Count the Subset Sum

Problem link - https://practice.geeksforgeeks.org/problems/perfect-sum-problem5633/1

Flow

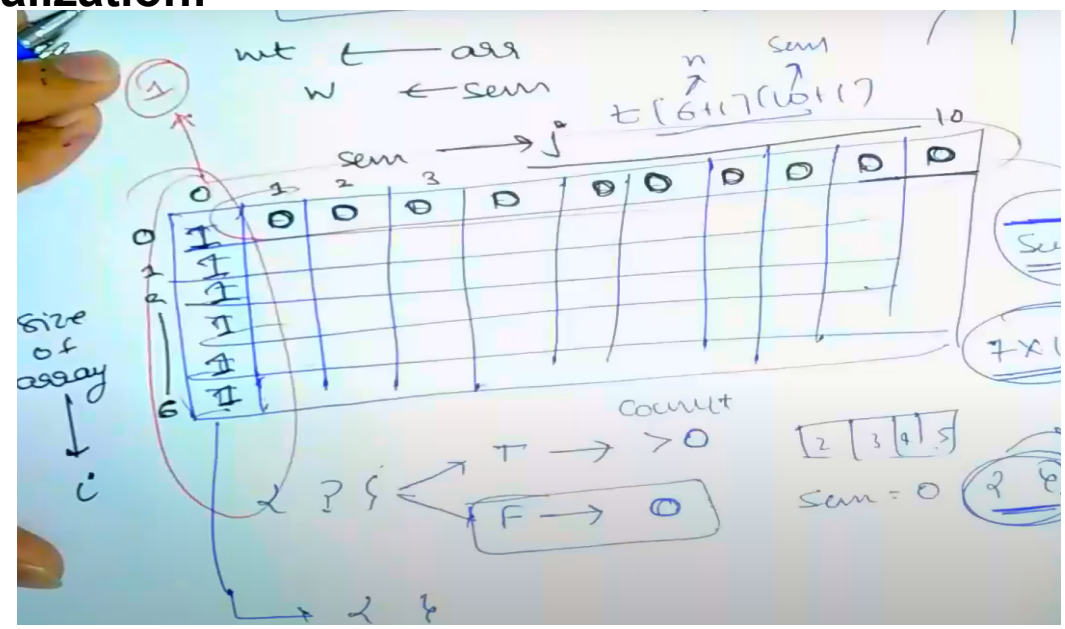
1. **Problem statement**
2. **Similar to subset sum**
3. **Return type**
4. **Code variation**

Problem statement - Given an array arr[] of non-negative integers and an integer sum, the task is to count all subsets of the given array with a sum equal to a given sum.
Note: Answer can be very large, so, output answer modulo 109+7

Similar to Subset Sum

1. In subset sum there is also given a array and sum
2. In the subset sum we have to find if the sum is equal to the subset sum or not and return true or false. But In this case we have to return the number of the subset present in the array that is equal to sum.
3. In subset sum we only go to that number where we find the True , if we find the True we put this value and return it but in this case we have generated all the subset and checked whether the sum of the subset is equal to sum or not and also count the number.

Initialization:



Code Variation

Subset Sum main Code

```
for(int i=1; i<n+1; i++){
    for(int j=1; j<sum+1; j++){
          if(arr[i-1] <= j){
                dp[i][j] = (dp[i-1][j-arr[i-1]] || dp[i-1][j]) ;
             }
              else{
                  dp[i][j] = dp[i-1][j];
              }
     }
 }
```

In this code “||” sign represents if we find the True then return that value else continue for the search of the True.
But what if we replace that “||” sign to “+” here + denote to go to the end and generate all the subset and find if that subset sum is equal to the given sum.

```
      public:
      int perfectSum(int arr[], int n, int sum)
      {
            long long mod=1e9+7;
        vector<vector<int>>dp(n+1 , vector<int>(sum +1, 0));
        for(int i = 0; i<n+1; i++){
            dp[i][0] = 1;
        }
        for(int i=1; i<n+1; i++){
            for(int j=0; j<sum+1; j++){
                if(arr[i-1] <= j){
                    dp[i][j] = (dp[i-1][j-arr[i-1]] + dp[i-1][j]) % mod  ;
                }
                else{
                    dp[i][j] = dp[i-1][j] % mod;
                }
            }
        }
        return dp[n][sum] % mod;
      }

};
```