EXPERIMENT – 6

Student Name: Gaganjot Singh **UID: 22BCS14843**
Branch: BE-CSE **Section/Group:** 22BCS_JT_802-B
Semester: 5th **Date of Performance:** 2024
Subject Name: Design & Analysis of Algorithms **Subject Code:** 22CSH-311

1. **Aim:** Develop a program and analyze complexity to implement subset-sum problem using Dynamic Programming.

2. **Objective:** To determine if there exists a subset within a given set of integers that sums up to specified target value. It does by generating possible subsets, calculating their sums, and checking if any subset matches target sum.

3. **Algorithm**
   - Start
   - Initialize the array and target sum
   - Iterate through all possible subsets using bit manipulation.
   - For each subset, calculate the sum by checking its bits.
   - If the sum of the current subset equals the target value, return true else false.
   - Continue process for other subsets.
   - If no subset matches the target sum after all iterations, return false.
   - Output the result based on the overall result.
   - End

4. **Implementation/Code:**

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

int subsetSum(int arr[], int n, int totalsum)
{for (int r = 1; r < (1 << n); ++r)
   {int subset_sum = 0;
    for (int i = 0; i < n; ++i)
      {if (r & (1 << i))
         {subset_sum += arr[i];}
      }
      if (subset_sum == totalsum)
      {return true;}}
   return false;}

int main(){int n;
   cout << "Enter the number of elements in the array: ";
```

```
cin >> n;

int *arr = new int[n];
cout << "Enter the array elements: ";
for (int i = 0; i < n; ++i)
{cin >> arr[i];}

int value;
cout << "Enter the target sum: ";
cin >> value;

if (subsetSum(arr, n, value))
{cout << "Found a subset with sum " << value << endl;}
else
{cout << "No subset with sum " << value << " exists" << endl;}
return 0;}
```

5. Output

```
Enter the number of elements in the array: 2
Enter the array elements: 4
8
Enter the target sum: 10
No subset with sum 10 exists
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS  SEARCH ERROR

PS E:\CU Study\22CSH 311 DAA\codes> cd "e:\CU Study\22CSH 311 DAA\co
Enter the number of elements in the array: 9
Enter the array elements: 1
4
526
8
5
95
41
2
5
Enter the target sum: 100
Found a subset with sum 100
PS E:\CU Study\22CSH 311 DAA\codes>
```

6. Time Complexity :

The time complexity of the code can be broken down into two main components:

1. **Input and Initialization**: The time complexity of reading the array elements and target sum from the user is O(n), where n is the number of elements in the array. This is because we have a loop that iterates n times to read the array elements.
2. **subsetSum Function**: The time complexity of the subsetSum function is O(2^n), where n is

the number of elements in the array. This is because the function iterates over all possible subsets of the array, which is 2^n in total.

Therefore, the overall time complexity of the code is O(n + 2^n).

7. Learning Outcomes:

- Understood brute force approach to solve the subset sum problem, which involves iterating over all possible subsets of the array.
- Analyzing the time complexity of an algorithm, including identifying the dominant term and calculating the overall time complexity.
- Tradeoff between time and space complexity, as the dynamic programming approach used in the subsetSum function reduces the time complexity but increases the space complexity.
- Understanding of Dynamic Programming, which involves breaking down a complex problem into smaller subproblems and solving each subproblem only once.
- Exploring the state space of possible subsets of the array, which is a key concept in dynamic programming.