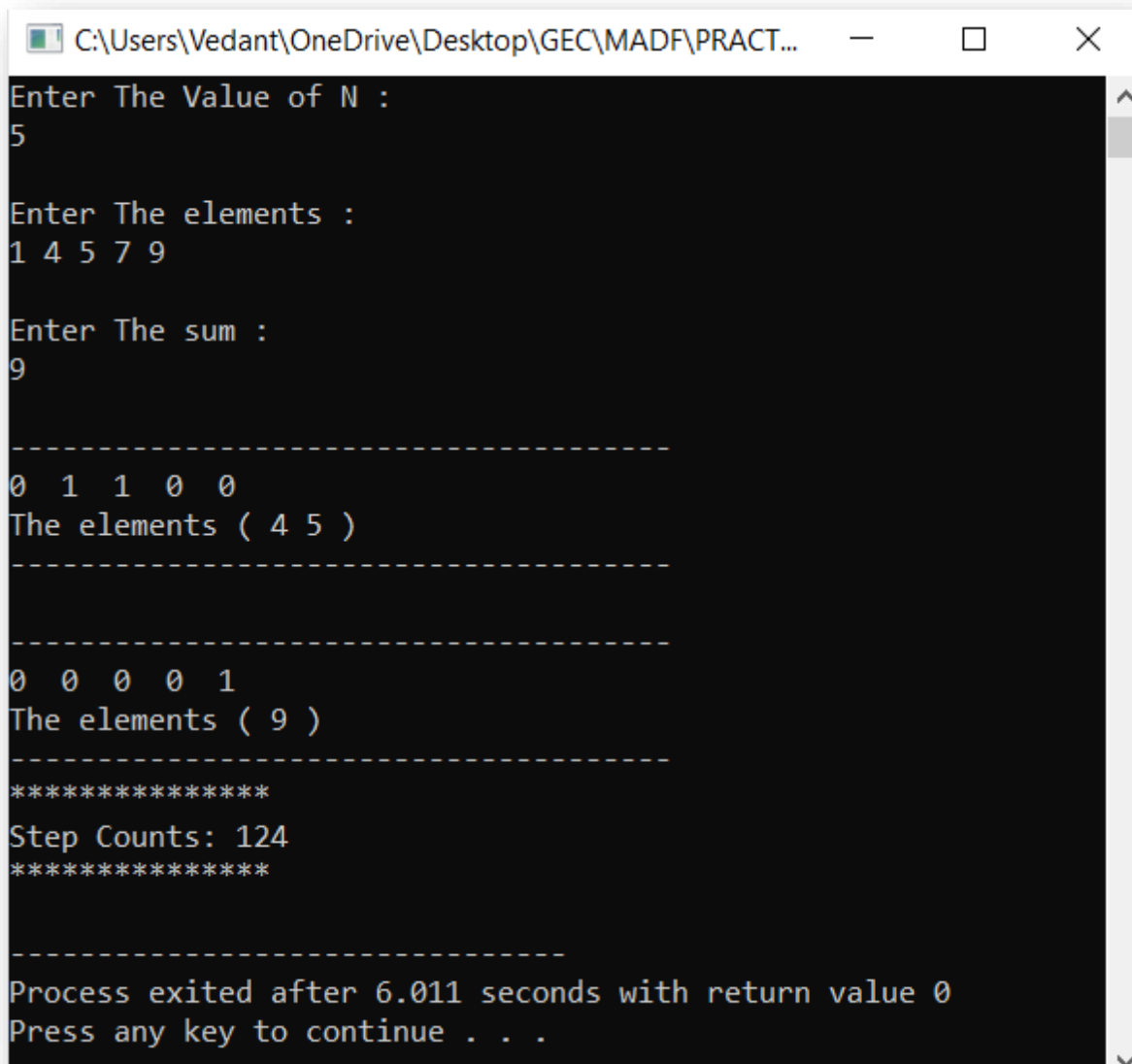
## EXPERIMENT 12

```
void print(vector<int> x,int a)
{
    cout<<endl<<"-----"<<endl;
    for(int i=1;i<=a;i++)
    {
        c++;
        c++;
        cout<<x[i]<<" ";
    }
    cout<<endl<<"The elements ( ";
    for(int i=1;i<=a;i++)
    {
        c++;
        c++;
        if(x[i]==1)
        {
            c++;
            cout<<v[i]<<" ";
        }
    }
}
```

## EXPERIMENT 12

```
cout<<"");  
  
cout<<endl<<"-----"<<endl;  
  
}
```

### Output



```
C:\Users\Vedant\OneDrive\Desktop\GEC\MADF\PRACT...  
Enter The Value of N :  
5  
  
Enter The elements :  
1 4 5 7 9  
  
Enter The sum :  
9  
  
-----  
0 1 1 0 0  
The elements ( 4 5 )  
-----  
  
-----  
0 0 0 0 1  
The elements ( 9 )  
-----  
  
*****  
Step Counts: 124  
*****  
  
-----  
Process exited after 6.011 seconds with return value 0  
Press any key to continue . . .
```

## EXPERIMENT 12

```
C:\Users\Vedant\OneDrive\Desktop\GEC\MADF\PRACT...
Enter The Value of N :
4
Enter The elements :
1 5 3 6
Enter The sum :
6
-----
1 0 1 0
The elements ( 1 5 )
-----
-----
0 0 0 1
The elements ( 6 )
-----
*****
Step Counts: 92
*****
-----
Process exited after 9.54 seconds with return value 0
Press any key to continue . . .
```

### Conclusion

- Detailed concept of Sum of Subset problem (Using Backtracking) was studied successfully.
- Program using Sum of Subset Algorithm was executed successfully.
- The step count for the Sum of Subset Algorithm was obtained.